1985

Program development using a team approach: Thesis document

Michele Miley

The University of Montana

Let us know how access to this document benefits you.
Follow this and additional works at: https://scholarworks.umt.edu/etd

Recommended Citation
https://scholarworks.umt.edu/etd/5547

This Thesis is brought to you for free and open access by the Graduate School at ScholarWorks at University of Montana. It has been accepted for inclusion in Graduate Student Theses, Dissertations, & Professional Papers by an authorized administrator of ScholarWorks at University of Montana. For more information, please contact scholarworks@mso.umt.edu.
COPYRIGHT ACT OF 1976

THIS IS AN UNPUBLISHED MANUSCRIPT IN WHICH COPYRIGHT SUBSISTS. ANY FURTHER REPRINTING OF ITS CONTENTS MUST BE APPROVED BY THE AUTHOR.

MANSFIELD LIBRARY
UNIVERSITY OF MONTANA
DATE: 1985
PROGRAM DEVELOPMENT USING A TEAM APPROACH:

THESIS DOCUMENT

by

Michele Miley

B. A., University of Montana, 1981

Presented in partial fulfillment of the requirements

for the degree of

Master of Science

Department of Computer Science

University of Montana

1985

Approved by

Chairman, Board of Examiners

Dean, Graduate School

Date
An emphasis has been placed today on the team development effort in the area of program development using a software engineering approach, in general because of increased program complexity and size. While the academic atmosphere tends to provide experience in individual efforts and group projects, little formal attempt is made to address difficulties encountered when programmers work from the specification or design created by another individual. This study addresses this topic through completion of a major programming project by two individuals, but rather than approaching it as a team effort, the project was divided into two logical portions and each team member completed one portion of the work in each phase, switching portions at the beginning of the next program development phase. The programming project used was the University of Montana Physical Plant Key Inventory Control System, a computer program requiring two logical portions, a front-office/bookkeeping portion, and a locksmith/key production portion. It was found that more complete and concise documentation is needed to continue from another individual’s work, and that while the task may be more difficult, it is a more realistic approach to current programming practices. Furthermore, while a dichotomy of tasks was achieved, it was found that occasional collaboration and consultation was necessary to clarify the logic and to standardize the final programming product. Working concurrently proved to produce higher motivation, both for meeting scheduled deadlines and for producing more detailed and explanatory support documentation of the product. Finally, because of the opportunity for cross-validation and verification of work produced, the final product was of higher quality than would have been achieved individually.
Table of Contents

Introduction .................................. 1
Method ......................................... 8
Results ........................................ 15
Discussion ................................. 20
References .................................. 23
Appendixes:
  Appendix A: Proposal .................... 24
  Appendix B: Feasibility Study .......... 39
  Appendix C: Requirements and
         Specification Document ............. 71
  Appendix D: Design Document .......... 175
  Appendix E: Test Documentation ........ 339
  Appendix F: User Manual ................. 399
  Appendix G: Program Code ............. 445
Introduction

Traditionally, creating a computer program has been an individual accomplishment. However, as program size and complexity has increased, it has become increasingly common to find programmers working in a group or team situation, or completing the work that has been begun by another individual [8]. Rapid employee turnover may also be a contributing factor to this trend. In addition, more attention has been paid recently to a "software engineering" approach to program development in which the entire task is broken into parts, with an emphasis of effort on early analysis and design of the program rather than effort at the coding phase [8][1].

The advantage of several people working on a project is that each person acts as a check on the others, and incomplete or insufficient ideas tend to be discovered early. In addition, one person may not be capable of conceptualizing the entire task when the program is very large, so it is broken into parts and several people or teams may complete the task [9]. The disadvantage is that
lines of communication are increased, and using too many people can actually slow a project down rather than increase productivity [2].

The use of software engineering techniques provides for early attention to problem definition and an approach to solving the problem prior to actual coding. This process not only clarifies the task at hand, but provides a method for breaking the task into parts. Also process provides documentation of the program to be developed before it is implemented [1][8]. This documentation is vital to those who will maintain the program after it is put into production [5].

The most often used software engineering approach to program development is that of the waterfall model. In this model, the program development process is broken into several phases [1]. The first is that of the system proposal and feasibility study. This provides a first glance assessment of the problem and an initial determination of the viability of producing software to solve that problem. At this phase the problem is not well defined, therefore much estimation must
be used [1][8]. In general, there are no formal techniques for preparing the proposal, however, several estimation techniques exist for use in the feasibility study. At best, however, these techniques produce very unreliable estimates because of the lack of information available at this stage [1][3].

The next phase is that of the software plan and requirements. In this stage, the problem is broken down to determine its subparts, and the current system is analyzed according to its operations and data flows. Once it is determined what the problem is and how the system is currently operating, it may be determined how software will be constructed to solve the problem. This phase, then, consists mostly of a partitioning of the system into manageable parts [1][8]. A popular approach is that of "structured analysis" where a system is partitioned by its data flows and functions, and control information is completely ignored [4]. While this works wonderfully for the informational portions of the system, it has been the experience of the author that it is insufficient when specifying the user interface portion
of a project. An alternative to the structured analysis approach is the use of Warnier-Orr diagrams. This may be more effective in specifying flow of control within the system. Once the system has been specified, a re-estimation of resources required may be made.

The third phase is that of design. There are two parts to design, high level, or product design, and low level, or detailed design. The product design provides a hierarchical breakdown of the system to be developed and provides logical methods of configuring the system. The detailed design depicts file structures, data structures, and algorithmic descriptions of each of the modules produced to solve a portion of the problem, in addition to a definition of the interfaces necessary between the different portions of the system [1][8]. "Structured design" is an approach using the structured analysis produced in the previous phase for breaking the system into afferent, efferent, and transform modules, and to minimize the coupling between modules and maximize internal module cohesion [9]. "Afferent" describes processes accepting data into the system,
"efferent" denotes those processes outputting data, and "transform" applies to the modules that change the data within the system. The final product of this phase is the program design, a detailed blueprint outlining the shape and content of the actual code to be produced.

Once the design has been completed, the coding phase begins. The code is produced from the algorithmic descriptions found in the design and follows the hierarchical structure laid out. Each module is coded and tested individually then integrated with other modules and tested again for correctness. Testing is vital to increase the reliability of the program, although it is very difficult to prove the correctness of a large program. One approach often used is where each path of the program is executed and tested for correct results. This is a type of "white box" testing where the tester must be aware of the structure of the code in order to test each portion of the program [7].

The final phase in program development occurs once the code has been fully tested. This is the
implementation phase where the program is put into production use. Data entry is performed at this phase, most often not by the persons developing the system. The hardware is completely set up, and if the project was developed on another machine than the target machine, the code is moved to the target machine [8].

Another phase occurs after implementation, that of operations and maintenance. However, this is not part of the initial program development. Although it is not always categorized with program development, maintenance can use as much as 70% of the total program cost [5]. While maintenance is not the subject of this paper, care has been taken to produce an easily maintainable program via the tools previously mentioned for analysis and design of the system in addition to complete and thorough documentation of methods used.

All these phases constitute the waterfall model of the software life-cycle. It is called "waterfall" because each phase produces information used by the next phase. In any one of the phases, it may be necessary to return to an ear-
lier stage if an error is discovered, however, with this approach, errors tend to be caught earlier and backtracking is much less than the traditional propose-and-code approach to program development [1][6]. In addition, emphasis is placed on early analysis and design effort, and then later testing effort, rather than the traditional emphasis on the coding phase of program development [8].

The purpose of this study, then was to use both the team approach and software engineering methodology in program development. Because the thesis is inherently an individual process, it was decided to split the program task into two portions and individually work on these, however, at each major step in the waterfall breakdown of phases, these tasks were to be switched, so that each individual worked from the specifications and design of the other. In addition, as has been recommended, the individual programmer would not test his/her own code [7], but have the other test it, so that a developer's bias would not be present in the testing phase. It was believed that this approach would slow development somewhat, but
would produce a higher quality product. Furthermore, each step of the process would be completed in greater detail to allow the person carrying on that portion of the project the thoroughness necessary to understand the underlying logic.

Method

The method used to do this task was, as mentioned previously, to divide a substantial programming project into two parts and have two individuals work on those parts, but at each major phase, these persons would switch tasks. The project chosen was the University of Montana Physical Plant Key Inventory Control System (KICS). Because it was logically divided into two subsections, a front-office/bookkeeping section and a locksmith/key production section, it was approved to be suitable for a two member team.

The front office portion of the system is required to handle several tasks, most of which involve data files of one type or another. Persons requesting keys are to be issued the keys, and records kept of the issuee and deposit paid, while persons returning keys are refunded their initial
deposit and their record of possessing a key is removed. Bookkeeping is necessary to keep track of the money transactions occurring in these instances. The level of keys maintained ready for issue has to be kept track of and orders given to the locksmith to make more keys as the supply runs low. With the manual filing system, it is impossible to trace each keyholder and request returns as they are leaving the University system. Furthermore, manual tracking of inventory levels is not always effective, and orders for production of more keys is usually made when it is obvious that there are no keys available for issue. Requests for information concerning who is issued what keys, or what group of people possessed which keys is often unavailable because of the manual filing system. The major purpose, then, of the front office portion of the system is to provide easier access to file information under program control.

The locksmith portion of the system is required to make keys as they are needed by the front office and to rekey locks as requested by other departments. Inventory of key blanks has to be maintained and records kept regarding which
keys previously used and now retired are available for use again. Service has to be performed on malfunctioning locks. While certain portions of this system are not amenable to automation, other parts are. The main purpose of the locksmith portion of the program, then, is to automatically generate new key numbers available for use in rekeying locks, and to track the inventory of key blanks. Furthermore, orders coming from the front office would be generated automatically as supplies became low, not as supplies became non-existent, producing a more uniform arrival of orders for key production.

Initially, the proposal and feasibility study were written as a joint effort. However, once these were completed, the work was split. The estimation method used to determine feasibility and amount of effort required was determined using the COCOMO estimation model [1]. The estimate indicated that two people would be required to complete the system on schedule (in fact, that the schedule for a two member development team would be rather rigorous). For details on the estimate and feasibility options covered, the reader is
referred to Appendix A and B, for copies of the proposal and feasibility study produced for this project. Because the system under consideration was primarily a data base system and the focus of the system was to improve key recovery by interfacing with other University files, it was at this time that it was decided to use the unstructured data base language of PL1022. This particular language was chosen because of its facility for file access and the fact that other University files would be accessible via this language [10].

During the requirements analysis phase, one person had the task of specifying the front office procedures, and the other, the task of specifying the locksmith procedures and writing the narrative associated with documentation. This was decided to be a fair dichotomy because the locksmith portion of the program was somewhat smaller than the front office portion. The approach used to specify the system was the "structured analysis" approach in which data flows, functions, and data definitions are specified and control information is not addressed [4]. The system was first partitioned according to current procedures, then into logical
partitions suggested by the current procedures which would be more amenable to automation. The reader is referred to Appendix C for the documentation produced by this method.

Once the specification was done, the author designed the front office portion of the system, and the other team member designed the locksmith section and wrote the associated narrative. The partitioning produced during the analysis phase was broken further into afferent, efferent, and transform processes, and structured design techniques were used to produce the modular structure of the system. An effort was made to maximize module cohesion and minimize intermodule coupling, and all process descriptions were designed to utilize only structured programming techniques of internal decision and flow of control [9]. At this point, an effort was made to use the third normal form of file structure to reduce the redundancy of the information in the files. However, because of disk space limitations, one file remained with a many to many relationship. Pseudocode was produced for each module to act as a guide for the logic to be used at programming time. While the logic of
each module was designed, user interactive error-checking was left for the individual programmer to add to the actual code [3]. The user interface was designed at this phase, and some collaboration was necessary in order to produce a uniform user interface across the two portions of the program. Appendix D contains the final draft of the design document which accurately reflects the structure of the final software product.

Again, as this was completed, the author coded the locksmith portion of the program and designed files to be used for testing the entire program, while the other coded the front office procedures. The structure of the module hierarchy was adhered to except in cases where it was found to be more elegant to change the structure because of user requests to change portions of the system. Each module was coded as a separate file and a method of delineating local and global variables was used to avoid global data problems and adhere to structured programming principles. Unit testing was performed by each of the individual coders on their own code before turning it over to the other for integration testing [7]. As the user interface
was coded and ready for viewing, the program was turned over to the eventual users to experiment with, and a period of time was allowed where the users could suggest changes they would like to see. A preliminary user manual was produced to aid the users in working with the program. Changes were made according to user requests. Appendix G contains the final code of the program while Appendix F contains the final user manual produced for those to be running the program.

Finally, the code was switched, and each individual tested the other person's code. The method of testing used was the white box method of testing each path of execution and determining that the program was functioning correctly according to the specifications produced earlier. A transcript of the final integration test of the code may be found in Appendix E. If future modification is performed on this system, it is suggested that that particular test sequence be run as a regression test to insure no errors were introduced into the code [7]. Although operational, the program has not yet been implemented, pending data entry.
Results

It was found that in general, knowledge that the work completed in one phase would be turned over to another to finish, produced a more detailed product. Several logic errors and incomplete portions were found, necessitating some collaboration, however this was minimal. Work on areas such as coding standards and user interface standardization was also found to require collaboration between the individuals involved.

Because the proposal and feasibility study portion of the project were completed as a joint effort, both individuals involved had an initial understanding of the entire system. Thus, when the specification portion of the system was begun, both started out with the prior knowledge necessary to complete the analysis phase. The only issues encountered during this process were those normally encountered during analysis, such as frequent contact with the user to verify current operations and the requirements for the system to be developed. Progress in this phase advanced as would be expected of any system despite parallel
development.

Completion of another's work did not become an issue until the design phase was reached, and clarification of some points was necessary to produce an accurate design. As the logical partitioning of the system was further divided to produce afferent, efferent, and transform demarcation of the system, some logical errors became apparent. Each time the partitioning of the future system was to be changed, the team members conferred and agreed on the change, rather than making arbitrary and unapproved changes. The specification document was updated to reflect the modifications made, although a log was not kept of the changes made. As previously mentioned, structured design techniques were used to produce the modular partitioning of the automated system, and as a result, a model of the system was produced that would be easily coded, tested, and maintained.

Again at the coding phase, clarification of some portions of the design was necessary. As agreed at the design phase, pseudocode did not include user interface error checking, therefore,
each programmer was free to code this portion herself. On some occasions, the pseudocode did not provide for the necessary decisions to be made, and this was modified by the individual programmer also. In only one instance was it necessary to change the structure of the modular layout, and this was in response to a user requested change. The modification of the design was updated in the design document. One result that became clear during coding was that each programmer had prior knowledge of the structure of the system because that programmer had specified that portion of the system. In some cases, when the pseudocode did not follow the specification process descriptions, this became confusing, although the logic tended to be correct in the pseudocode. Also, it was found that attention must be paid to the language at the design phase because of the inherent differences in language structure. In this case, the data base language of PL1022 provides low level routines such as sorting and file access [10] which need not be addressed at the design phase, whereas another language would have had to specify these functions. Because PL1022 is not a
structured language, and the design was based on a structured programming approach, a standardized method of delineating local and global variables was determined to avoid global data problems. It was found that the approach used was quite sufficient to avoid problems.

During testing, it was often necessary to revert to the specification and design documents to ensure that the code was functioning as intended, however, this is advocated in all testing [7], and is not unique to this project. It was found to be easier to test another person's programs because the tester was not attached to the code as her own creation. It was not as detached a process as testing completely unfamiliar code, however, because there was an inherent interest in seeing the program function correctly since it was all part of the same system. It appears that more errors were discovered in this manner. In addition, it was helpful to allow the users to have access to the program before it was fully implemented. This uncovered errors and omissions earlier than would have been discovered if the user had not seen the program until implementation
While the documentation tended to be more complete, working with another individual also tended to increase motivation to complete designated tasks by the scheduled deadline. Finally, the resultant product appears to be more complete and useful than if it were developed by one individual.
Discussion

In general, it appears that using a team approach to program development has the advantages of producing more complete and accurate documentation in addition to a superior final product. Furthermore, individual motivation and performance appear to be enhanced. Knowledge that another person is dependent upon the quality and timeliness of the work produced tends to positively influence both aspects of the product. The structure of this particular project may have increased the benefits found, however, because of the ease with which the program was split into two separate portions. Because these portions of the program interfaced only through files, and not through data passed, it was not necessary to spend a large amount of time specifying the program interfaces. Furthermore, effort on each of the portions could proceed with little collaboration except in the user interface and standards areas.

Unique, also, to this project, was the practice of exchanging tasks at each major phase in development. This method was chosen both because
one part of the system was somewhat smaller than the other, and because it would give each individual involved experience with the entire system. The more common case would be that team members would complete their entire portion of the project from start to finish. The approach used here seems to have added to the accuracy of the final product. However, may have slowed development time. This may not be a viable approach to system development in which rapid completion is a main objective.

In the future, it may be helpful to delineate those tasks that are best approached in a group manner and those that may be allocated individually, should this method be used again. The persons involved in this study were both familiar with both portions of the system because of the switching of tasks, but it may be of interest to allocate tasks to individuals who had no prior knowledge of the system being developed. This would provide a better test of system documentation accuracy.
In addition, the individuals working together had prior experience and respect for each other's quality of work, and therefore trusted the decisions made by the other. This situation may be different between individuals unfamiliar with each other's work, as may be encountered more often in the working environment.

In summary, then, the approach used slowed development time, but produced a final product of greater quality. Because the system was easily dichotomized into two relatively discrete parts, the same results may not be obtained on a more tightly coupled system. Similar systems should produce the same effects if approached in the same manner.
References


THESIS PROPOSAL:

KEY INVENTORY CONTROL

by

Robin M. Fauntleroy
Michele Miley

April 4, 1985
Table of Contents

Introduction ............................................. 26

Current System Description .............................. 27

Problems With the Current System ...................... 30

Proposed Areas of Solution ............................. 31

Justification ............................................. 34

Conclusion ................................................ 38
Introduction

Prior to the commencement of effort involved with a major undertaking such as a thesis project, clarification of the problem is attained through the written proposal. The purpose of this document is, therefore, to provide a general understanding of the problem as it exists, an overview of the scope of possible solutions, and a rationale for resolution. This proposal is prepared as a preliminary step toward the completion of a master's thesis project for the Department of Computer Science within the University of Montana.

The project under consideration has been requested by the University of Montana Physical Plant, the department responsible for campus security and maintenance. The need for an improved key inventory control system is recognized as a priority within the Physical Plant and by the State of Montana auditors. It was suggested that
computerization would meet this need.

Current System Description

There are two major divisions within the current key inventory control system. The first is the actual locksmith operation, in which not only are keys made, but records are kept of which keys open which locks. There are approximately 7000 locks on campus, including both door locks and padlocks. All types of locks are referred to as cylinders. Keys are sorted into four levels. The lowest level of key is a change key (CK) which will open at most only a few doors. The next level is the master key (M) which opens several doors. There are various kinds of master keys, such as those issued for faculty, staff, or departmental specialty. Above the master key is the grand master (GM) which may open all locks in an entire building and is issued for the most part to custodial staff. The highest level is the great grand master (GG). This key is not an issue
key, and is distributed exclusively to maintenance and security personnel. It has the capability of opening all cylinders in several buildings. Currently, there are 18 of these, but it is hoped that the number be reduced to 8 to 10 in the future.

Each key has two numbers on it, the first corresponding to the keyway, and the second reflecting the cut or bit numbers. The keyway refers to the slots on the side of the key, whereas the bit numbers refer to the actual cylinder arrangements of the lock which the key is cut for. The keyway prefix is given with each key blank, whereas the bit number is not stamped until the bit cut has been made. There are six points to each key where a bit cut may be made. On at least one cut per key, there must be a minimum of two bits difference from any other key. When a lock is rekeyed, the old key is retired for one year, prior to reuse.
The locksmith must maintain records of key numbers currently in use, those available for use, those retired, and the length of retirement. Furthermore, an inventory must be maintained of key blanks and other lock components. Presently, the locksmith is notified as particular keys are disbursed and supplies are depleted, thereby necessitating their duplication.

The second major portion of the key inventory system is key issue control. Records must be kept of all persons possessing a given key, the deposits received, keys issued to an individual, keys returned by an individual and deposit refunds. Although the procedure for this is relatively straightforward, the amount of information that must be processed is quite extensive. Accuracy is essential for university security. The only account of key distribution is maintained by this office.
Problems With The Current System

Recently, an unofficial audit was performed by State of Montana auditors on the key inventory control system, which suggested improvements might be in order. Because of the large number of keys involved, it has become extremely difficult and tedious to perform manual searches of records for desired information. Furthermore, much valuable employee time is consumed in this process. With the amount of transactions performed daily in the office, the possibility of an error exists. In addition, prompt service is often requested and may not be possible due to the current manual system.

Two factors contribute to the cost of the current system. One previously discussed, is that of employee time spent manually searching records. The second is the cost incurred when a key is not recovered. The cost of rekeying a building due to the loss of a grand master may easily approach
$20,000. Should a great grand master be lost, the cost would be phenomenal. With current procedures, recovery of keys is not always possible as people leave the university system.

Proposed Areas of Solution

It is proposed that a computerized system would address the following areas:

1. A key inventory control system, to provide instantaneous information regarding the number of any one key available to be issued, the number of those keys already issued, a threshold number for ordering the duplication of individual keys, and the number of duplications to be made.

2. A parts inventory control, which would maintain a count of all key blanks for each keyway. Yearly, an inventory re-order list would be generated. In addition, a count of lock components would also be maintained.
3. Bit pattern generation, of numbers available to use for any one lock, providing a number for the lock which would be compatible with the master keys opening the lock in question. This generation of numbers would also take into account any numbers which are on the retired list, and requirements concerning bit cuts.

4. A rekey history would provide a means for tracing the history of a lock which was rekeyed.

5. Database information would include files containing information on what keys an individual is issued, which individuals had a particular key, what keys open what doors, what keys belong to a building, what keys (or doors) a master key would open, and how long a key has been out.

6. Deposit accounting would trace the deposits entering the system and the refund of deposits after the return of a key.
7. Expenditure accounting would provide the department with a trace of expenditures on inventory and other aspects of the system.

8. Interface with current administrative programs. An interface with payroll is possible to allow for tracking of keys when an employee, staff or instructor, with a key leaves the university. Furthermore, an interface with the registrar's office would provide lists of those persons graduating, to cross-reference with outstanding keys.

9. Correspondence generation, to provide for an automatic generation of form letters requesting the return of keys prior to graduation, or summer break, etc.

10. Information security, providing for restricted access to the key inventory system. Accessibility of information within different parts of the system would be limited.
Justification

Due to the nature of this project, involving the security of the university, a complete and thorough job of the entire life cycle is indicated. The size of the project, items requested, and the expectation of a system that is operational, thoroughly tested, and complete with detailed documentation and a user’s manual indicate that the project could be allocated to two people.

The following reasons are presented for working on this project as a team:

1. A well written, documented system meeting the needs and requirements of the Physical Plant, as has been requested, would enhance relations between the computer science department and other departments involved.

2. Provides graduate students an opportunity to work on a team project, defined by an outside
group. This will generally be the method of producing a system in future career situations.

3. Provides for rigorous and impartial testing, as functions, procedures, subsystems, etc. can be tested by the other member of the team, who is not personally 'attached' to the code. This will provide for a more reliable and bug-free program for the Physical Plant. Because this system will provide for the security of the entire University, it's correct operation is essential.

4. Provides a cross-check on validation of the stages involved with the life-cycle of the system to ensure the writing of the 'right' program.

5. Cross-check on verification of the stages involved with the life-cycle of the system to ensure writing the program correctly.

6. Assurance of program sufficiency for the user. Because cross-checks can be performed at
various points within the life-cycle of the project, precision and depth of work in each phase can be expected, leading to better assurance of user satisfaction with the system. Furthermore, complete, detailed and correct documentation will be a result of periodic review by someone besides the author of the documentation. Finally, the more rigorous approach to the life-cycle will produce a product which is easily maintainable in the future.

7. Required effort estimation. There are ten proposed areas of solution to the problem, and each represents a separate program. At an average of 500 lines per program, this produces an overall estimate of 5,000 lines of code at completion. Using formulas as discussed in the COCOMO model of estimation, this yields 13 man-months of work, with a time for development of 6.6 months, and an average of 2 full time persons working on the project. The formulas are as follows:
8. Specific division of labor. The structure of the problem is such that there are two major areas to be addressed. These are the locksmith function areas and the record-keeping, or front office procedures. With this dichotomy, each team member may work on a discrete portion of the project in the development phase with the added advantage of cross-checking at validation and verification checkpoints.
Conclusion

In conclusion, several major points stand out from the foregoing discussion as being particularly important in the proposed project. First of all is the critical nature of the problem. Since campus security depends upon the correct and complete resolution of the problem, care must be taken to provide appropriate and thorough solutions. This can be provided via a joint effort and system of cross-checks throughout the development of the system. Second, as demonstrated by the COCOMO model of estimation, the amount of labor required is sufficient to justify the work contribution of two persons. And finally, a commitment to complete the project and produce a fully documented and tested system to be used by the Physical Plant by August, 1985, requires that enough manpower be provided to ensure successful conclusion of the project.
FEASIBILITY STUDY:

KEY INVENTORY CONTROL

by

Michele Miley

Robin Fauntleroy

April 29, 1985
# Table of Contents

Introduction .................................................. 41  
Management Summary and Recommendations .............. 42  
Current System Description .................................. 44  
  Overview ................................................. 44  
  Constraints ............................................. 46  
  Prioritized Subsystems ................................. 47  
Alternative Solutions ........................................ 50  
  DEC 2065 .................................................. 50  
  VAX 11/785 ............................................... 57  
  TRS 80 ..................................................... 57  
  Combination ............................................. 61  
  HP 150 ...................................................... 63  
  IBM XT ...................................................... 65  
  Apple IIe ............................................... 67  
Recommendation and Justification ......................... 68  
Proposed Schedule of Development ......................... 69
Introduction

Some purposes for the completion of a feasibility study are to determine whether there is a new way of doing business that justifies the expense of a project, and to document the parameters that would govern such a project. The development of the feasibility document also benefits the user in that it will determine if the system specified can be implemented within given constraints, assures that the project the user wants will be developed within the forementioned constraints, and assures that the user and developer have the same time and cost expectations. Furthermore, it will determine if time, money and other resources are adequately available to complete a given project and provide a milestone from which the next phase of the software life cycle may proceed. Finally, some of the purposes are to determine whether the objectives stated in the proposal are attainable, and if not, what constraints must be removed, to define the major problems existing so that the systems analyst can plan his strategy for the full investigation, to define the areas where potential
exists for making savings of money time or effort, and to decide if specialists will be needed to render assistance in the full investigation.

In summary, the feasibility study is an integral part of the production of a viable software product. If researched thoroughly, the conclusions reached will serve as a framework for future stages in system development.

A proposal was submitted, April 4, 1985, to involved parties, outlining the keys inventory control project under consideration. This document is a feasibility study of that system containing possible alternative solutions, evaluation of those alternatives, and recommendations.

Management Summary and Recommendations

Based on the following discussion, it is the recommendation of this study that the first alternative discussed, that of implementing the entire system on the DEC 2065 is the most feasible solu-
tion. Although some of the assumptions made concerning the TRS 80 may be realized at a future date, current conditions are such that a decision should not be based on these assumptions. The adoption of this alternative, of course, is contingent upon approval by the Computer Center. Along the same lines, alternative options discussed which involve the purchase of a microcomputer, assume that the purchase could be approved and made in a timely manner. As these conditions cannot be guaranteed, basing a decision on these assumptions would not be warranted.

In summarizing the reasons for recommending this alternative, it has been found that the DEC 2065 provides the most amenable environment for the project development. User access to this system is satisfactory, along with response time, and program interface with administrative systems is feasible. Increased efficiency of key issue control may be realized via this alternative. Furthermore, preliminary estimates of disk space
requirements are in a reasonable range.

Current System Description

Overview

The project under consideration has been requested by the University of Montana Physical Plant, the department responsible for campus security and maintenance. The need for an improved key inventory control system is recognized as a priority within the Physical Plant and by the State of Montana auditors. It was suggested that computerization would meet this need.

There are two major divisions within the current key inventory control system. The first is the actual locksmith operation, in which not only are keys made, but records are kept of which keys open which locks. The locksmith must maintain records of key numbers currently in use, those available for use, those retired, and the length of retirement. Furthermore, an inventory
must be maintained of key blanks and other lock components. Presently, the locksmith is notified as particular keys are disbursed and supplies are depleted, thereby necessitating their duplication.

The second major portion of the key inventory system is key issue control. Records must be kept of all persons possessing a given key, the deposits received, keys issued to an individual, keys returned by an individual and deposit refunds. Although the procedure for this is relatively straightforward, the amount of information that must be processed is quite extensive. Accuracy is essential for university security. The only account of key distribution is maintained by this office.

Recently, an unofficial audit was performed by State of Montana auditors on the key inventory control system, which suggested improvements might be in order. Because of the large number of keys involved, it has become extremely difficult and
tedious to perform manual searches of records for desired information. Furthermore, much valuable employee time is consumed in this process. With the amount of transactions performed daily in the office, the possibility of an error exists. In addition, prompt service is often requested and may not be possible due to the current manual system.

Constraints

The major constraint foreseen for the project concerns available hardware. Currently there are two machines feasible for consideration. These are the DEC 2065 and a TRS 80 previously purchased by the University of Montana Physical Plant. It is possible that a hardware purchase will be made in the future, however, not necessarily in time to host the key inventory system. Included here, is a study of several micro computers on the market today, which could sufficiently contain the key inventory system.
Furthermore, response time to user requests is required within a reasonable amount of time. It must be faster and more efficient than the current manual lookup system. Sufficient storage must be provided to accommodate the amount of data necessary on-line, and off-line storage must be available for audit and security purposes.

Prioritized Subsystems

The following subsystems are components of the key inventory system, as previously listed in the project proposal. They are listed in order of decreasing importance.

1. The database information would include files containing information on what keys an individual is issued, which individual had a particular key, what keys open what doors, what keys belong to a building, what locks (or doors) a master key would open, and how long a key has been out.
2. Deposit accounting would trace the deposits entering the system and the refund of deposits after the return of a key.

3. Interface with current administrative programs. An interface with payroll is possible to allow for tracking of keys when an employee, staff, or instructor, with a key leaves the university. Furthermore, an interface with the registrars office would provide lists of those persons graduating, to cross reference with outstanding keys.

4. Information security, providing for restricted access to the key inventory system. Accessibility of information within different parts of the system would be limited.

5. Correspondence generation, to provide for an automatic generation of form letters requesting the return of keys prior to graduation, or summer break, etc.
6. Key inventory control system, to provide instantaneous information regarding the number of any one key available to be issued, the number of those keys already issued, a threshold number for ordering the duplication of individual keys, and the number of duplications to be made.

7. Bit pattern generation of numbers available to use for any one lock, providing a number for the lock which would be compatible with the master keys opening the lock in questions. This generation of numbers would also take into account any numbers which are on the retire list, and requirements concerning bit cuts.

8. A rekey history would provide a means for tracing the history of a lock which was rekeyed.

9. A parts inventory control, which would maintain a count of all key blanks for each keyway. Yearly, an inventory re-order list would be generated. In addition, account of lock components would also be maintained.
10. Expenditure accounting would provide the department with a trace of expenditures on inventory and other aspects of the system.

Alternative Solutions

The two computers available for use are the DEC 2065 and the TRS 80. The DEC 2065 is operated by the University of Montana Computer Center and is the host machine for many of the campus administrative programs. The computer center provides support services for qualifying programs. The TRS 80 is a micro-computer owned by the Physical Plant. It is currently used by several people in that department, however, investigations are being conducted to determine the feasibility of purchasing another microcomputer.

DEC 2065

The project under study is dependent upon extensive data files to which storage and access is necessary. This aspect of the system indicates
that a database approach is appropriate. While a database may be constructed in any one of the languages available on the DEC 2065, the amount of effort required to produce such a system would be prohibitive. The DEC 2065 does, however, provide a database manager, 1022, and its associated programming language, PL1022. Furthermore, support for this language is provided by the Computer Center and several of the administrative programs are written in PL1022, making it easier for these programs to be interfaced with each other. Therefore, it is recommended that the 1022 system be used for the current project. The following estimates and figures are based on this assumption.

A preliminary estimate has been made to determine the amount of space necessary to store both the proposed program and its files on the DEC 2065. File structures have been drafted to estimate the space requirements, however, this is not necessarily the final structure of the files. A more detailed study of the system will be required...
to determine the precise layout.

There are six files needed to store the necessary information in an efficient manner. These are the Name File, Key Id File, Master File, Master Key File, Change Key File, and Parts Inventory File. The fields of these files and the space required for each file is listed below.

**Name File:**

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Type</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security Number</td>
<td>integer</td>
<td>9</td>
</tr>
<tr>
<td>Name</td>
<td>text</td>
<td>30</td>
</tr>
<tr>
<td>Key Id Number</td>
<td>integer</td>
<td>5</td>
</tr>
<tr>
<td>Date Out</td>
<td>date</td>
<td>8</td>
</tr>
<tr>
<td>Deposit</td>
<td>integer</td>
<td>4</td>
</tr>
</tbody>
</table>

19,872 records X 52 bytes = 1,033,344 bytes

1,033,344 bytes / 2560 bytes/page = 403 pages

**Key Id File:**

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Type</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Identification</td>
<td>text</td>
<td>10</td>
</tr>
<tr>
<td>Key Id Number</td>
<td>integer</td>
<td>5</td>
</tr>
</tbody>
</table>

7500 records X 15 bytes = 112,500 bytes

112,500 bytes / 2560 bytes/page = 44 pages
Master File:  

<table>
<thead>
<tr>
<th>Field type</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Id Number</td>
<td>integer 5</td>
</tr>
<tr>
<td>Type</td>
<td>text 1</td>
</tr>
<tr>
<td>Number Out</td>
<td>integer 3</td>
</tr>
<tr>
<td>Total Number</td>
<td>integer 3</td>
</tr>
<tr>
<td>Status</td>
<td>text 1</td>
</tr>
<tr>
<td>Date</td>
<td>date 8</td>
</tr>
</tbody>
</table>

7500 records X 21 bytes = 157,500 bytes

157,500 bytes / 2560 bytes/page = 62 pages

Master Key File:  

<table>
<thead>
<tr>
<th>Field type</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Id Number</td>
<td>integer 5</td>
</tr>
<tr>
<td>Building</td>
<td>text 3</td>
</tr>
<tr>
<td>Lock/Room</td>
<td>text 4</td>
</tr>
</tbody>
</table>

5000 records X 12 bytes = 60,000 bytes

60,000 bytes / 2560 bytes/page = 25 pages

Change Key File:  

<table>
<thead>
<tr>
<th>Field type</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Id Number</td>
<td>integer 5</td>
</tr>
<tr>
<td>Building</td>
<td>text 3</td>
</tr>
<tr>
<td>Lock/Room</td>
<td>text 4</td>
</tr>
</tbody>
</table>

7000 records X 12 bytes = 84,000 bytes

84,000 bytes / 2560 bytes/page = 35 pages
### Parts Inventory File:

<table>
<thead>
<tr>
<th>Field</th>
<th>Field type</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyway</td>
<td>text</td>
<td>4</td>
</tr>
<tr>
<td>Number on Hand</td>
<td>integer</td>
<td>5</td>
</tr>
<tr>
<td>Reorder Number</td>
<td>integer</td>
<td>4</td>
</tr>
</tbody>
</table>

30 records X 13 bytes = 390 bytes

390 bytes / 2560 bytes/page = 1 page

In 1022, all fields are stored as text fields, even though treated mathematically as numeric fields. Therefore, space estimates may be determined according to the maximum number of characters a given field will occupy. 2,560 characters may be stored on a disk page. In the format previously listed, the files for the proposed project will require 570 disk pages, while it is estimated that the code itself will need 100 pages. Therefore, the total space required to implement the entire project on the DEC 2065 will be 670 disk pages. Approval of this amount of storage is necessary from the Computer Center, before development can proceed.
Access to the DEC 2065 is possible from the terminals already possessed by the Physical Plant. Furthermore, compatible printers are available in the department. Because of the amount of usage the DEC 2065 is exposed to, a delay in logon access time may occur. Once logged on, however, response time is satisfactory.

An advantage to considering this alternative is that the interface with other administrative programs would be of relative ease. Access to files utilized by other administrative departments would increase the accuracy of information used by the proposed system, by providing missing information. In particular, social security numbers would be accessible, thereby improving the possible return rate of keys. Also, support provided by the Computer Center would allow for future maintenance and enhancements, should these be necessary.
The major disadvantage to utilizing the DEC 2065 is that delays may be unavoidable when logging on. Periodic equipment downtime may occur, however, this is a difficulty encountered with any machine. Of course, maintenance on the DEC 2065 is provided by the Computer Center, whereas, departmentally owned hardware is the responsibility of that department.
A second mainframe computer is owned by the University of Montana and maintained by the Computer Center. This is the VAX 11/785. It was recently acquired through student computer fees, and has been installed on the condition that its use will be by students only, and that administrative programs would not be run on it. It is, therefore, not a viable alternative for the current key inventory control system.

As previously mentioned, a 64K TRS 80 microcomputer is located at the Physical Plant. Currently, use of this machine is quite high, however, the possibility exists that a new system will be purchased, thus freeing space on the TRS 80. Database software is available for this system, however, none of these packages are owned by the Physical Plant. The possibility exists for programming a database manager on the TRS 80, but
this is outside of the scope of the current project. Therefore, for this system to be feasible on the TRS 80, the purchase of a database package would be required.

Assuming that access difficulties are resolved, and database software becomes available, consideration of this system as an alternative may be made. A survey of available software reveals that database systems may be purchased with storage capacity from 700 records up to 65,536. These range in price from $200 to $700 and require two to four floppy disk drives, or an external Winchester hard disk drive. There are four floppy disk drives installed on the available TRS 80.

File structure on the TRS 80 would be different from that shown above for the DEC 2065. In this particular configuration, only three files would be necessary, the largest of these containing 19,872 records. The record width would be up to 102 characters wide. Available software will
handle records up to a width of 255 characters. The following shows the breakdown of these three files.

<table>
<thead>
<tr>
<th>Name File:</th>
<th>Field Type</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security #</td>
<td>integer</td>
<td>9</td>
</tr>
<tr>
<td>Name</td>
<td>text</td>
<td>30</td>
</tr>
<tr>
<td>Key Number</td>
<td>integer</td>
<td>5</td>
</tr>
<tr>
<td>Date Out</td>
<td>date</td>
<td>8</td>
</tr>
<tr>
<td>Deposit</td>
<td>integer</td>
<td>4</td>
</tr>
<tr>
<td>Building</td>
<td>text</td>
<td>9</td>
</tr>
<tr>
<td>Room/Lock</td>
<td>text</td>
<td>40</td>
</tr>
<tr>
<td>Key Type</td>
<td>text</td>
<td>1</td>
</tr>
</tbody>
</table>

19,872 records x 102 bytes = 2,026,944 bytes

2,026,944 / 184K bytes/disk = 11 floppy disks

<table>
<thead>
<tr>
<th>Master File:</th>
<th>Field Type</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Id Number</td>
<td>integer</td>
<td>5</td>
</tr>
<tr>
<td>Type</td>
<td>text</td>
<td>1</td>
</tr>
<tr>
<td># Out</td>
<td>integer</td>
<td>3</td>
</tr>
<tr>
<td>Total #</td>
<td>integer</td>
<td>3</td>
</tr>
<tr>
<td>Status</td>
<td>text</td>
<td>1</td>
</tr>
<tr>
<td>Date</td>
<td>date</td>
<td>8</td>
</tr>
</tbody>
</table>

7500 records x 21 bytes = 157,500 bytes

157,500 / 184K bytes/disk = 1 floppy disk
From these figures, it can be seen that, although possible, swapping eleven disks to access the information in one file, may get rather tedious. The purchase of a Winchester Hard Disk would alleviate the problem, however, the cost could approach approximately $2,000.

The major advantages of using the TRS 80 are that security of the key inventory system would be enhanced by limited access. Access would be limited only to in-house users, as opposed to use by the entire university community of the DEC 2065.

The major disadvantages are that at present, access is limited because of the heavy use of the system. Furthermore, the prohibitive size of the
files creates a response time difficulty that could not be alleviated without major hardware purchase, even assuming the purchase of database software. Also, there would not be the access of additional information, through other administrative programs.

Combination

The possibility exists of dividing the proposed project between the two previously mentioned computers. The portion of the system that must interface with administrative programs is that of the key issue control, which could be implemented on the DEC 2065, while, the locksmith portion could be developed for the TRS 80. Five files would remain on the DEC 2065, these being the Name File, Key Id File, Master File, Master Key File, and Change Key File. A reduction in the size of the Master File would be realized because two fields would be eliminated. This would result in an overall page reduction from 670 pages to 642.
Two files would be required on the TRS 80, the Master File and the Parts Inventory File. Because it is no longer an integral system, the programs would not interface, therefore duplication of files would be necessary. One disk would be needed for both of the files on the TRS 80, necessitating only two floppy disk as opposed to the original 13 disks.

Assuming that the previously mentioned access problems are resolved, the locksmith may have faster access to his portion of the program on the TRS 80, as opposed to the DEC 2065. However, this also assumes that database software purchases will be approved in a timely manner.

The major disadvantage to this approach is that the key inventory system will be divided into two separate subsystems, necessitating both a duplication of files and information, and a lack of interface between the two systems. In addition, the assumptions of increased access time and
purchase of database software may not be realized.

HP 150

Hewlett Packard has a micro computer which could be purchased to fill the needs of the Physical Plant and the key inventory system. The HP 150 boasts a 15M hard disk with memory boards available for additional storage capacity. The hard disk would fill the needs of the key inventory system, and provide a back up on the floppy disks. The file structure, should this computer be purchased would be identical to the one suggested for the TRS 80. However, as there are up to 710K bytes of storage on a floppy disk, only 4 floppy disks would be needed to store the back up files. The cost of this system would be approximately $6,000 to $7,000. This would include any additional memory boards, the MSDOS operating system, and DBaseII, for the data base manager.

The major advantages of this system, should it be purchased, would be much the same as the TRS
80 in that the access to the files would be limited to only those individuals needing to use it. Also, as this computer has a much greater memory capacity in its hard disk, floppy disks would only be used for back up. This would alleviate the problem of constantly changing disks. The HP 150 also provides a unique new method of 'program' control in its TOUCHSCREEN option. This allows the novice user to simply point (touch) the screen where the change or option is wanted. This may make it easier for the new and/or novice user to manipulate the programs without having to deal with the sometimes threatening idea of a keyboard.

The major disadvantage of using the HP 150 is that by putting the key inventory system on a micro computer, all interfaces with other administrative programs would be extremely difficult, if not impossible. The additional information gained from these other programs would greatly enhance the Physical Plant's recovery rate of outstanding keys. Furthermore, the system would have
to be purchased, which can be a lengthy process in a state institution such as the University of Montana.

IBM XT

The IBM XT provides an alternative that could also be purchased within the same price range as the HP 150. The basic system with 10 Megabytes of memory on a hard disk would cost approximately $4500. Memory can be added in increments of 256k to 640K. This machine would require a one week order delay. The floppy disk, used by the IBM XT stores from 300K to 350K. Therefore, use of this system would require twice as many floppy disks to store the back up files as the HP 150 did. The purchase of this system, with the data base software, would run $6000 to $7000.

As mentioned above, the major advantage of the micro computer owned and operated by the Physical Plant is that access to the key inventory files could be limited to only those individuals
needing access. The ability to log on to the computer would not be a function of the number of individuals attempting to use the system. Therefore, access to programs could be more timely.

Disadvantages, also, would be seen, in the inability to interface with on-campus programs. Also, as the system would not be on a University Computer, maintenance of the programs and any problems which might arise would need to be resolved by the Physical Plant.
Apple IIe

The Apple IIe microcomputer is also available as an alternative hardware system for the key inventory control program. Its purchase price including database software (either DBaseII or DBMaster) and the DOS 3.3 operating system plus a dual disk drive and hard disk drive is approximately $3500 to $4000. The hard disk drive has 10 Megabytes of storage and each of its floppy disks may hold up to 150K bytes of information.

The advantages of using a microcomputer for the key inventory control system all apply to this alternative as they did to the HP 150 and IBM XT, however, because of the limited storage on the floppy disks for the Apple IIe, back up storage would be more of a problem. It would take twice as many disks to store the back up information on the Apple IIe as the IBM XT, and five times as many as the HP 150. Furthermore, the maintenance and lack of interface disadvantages also apply to this
Recommendations and Justification

Based on the foregoing discussion, it is the recommendation of this study that the first alternative discussed, that of implementing the entire system on the DEC 2065 is the most feasible solution. Although some of the assumptions made concerning the TRS 80 may be realized at a future date, current conditions are such that a decision should not be based on these assumptions. The adoption of this alternative, of course, is contingent upon approval by the Computer Center.

Each of the microcomputers available on the market but not currently owned by the Physical Plant are viable alternatives to implementing the entire key inventory control system. However, even at $3500, the systems purchase price may be a difficulty, while the DEC 2065 is available for administrative use free of charge. Also, utilization of any of these alternatives would create
difficulties in interfacing with other University programs and would not be maintainable by the Computer Center.

While any of the microcomputers discussed would be a possible candidate for implementing a portion of the system on them, and the other portion on the DEC 2065, the interface problems still exist, and the disadvantage of needing to purchase the system is also a consideration. Furthermore, the disk space savings realized by moving a portion of the program to another computer is not enough to justify losing the interface between program portions, duplication of file information, and loss of guaranteed program maintenance.

Proposed Schedule of Development

Following is a projected schedule for the completion of the proposed project. These dates are approximate, due to the inexact art of estimation.
4/22/85  -  System Feasibility
4/24/85  -  Validation
5/6/85   -  Software Plans and Requirements
5/8/85   -  Validation
6/3/85   -  Product Design
6/5/85   -  Verification
6/26/85  -  Detailed Design
6/28/85  -  Verification
7/10/85  -  Coding and Unit Testing
7/31/85  -  Integration Testing
8/9/85   -  Final Implementation and User Manual
Appendix C: Requirements Document

REQUIREMENTS AND SPECIFICATION DOCUMENT

for CS 599

by

Robin Fauntleroy

Michele Miley

May 5, 1985
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>74</td>
</tr>
<tr>
<td>Requirements and Specifications</td>
<td>80</td>
</tr>
<tr>
<td>Data Flow Diagrams</td>
<td>80</td>
</tr>
<tr>
<td>Data Structure Representation</td>
<td>101</td>
</tr>
<tr>
<td>I. Current Front Office Files</td>
<td>101</td>
</tr>
<tr>
<td>II. Current Locksmith Files</td>
<td>103</td>
</tr>
<tr>
<td>III. Proposed Front Office Data Structures</td>
<td>106</td>
</tr>
<tr>
<td>IV. Proposed Locksmith Data Structures</td>
<td>108</td>
</tr>
<tr>
<td>Data Dictionary</td>
<td>108</td>
</tr>
<tr>
<td>System Interface Description</td>
<td>117</td>
</tr>
<tr>
<td>I. Current Front Office System Interfaces</td>
<td>117</td>
</tr>
<tr>
<td>II. Current Locksmith System Interfaces</td>
<td>118</td>
</tr>
<tr>
<td>III. Proposed Front Office System Interfaces</td>
<td>119</td>
</tr>
<tr>
<td>IV. Proposed Locksmith System Interfaces</td>
<td>120</td>
</tr>
<tr>
<td>Functional Description</td>
<td>122</td>
</tr>
<tr>
<td>Functions and Internal Interfaces</td>
<td>123</td>
</tr>
<tr>
<td>I. Current Front Office Functions and Internal Interfaces</td>
<td>123</td>
</tr>
<tr>
<td>II. Current Locksmith Functions and Internal Interfaces</td>
<td>126</td>
</tr>
<tr>
<td>III. Proposed Front Office Functions and Internal Interfaces</td>
<td>130</td>
</tr>
<tr>
<td>IV. Proposed Locksmith Functions and Internal Interfaces</td>
<td>138</td>
</tr>
</tbody>
</table>
Introduction

The software requirements analysis attempts to satisfy the following objectives: (1). provide a foundation for software development by uncovering the flow and structure of information, (2). describe the software by identifying interface details, providing an in-depth description of functions; determining design constraints and defining software validation requirements, and (3) establish and maintain communication with the user and the requester so that the above two objectives may be satisfied. In the context of this document, then, the requirements and specifications of the key inventory control system for the Physical Plant at the University of Montana will be delineated.

There are two major divisions within the current key inventory control system. The first is the actual locksmith operation, in which not only are keys made, but records are kept of which keys open which locks. The locksmith must maintain records of key numbers currently in use, those available for use, those retired, and the
length of retirement. Furthermore, an inventory must be maintained of key blanks and other lock components. Presently, the locksmith is notified as particular keys are disbursed and supplies are depleted, thereby necessitating their duplication.

The second major portion of the key inventory system is key issue control. Records must be kept of all persons possessing a given key, the deposits received, keys issued to an individual, keys returned by an individual and deposit refunds. Although the procedure for this is relatively straightforward, the amount of information that must be processed is quite extensive. Accuracy is essential for university security. The only account of key distribution is maintained by this office.

Recently, an unofficial audit was performed by State of Montana auditors on the key inventory control system, which suggested improvements might be in order. Because of the large number of keys
involved, it has become extremely difficult and tedious to perform manual searches of records for desired information. Furthermore, much valuable employee time is consumed in this process. With the amount of transactions performed daily in the office, the possibility of an error exists. In addition, prompt service is often requested and may not be possible due to the current manual system.

The following subsystems are components of the key inventory system, listed in order of decreasing importance.

1. The database information would include files containing information on what keys an individual is issued, which individual had a particular key, what keys open what doors, what keys belong to a building, what locks (or doors) a master key would open, and how long a key has been out.
2. Deposit accounting would trace the deposits entering the system and the refund of deposits after the return of a key.

3. Interface with current administrative programs. An interface with payroll is possible to allow for tracking of keys when an employee, staff, or instructor, with a key leaves the university. Furthermore, an interface with the registrars office would provide lists of those persons graduating, to cross reference with outstanding keys.

4. Information security, providing for restricted access to the key inventory system. Accessibility of information within different parts of the system would be limited.

5. Correspondence generation, to provide for an automatic generation of form letters requesting the return of keys prior to graduation, or summer break, etc.
6. Key inventory control system, to provide instantaneous information regarding the number of any one key available to be issued, the number of those keys already issued, a threshold number for ordering the duplication of individual keys, and the number of duplications to be made.

7. Bit pattern generation of numbers available to use for any one lock, providing a number for the lock which would be compatible with the master keys opening the lock in questions. This generation of numbers would also take into account any numbers which are on the retire list, and requirements concerning bit cuts.

8. A rekey history would provide a means for tracing the history of a lock which was rekeyed.

9. A parts inventory control, which would maintain a count of all key blanks for each key-way. Yearly, an inventory re-order list would be generated. In addition, account of lock components would also be maintained.
10. Expenditure accounting would provide the department with a trace of expenditures on inventory and other aspects of the system.
Requirements and Specifications

Requirements analysis is the last step in the initial planning phase of the software life-cycle. The analyst must evaluate the flow and structure of information, refine all software functions in detail, establish system interface characteristics, and uncover design constraints. This section introduces partitionings and definitions of current procedures and the partitioning of the proposed system via the use of graphic aids, data dictionaries, and functional descriptions of processes.

Data Flow Diagrams

The following pages comprise a graphic representation of the key inventory system as it currently operates. Circles indicate a process being performed, while arrows indicate the information flowing between the processes. Processes may be broken down into subprocesses, and will be labeled with a number to refer to the original
process. For example, subprocesses of process number two will be labeled 2.1, 2.2, and so on. The first section shows the diagrams for the current operation of both the front office and locksmith procedures, while the second section depicts those for the proposed operation of both systems.
FIGURE 1
LEVEL 0
CONTEXT DIAGRAM
(CURRENT)
FIGURE 2
LEVEL 1
CONTEXT DIAGRAM
(CURRENT)
FIGURE 3
LEVEL 1 FRONT OFFICE
CONTEXT DIAGRAM (CURRENT)
FIGURE 4
LEVEL 2 FRONT OFFICE
PROCEDURES (CURRENT)

KEY REQUEST
KEY
RECEIPT

KEY

DEPOSIT
REFUND

BANK DEPOSIT

RECORD
CONTROL
1.1

KEY RECORD
KEY CARD

INFORMATION
REQUEST

INFORMATION

ORDER
KEY
1.2

DISTRIBUTE
INFORMATION
1.3
FIGURE 5
LEVEL 3 FRONT OFFICE
RECORD CONTROL (CURRENT)
FIGURE 6
LEVEL 1 LOCKSMITH
CONTEXT DIAGRAM
(CURRENT)
FIGURE 7
LEVEL 2 LOCKSMITH
PROCEDURES (CURRENT)

KEY FILE

PART FILE

MAKE KEY 2.1

REKEY LOCK 2.3

SERVICE LOCKS 2.2

REKEY REQUEST

RECYCLE KEYS 2.5

CONTROL INVENTORY 2.4

FUND REQUEST

FUND APPROVAL

PART ORDER

PART

PATTERN KEY FILE

AVAILABLE KEY FILE

RETIRED KEY FILE

KEY ORDER

KEY ORDER

KEY ORDER

KEY ORDER
FIGURE 8
LEVEL 3 REKEY LOCK
(CURRENT)
FIGURE 9
LEVEL 3 CONTROL INVENTORY
(CURRENT)
FIGURE 12
LEVEL 3: RETURN KEY
(PROPOSED)

- CHECK ALL NAMES 1.4.1
- DELETE FILES 1.4.2
- INCREASE KEY AVAILABLE 1.4.3
- IND.RECORD
- NAME
- REFUND INFO
- ID
- SSN
- NAME
- INVENTORY FILE
- KEY RETURN
- KEY
- AVAILABLE
- KEY
- AVAILABLE
- KEY
LEVEL 3: BOOKKEEPING
(PROPOSED)

FIGURE 13

REFUND INFO

REFUND CHECK 1.5.3

AMOUNT

DATE, AMOUNT

REFUND

UPDATE CHECKBOOK 1.5.4

GENERATE DEPOSIT 1.5.2

CHECKBOOK

BANK DEPOSIT

GENERATE RECEIPT 1.5.1

RECEIPT INFO

RECEIPT

GENERATE DEPOSIT
FIGURE 14
LEVEL 3: GENERATE CORRESPONDENCE (PROPOSED)
FIGURE 15
LEVEL 3: BACKUP INFO
(PROPOSED)
FIGURE 16
LEVEL 1: LOCKSMITH CONTEXT DIAGRAM
(PROPOSED)
FIGURE 17
LEVEL 2: LOCKSMITH PROCEDURES (PROPOSED)
FIGURE 18
LEVEL 3: REKEY LOCK
(PROPOSED)

INVENTORY FILE

GENERATE KEY NUMBER 2.3.1

DETERMINE LOCK NUMBER 2.3.2

REKEY ORDER

PIN LOCK 2.3.3

INSTALL LOCK 2.3.4

PART FILE

KEY NUMBER

REKEY REQUEST

LOCK NUMBER
FIGURE 19

LEVEL 3: CONTROL INVENTORY
(PROPOSED)

PART FILE

STORE PART 2.4.1

CHECK SUPPLY 2.4.2

ESTIMATE NEED 2.4.3

ORDER PART 2.4.4

INVENTORY FILE

KEY/AY FILE
Data Structure Representation

This section describes the format of necessary files and other major data structures within the system. Because of the large amount of data contained within the key inventory control system, the format of stored information is of vital concern, both to maximize the efficiency of accessing the data, and to minimize the amount of disk storage required. The following description attempts to meet both of these goals.

I. Current Front Office Files

Currently, the front office files consist of three separate files. These can be seen in the data flow diagrams as the key file, the key record file and the key card file. These files are described below, with an indication of the current information within each file. For the proposed files, the reader is referred to the section describing proposed front office data structures.
The key file is maintained for both front office and locksmith accessibility. It contains one or more keys for each building and room number available for issue and some master keys which are issued. Approximately 7,000 individual keys are maintained in this file.

The key record file is maintained on each individual who has a key issued to him. Please refer to the data dictionary for a complete description of what is contained in this file. The key record is maintained on the individual until all keys issued to him have been returned. There are approximately 20,000 key records maintained.

The key card file contains a card for each key which is issued to an individual. Each card is maintained in the file until the key has been returned. When a key is returned the key card is attached to the key record file. When all keys belonging to an individual have been returned,
then the key record and the key cards are removed from the file.

The master key record file contains a card for each individual who has been issued a master key. A card may contain information on more than one key issue. These cards are kept, and updated as keys are returned, until all master keys issued to an individual are returned.

II. Current Locksmith Files

Presently, there are seven areas where information is stored in the locksmith procedures. These are shown in the data flow diagrams as the key file, the pattern key file, the retired key file, the available key file, the part file, the generated key number file, and the salvage file. The structure listed below describes the current information residing in those files, with no attempt to optimize storage. For the proposed files, the reader is referred to the section describing proposed locksmith data structures.
The current key file is the same as the one described for the front office. Both front office and locksmith operations need access to this file. For each building and room number, it contains one or more keys, in addition to some master keys available for issue to authorized persons. Approximately 7,000 keys are maintained here.

The pattern key file is a physical file, and contains a copy of each possible key, including masters, grand masters, and great grand masters. These keys are never issued, and are only used as a pattern to make additional keys. Each key is associated with a building (or buildings for a great grand master), and a room number or room numbers. Again, there are approximately 7,000 keys kept in this file.

The retired key file, again a physical file, consists of keys that have been taken out of circulation after a lock has been rekeyed. Each key has a date associated with it to indicate when it
was taken out of service. It is stored here for one year before that particular pattern may be used again. The number of keys stored is variable.

The available key file consists of keys that have been retired for one year or more. These keys may now be used to rekey a lock. As with the retired key file, the number of keys stored here is variable.

The part file, a physical file, contains parts (locks, pinned locks, pins, and key blanks) which may be used for the construction of new locks or new keys. Each part has its own part number. The number of parts is variable, although a standard amount of stock is ordered on an annual basis.

The generated key number file is a repository for numbers available for use in rekeying a lock. The restrictions on these numbers is such that on at least one cut per key there must be a minimum
of two bits difference. This file must contain only those numbers that are sufficiently different from any key made previously, including those in use, those retired, and those already made and available for use. The amount of numbers available for use may vary.

The 'salvage file' is used to store parts that are no longer of use. This is a physical file. When a sufficient quantity is reached, the parts stored here are sold as scrap metal. It may contain any of the parts previously listed and is also used by other departments of the Physical Plant. No differentiation is made between any of the parts, and the amount of parts stored is variable.

III. Proposed Front Office Data Structures

The following files have been determined as being sufficient to store the data required to maintain the front office key issue procedures. There are four files required for the data base
information. Note that the Master Key Record File has been eliminated and the information contained there is stored within the Inventory File as a field indicating the 'type' (master, grand master, change, etc) of the key. These are:

**Ind_Record File:**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security Number</td>
<td>integer</td>
<td>9</td>
</tr>
<tr>
<td>Key ID Number</td>
<td>text</td>
<td>10</td>
</tr>
<tr>
<td>Date Out</td>
<td>date</td>
<td>8</td>
</tr>
<tr>
<td>Deposit</td>
<td>integer</td>
<td>4</td>
</tr>
</tbody>
</table>

**Name File:**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security Number</td>
<td>integer</td>
<td>9</td>
</tr>
<tr>
<td>Name</td>
<td>text</td>
<td>30</td>
</tr>
</tbody>
</table>

**Inventory File:**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key ID</td>
<td>text</td>
<td>10</td>
</tr>
<tr>
<td>Type</td>
<td>text</td>
<td>1</td>
</tr>
<tr>
<td>Quantity on Hand</td>
<td>integer</td>
<td>3</td>
</tr>
<tr>
<td>Total Number</td>
<td>integer</td>
<td>3</td>
</tr>
<tr>
<td>Status</td>
<td>text</td>
<td>1</td>
</tr>
<tr>
<td>Date</td>
<td>date</td>
<td>8</td>
</tr>
</tbody>
</table>
IV. Proposed Locksmith Data Structures

In the proposed locksmith portion of the key inventory control system, most of the files will remain the same with the exception of the parts file, the retired key file, and the available key file. The retired key and available key files will no longer be used, while the used keys formerly stored there will now be stored in the parts file. This is not a computer data file. An additional file will be used, that of the master file as described in the proposed front office data structure representation.

Data Dictionary

Entries in the following pages provide a concrete definition for the flow of data between system processes. Each piece of data describes an
item of information needed for a process to perform its function, or output desired by the users. Curly braces indicate that there may be a repetition of information within a data element, a plus sign indicates 'and', and a vertical bar indicates 'or'. Parentheses depict an item that is optional. Comments about legal data values are enclosed by a slash and an asterisk. Data flows for both the front office and locksmith procedures are defined together, and both the current and proposed system may be found in this section.

\[
\text{amount} = \text{dollar amount of deposit} \mid \\
\text{dollar amount of refund}
\]

\[
\text{bank deposit} = \text{dollar amount} + \\
\text{date} + \\
\text{account number}
\]

\[
\text{date} = \text{date of deposit received}
\]
deposit = amount paid when key is issued

deposit refunds = name +
    date +
    dollar amount

fund approval = fund request +
    signature of approval

fund request = needed part count +
    estimated dollar amount

ID = name |
    social security number

info = information
info request = building + room number | key number | ID

information = keyholder | key number | location | bank deposit | inventory | keys held

information requests = room number | building | {key owners}

key = key blank +
    key number

key number = bit cut number +
    keyway number
key order = key number +
amount needed

key request = ID +
date +
user's position +
building +
room number +
approval signature +
deposit amount

key return = ID +
key number

letter = letter to terminated personnel |
letter to graduating students

list = name +
SSN +
address
/* list of graduating students */
location = building +
    room number

lock number = key number

maintenance request = building number +
    room number +
    description of
    problem

name = /* name of keyholder */
    /* stored in Personnel,
    Registrar, or
    Name File */

needed part count = part +
    part number +
    amount needed
part = lock | pin | pinned lock | key blank

part count = part + part number + amount on hand

part number = lock number | keyway number | pin size

part order = needed part count + fund approval

pinned lock = lock + lock number
query = keyholder report
    | inventory report
    | locksmith report
    | location report

receipt = receipt info

receipt info = name +
    amount +
    date

record info = ID +
    key number +
    date +
    amount

refund = refund info +
    dollar amount of refund

refund info = name +
    amount +
    date
rekey order = key order +
     building +
     room number

rekey request = building name +
     (room number)

request info = info request

requested = /* flag to indicate user
request for
maintenance program
entry */
/* value is true or false */

report = keyholder |
        inventory |
        locksmith |
        location |

restore request = requested
room number = number assigned to given room

SSN = social security number

System Interface Description

This section describes the transactions that the system makes with outside entities, both data received and information that must be output. For a detailed definition of each of the interface data elements described here, the reader is referred to the data dictionary.

I. Current Front Office System Interfaces

The front office key issue system currently interfaces with two outside entities. These are the Physical Plant locksmith operation and a bank. The key orders are sent to the locksmith when the supply of a key is determined to be low, and keys are then received from the locksmith and stored in
the key file. Deposits are made regularly to a local bank which consist of the key deposits collected for each key issue.

II. Current Locksmith System Interfaces

There are presently three outside entities with which the locksmith operation interfaces. These are the Physical Plant front office, parts suppliers, and University of Montana administration. Key orders are received from the front office, and keys made according to those orders are then stored in the front office key file. Parts are ordered from suppliers for the most part on an annual basis, and then received from them. On occasion, parts must be ordered at irregular intervals rather than on the yearly order. Finally, maintenance and rekey requests are received from U of M administration, and either the necessary service is performed, or the appropriate part is distributed to the requester. Fund requests for parts orders are submitted to U
of M administration, and fund approvals are received back.

III. Proposed Front Office System Interfaces

The proposed front office key issue system will be adding several interfaces to those already existing. The new system will be combining the front office procedures with those of the locksmith. Therefore, the existing outside interface with the locksmith will cease to exist as such, and will become an internal interface. However, the interface with the bank will still remain unchanged. Additionally, there will be interfaces with U of M Personnel, and Student Records. These interfaces will provide the Physical Plant with two enhancements to the system that do not exist today. First, these files will be accessed for information which will correlate a social security number with a name. This will allow the data files necessary for the Physical Plant to be stored in a more economical and space
saving manner by not having to store the name. These files will be accessed at the initiation of the Key Inventory Control System to determine which records, as they exist now, can be stored with only the corresponding social security number. These files will then be used to access the proper record during general use of the system, by providing the corresponding social security number of an individual when given the individual's name. Secondly, these files may be used when personnel are leaving University payroll system, or when students are graduating. With this information, reminder letters may be sent to these persons, with the intent of greater return of issued keys.

IV. Proposed Locksmith System Interfaces

   The proposed locksmith system interfaces will remain the same as was described in the current system interfaces. The outside entities receiving and submitting data are the parts supplier, the
Physical Plant front office, and University of Montana Administration. The exchange of information will also remain unchanged.
Functional Description

The functional description portion of this document describes each of the functions delineated in the data flow diagrams both in a narrative style, and then in an algorithmic or structured English version in order to clarify the procedures performed by the system. Data and control interfaces between processes within the system are also described. As with the system interface description, the reader is referred to the data dictionary for a more complete definition of each of the internal data flows, and to the data structure representation for detail on the data base and file structure. Each interface is described in terms of the data flow diagram processes, therefore the reader is referred to those pages while reading this section.
Functions and Internal Interfaces

I. Current Front Office Functions and Internal Interfaces

The current front office portion of the key inventory system utilizes four files. These are the key file, key record, master key record and key card file. The processes within the system can be divided as follows:

1.1 Record Control. This module currently maintains the records of all keys issued to individuals. This process consists of adding a key record or master key record and key card to the respective files, and decreasing quantity on hand from the key file when a key is being issued. When a key is then returned, the opposite procedure is followed. Information on money collected and disbursed is also maintained. This module may be then subdivided as follows:
1.1.1 Issue Key. This subsystem is activated by receipt of a key request. This consists of a card and an appropriate deposit. Currently, a key is retrieved from the key file, the key request form is separated and the appropriate parts are placed in the key record and key card files. If the request is for a master key, a master key card is filled out, or the information is added to an already existing card for that individual. A receipt is then written to reflect the deposit amount. This module interfaces with the Generate Deposit module, by means of the deposit amount.

1.1.2 Generate Deposit. This module is passed the deposit amount. As monies are collected a total is maintained. This total must agree with the checkbook.

1.1.3 Return Deposit. This module is used when a key is being returned. As a key is received in the Physical Plant, the key is
replaced within the key file, and the corresponding key card is retrieved from the key card file. The key record card is also retrieved from the key record file and marked returned. If the individual has no other key issued to him, then the key record is disposed of, however, if other keys are outstanding, the old record is stapled to the current ones and are maintained until all keys have been returned. This module will add to the refund list, the name and total amount of refund, as refund checks are not issued from the Physical Plant office. At the same time, the checkbook is decremented to reflect the refund. This module will access the information in all three files.

1.2 Order Keys. This subsystem interfaces only with the key file. As supplies of keys get low, an order for additional keys is made to the locksmith division of the Physical Plant. Currently, this subsystem is entirely manual, and relies on the user of the system to be aware of how often a key is needed and what additional
amount of keys might be necessary. There are approximately 7,000 keys in the key file.

1.3 Distribute Information. This subsystem interfaces with all three files and attempts to answer questions posed to the Physical Plant with reference to key situations and problems. Currently, to answer a question, the user must access the data in the files manually. This involves accessing the keys file, which contains around 7,000 keys and the key record file,

II. Current Locksmith Functions and Internal Interfaces

The top level of the locksmith operation can be divided into five processes. The first of these is Make Key which receives key orders from the front office and the Service Locks procedure, and rekey orders from the Rekey Lock process. From these orders, new keys are made, using a pattern key from the pattern key file as a model. Keys made are either returned to the front office key
file or the Service Locks process. In addition, periodically the Make Key function will make a new pattern key as old ones wear out or as locks are rekeyed, to be stored in the pattern key file.

Service Locks receives maintenance requests from the University of Montana Administration. The type of service necessary is determined, and the service performed. From time to time, it may be necessary to make a new key, at which time the Make Key process is sent a key order. Information concerning key numbers or lock numbers is obtained from the pattern key file.

Rekey Lock receives rekey requests from the University of Montana Administration. It accesses the available key file to determine if there is a valid key number already available, and places old keys in the retired key file. It receives necessary parts from the part file. Rekey Lock may be broken into four subprocesses. These are Generate Key Number, Determine Lock Number, Pin Lock, and
Install Lock.

Generate Key Number accesses the pattern key file, the available key file, and the retired key file to determine bit cut numbers that have not yet been used, and stores these in the generated key number file.

Determine Lock Number first checks the available key file for bit cuts fitting the key requirements as rekey requests are received. If no bit cuts are appropriate, then it accesses the generated key number file for an appropriate bit cut. Once a key number has been determined, the lock number is sent to the Pin Lock process while a rekey order is sent to the Make Key function.

The Pin Lock procedure takes locks and pins from the part file and installs the pins in the lock according to the lock number. The pinned lock is then passed along to the Install Lock procedure.
Install Lock, as the name implies, installs locks in doors. Old locks may be returned to the part file if still of use, otherwise, they are placed in the salvage file. Keys corresponding to the old lock are either put in the retired key file or the salvage file.

The Control Inventory function receives parts from suppliers, sends parts orders to suppliers, sends fund requests to the University of Montana Administration, and receives fund approvals. It accesses the part file to determine the amount of parts needed and to store received parts. It may be divided into four subprocedures, Store Part, Check Supply, Estimate Need, and Order Part.

Store Part receives parts from the supplier and places them in the part file. Check Supply checks the part file and counts the amount of parts on hand. The part count is then sent to the Estimate Need function.
Estimate Need determines the past need of a given part and future need in relation to possible projects, and estimates the amount to be ordered. This needed part count is sent to the Order Part process. An estimate is also made of the dollar amount required to purchase the parts, and this is sent out as a fund request the University of Montana Administration.

The Order Part procedure waits until fund approvals are received from the University of Montana Administration, and once received, sends a part order to the supplier.

Recycle Keys accesses the retired key file to determine which keys have been out of circulation for a year or more. These keys are then transferred to the available key file.

III. Proposed Front Office Functions and Internal Interfaces
1.1 Distribute Information. This module interfaces with the existing data base files. It will allow the user to query the files for specific information to respond to questions posed to the Physical Plant. This will provide access, quickly, to such questions as 'How many keys does an individual have issued to him?', or 'What key opens a particular room in a particular building?'. This module will also interface with the Find SSN module by way of the 'ID'.

1.2 The Find SSN module is used to find the correct file to access for key records. This module is passed a name, and will check both the University Personnel records and the University Student records for that name and the corresponding social security number. If the name is found, then the social security number can be used to access the Ind Record file, otherwise the original name is used to access the Name file. The Find SSN module receives the name to search for from either Distribute Information, Issue Keys, or
1.3 Issue Key. This module maintains the Ind Record file and the Name file to reflect the current state of keys issued. As a new key is issued to an individual, a new record is placed in either the Ind Record file or the Name file to indicate this transaction. Current figures must also be maintained to reflect the additional deposit acquired. This record keeping is done within the bookkeeping module. The Issue Key module interfaces with the Find SSN module through the individuals name, if available, and to the bookkeeping module by way of receipt information. This module can be broken down as follows:

1.3.1 Find Key #. This module is passed the key request information. It uses the room and building number to find the appropriate key number from the Location File. The only other module that Find Key # interfaces with is the Update Name File, which is passed the record information.
1.3.2 Update Name will interface with the Find SSN module to determine which file to access. It will add a record for the current transaction, reflecting the issue of a key.

1.3.3 Check Key Quantity. This module checks the master file to be sure that there is a key available to be issued to the requester.

1.3.4 Decrease Key Quantity Available. This module will decrement the record field that indicates how many keys are currently available (Quantity on Hand). This provides a measure of the number of keys available for issue and allows for a means of re-ordering keys (through the Order Key module).

1.4 Return Key. This module is passed the key return information (ID and key number). It passes to Find SSN the individuals ID, if needed to find the name to access the proper file. It will change the key status in the proper file to reflect the returned key. It will also search the
file for all occurrences of that individual to determine if he has returned all of the keys he has been issued. If so, all records will be removed from the files. This module also access the Inventory File, and increments the number of keys available, for the particular key number, that has been returned. It can be divided into the following functions:

1.4.1 Check All Names. This module will peruse the appropriate file for all records belonging to that particular user. It will then check to see if all the keys issued to that individual have been returned.

1.4.2 Delete Files will delete all records pertaining to that individual.

1.4.3 Increase Key Available will access the Inventory File and increase by one the amount of keys available for that particular key number.
1.5 Bookkeeping. This module does the general bookkeeping tasks for the key inventory system. It is passed the name, date and amount from either of two modules, the Issue Key module or the Return Key module. It will generate either a receipt or a refund check. It also maintains a 'checkbook', which contains a current total of monies available. This module can be divided into the following subsystems:

1.5.1 Generate receipt is passed the name, date and amount from issue key. It then generates a receipt for that amount.

1.5.2 Generate deposit is passed the date and amount and will increment the deposit total. It will generate a daily total.

1.5.3 Refund check is passed the name, date and total amount of refund. It adds this information to the refund list, as checks are not issued directly from the Physical Plant.
1.5.4 Update checkbook is passed an amount from either refund check, in which case it decreases the checkbook amount, or from generate deposit, where it will increment the checkbook by the 'amount'.

1.6 Generate Correspondence. This module will provide the Physical Plant with a means of generating correspondence. Input to this module will be of two possible types. One is a query to the data base files, which is possible in the 1022 data base language. The other type of input would be a list, possible generated by the University Administration, or Registrars office. For example, a list of graduating seniors could be compared with those individuals having been issued keys and letters written reminding the students to return keys prior to leaving the University. By use of a query, specific information may be extracted from the files. Informal reports may then be generated to relay this information to the user. This module may be broken down into the
following subsystems:

1.6.1 Generate Letters. This module will access the Ind Record file only as all names coming from the University system will have a social security number. The module will interface with the Find SSN module to determine the proper name to access.

1.6.2 Generate Reports. This module may access many files and modules depending on the information requested. It has access to the Ind_Record file, the Name file, the Location file and the Inventory file.

1.7 Order Keys. This module interfaces with the Inventory file. It will check the threshold number and the number of key available for issue and order new keys, if necessary.

1.8 Backup Info. This module provides a means of creating backup files for security of information and audit purposes. This module will
access the University tape drives, and simply store the files requested on the tape. The Restore request will move the files off the tape, restoring them to disk, but providing another name so as not to overwrite the current files.

1.9 Find Name. This module will find the name associated with the given SSN in the Personnel, Student, and Name files, and return this to the calling procedures.

IV. Proposed Locksmith Functions and Internal Interfaces

Many of the locksmith procedures are not amenable to computerization, therefore most of the processes remain unchanged from the current system to the proposed system. However, there are some changes as described below.

The available key file and retired key file no longer exist as such in the proposed system, while necessary information about these keys is
stored in the on-line master file. All previous mention of accesses to these files, then, are replaced with accesses to the master file. In addition, the Recycle Keys function is not necessary, but is replaced by the process Update Key Status. Update Key Status accesses the master file and checks the retired status and date of each key record. If the key has been retired for a year or more, its status is changed to available.

Furthermore, an on-line inventory file is kept of all keyway numbers. Each time a key blank is used, then, the on-line count must be decreased, and as parts are received from suppliers, the count must be increased by the amount received.

Processing Narratives

This section provides an unambiguous description of processes within the system in the form of a structured English or algorithmic method of specification. This allows for more clarity in the
understanding of how the system works, and how the proposed system will carry out the same tasks. These processing narratives correspond to those processes found at the lower levels of the entire system.

I. Current Front Office Processing Narratives

1.1.1 Issue Key

with key request do
find key in key file
give to requester
separate key request card
place key record in key record file
place key card in key card file
if key is a master key
then check master key file for previous issues
if card exists
then add current key information to card
else create new card with current information
write receipt for amount
give receipt to requester
1.1.2 Generate Deposit

if date has not changed
then add deposit amount to bank deposit total
adjust checkbook total to reflect new amount
else print bank deposit
set bank deposit total to zero
add deposit amount to bank deposit total
adjust checkbook total to reflect new amount

1.1.3 Return Deposit

with a key return do
place key in key file
remove corresponding key card and dispose
find corresponding key record card and mark returned
find all key records for that individual
if all are marked returned then dispose of all of them
else staple returned key record to active key record
if key was a master key then find appropriate card in master key card file mark key returned
if all master keys have been returned then dispose of card
1.2 Order Key

check each key in key file
if key supply appears to be low
order more

1.3 Distribute Information

for each information request
determine appropriate file
if information can be accessed
then respond to request
II. Current Locksmith Processing Narratives
2.1 Make Key

as key order is received,
for each entry on key order,
match key number with
key in pattern file
select key blank from
parts file with same
keyway number as
pattern key
for each amount needed,
cut key blank
according to pattern
key bit cut
return pattern key to
pattern key file.

as rekey order is received,
for each entry on rekey order,
select key blank from
parts file with keyway
number indicated on
rekey order,
cut key blank according to
bit cut number
indicated on rekey order,
paint key red,
put key in pattern key file,
put old pattern key in
retired file and
mark date.

as needed,
check pattern key file for worn keys,
for each worn key found,
select key blank from parts
file with same
keyway number as worn key,
cut key blank according to
worn key bit cut,
paint key red,
put key in pattern key file,
put old key in salvage file.
2.2 Service Locks

as maintenance requests are received,
determine type of service necessary.  
if new key is required,
    send key order to make
    key process,
    receive key,
    distribute key to requester.
else, if service necessary,
    perform required service.

2.5 Recycle Keys

once per month,
    check retired key file for dates
    older than one year,
    move these keys to the available
    key file.

2.3.1 Generate Key Number

check all existing key numbers in
    pattern key file,
    available key file,
    and retired key file.
    determine key numbers not in use.
    of these, select numbers fitting
    bit cut number
    requirements and put in
    generated key number
    file.
2.3.2 Determine Lock Number

as rekey request is received, check available key file for key number fitting master, grand master, and great grand master requirements. if found, select first one as lock number, else, check generated key number file for key number fitting master, grand master, and great grand master requirements. select first one as lock number.

2.3.3 Pin Lock

for each number in lock number, determine size of pin needed, get pin from part file, place pin in lock in appropriate position.
2.3.4 Install Lock

as pinned lock is received,
install lock in door,
if parts are still usable,
place corresponding keys 
in retired file,
place other parts in part 
file,
else, place all parts in 
salvage file.

2.4.1 Store Part

as parts are received,
separate according to type  
and part number,
store in part file.

2.4.2 Check Supply

periodically,
for each part type,
for each part number,
count amount of 
parts on hand.
2.4.3 Estimate Need

determine projects to be completed.
determine parts needed for projects.
subtract part count from parts
    needed to get needed
    part count.
estimate dollar amount of needed
    parts.
    fill out fund request.

2.4.4 Order Part

as fund approval is received,
    mail part order to supplier.
III. Proposed Front Office Processing Narratives

1.1 Distribute Information

if info request refers to name of person, amount of keys, etc.
then check University Files for social security number
if found then find answer to question in Ind_Record file
respond to question
else look in Name file for appropriate social security number
map back to Ind_Record file
respond to question
else if info request refers to keys, key numbers, etc.
then find answer to question in master file
respond to question
else no response to question

1.2 Find SS#

check Personnel files for name
if found then return social security number
else check student files for name
if found then return social security number
else return not found
1.3.1 Find Key Number

while not found and not end of
  master key file
  compare building and room numbers
  if equal then found equals true
  return key number
else return invalid specifications

1.3.2 Check Key Quantity

while not found and not end of
  master file
  compare current number with key number
  if equal then found equals true
  if key quantity is greater than zero
    return available
  else return not available

1.3.3 Decrease Key Quantity Available

with current key number
  decrease key quantity available by one
1.3.4 Update Name File

If accessing Ind_Record file then create new Record
  add social security number
  add key number
  add date
  add amount of deposit
  insert in Ind_Record file
else
  create new record
  add name
  if available add social security number
  add key number
  add date
  add amount of deposit
  insert in Name file

1.4.2 Delete Files

  with the appropriate record
  delete from file

1.4.3 Increase Key Available

  with the appropriate master file record
  increase the keys available by one
1.4.1 Check All Names

if ID is a name
then while there are more entries in the Name file
  check to see if record matches name
  if matches, map back to Ind_Record file
    check for key returned
    set flag
else if ID is a social security number
  then while there are more entries in the Ind_Record file
    check to see if record matches name
    if matches, check for key returned
    set flag

1.5.1 Generate Receipt

print name
print date
print total amount of deposit

1.5.2 Generate Deposit

if date is changed
  then print out yesterday's total
  set total to 0
  increment total by deposit amount
else increment total by deposit amount
1.5.3 Refund Check

add name, date, and amount of refund to refund list

1.5.4 Update Checkbook

if amount is from generate deposit then add amount to checkbook amount store new amount in checkbook
else subtract amount from checkbook amount store new amount in checkbook

1.6.1 Generate Letters

while there are names or social security numbers on the list find appropriate records print reminder letter

1.6.2 Generate Reports

find information pertinent to the query print the information
1.7 Order keys

while there are more keys to check
    compare the number of keys available with
    the threshold number
    if the number is less than threshold number
    then order key made

1.8.1 Save Info

while there are more files to save
    save the file on tape

1.8.2 Restore Info

while there are more files to restore
    change name to 'file'.bak
    restore the file to disk

1.9 Find Name

check Personnel Files for SSN
    if found then return name,
    else check student files for SSN
        if found then return name,
        else return not found
IV. Proposed Locksmith Processing Narratives
2.1 Make Key

as key order is received,
for each entry on key order,
match key number with key
in pattern file
select key blank from parts
file with same
keyway number as pattern
key
subtract key blank from
keyway file.
for each amount needed,
cut key blank according
to pattern key
bit cut
return pattern key to pattern
key file
add key to master file.
as rekey order is received,
for each entry on rekey order,
select key blank from parts
file with keyway
number indicated on rekey
order,
subtract key blank from keyway
file
cut key blank according to bit
cut number
indicated on rekey order,
paint key red,
put key in pattern key file,
put old pattern key in retired
file and mark date,
add key to master file.
as needed,
check pattern key file for worn keys,
for each worn key found,
select key blank from parts file
with same
keyway number as worn key,
subtract key blank from keyway
file
cut key blank according to worn
key bit cut,
paint key red,
put key in pattern key file,
put old key in salvage file.

2.2 Service Locks

as maintenance requests are received,
determine type of service necessary.
if new key is required,
send key order to make key
process,
receive key,
distribute key to requester.
else, if service necessary,
perform required service.

2.5 Update Key Status

once per month,
check master file for retired key
dates older
than one year,
change status on these to available.
2.3.1 Generate Key Number

check all existing key numbers in master file,
determine key numbers not in use.
of these, select numbers fitting bit cut number requirements and put in generated key number file.

2.3.2 Determine Lock Number

as rekey request is received,
check available key file for key number fitting master, grand master, and great grand master requirements.
if found, select first one as lock number,
else, check generated key number file for key number fitting master, grand master, and great grand master requirements.
select first one as lock number.
2.3.3 Pin Lock

for each number in lock number,
  determine size of pin needed,
  get pin from part file,
  place pin in lock in appropriate
  position.

2.3.4 Install Lock

as pinned lock is received,
  install lock in door,
  if parts are still usable,
    add corresponding keys to
    master file,
    place parts in part file,
  else, place all parts in salvage
  file.

2.4.1 Store Part

as parts are received,
  separate according to type and
  part number,
  store in part file.
if key blanks are received,
  add amount to keyway file.
2.4.2 Check Supply

periodically,
  check keyway file for amount of
  key blanks in stock.
  for each other part type,
  for each other part number,
  count amount of parts on
  hand.

2.4.3 Estimate Need

determine projects to be completed.
determine parts needed for projects.
determine past amount used from
  master file,
subtract part count from parts needed
  to get needed
  part count.
estimate dollar amount of needed parts.
fill out fund request.

2.4.4 Order Part

as fund approval is received,
mail part order to supplier.
Resources

Several different types of resources are necessary for the development of a software product. These include human, hardware and software resources. The subject of this section is to discuss the resources available to the completion of the proposed system, and those necessary for successful completion. In addition, availability windows are given to describe the time that each resource is needed and when each will be available for use.

Human Resources

During the course of development, certain tasks must be completed, requiring sufficient manpower at these points. A two member development team is available for the duration of the project to complete both the documentation and implementation of the system. In addition, a four member committee is necessary for the review and approval of results at given milestones in the course of
system development. The committee is composed of the chairman, a professor of computer science whose special expertise lies in the area of data processing application programming, another computer science professor who has specific knowledge concerning the development of sufficient specifications and requirements for the system, an employee of the Computer Center, who brings expertise on University of Montana administrative programs, and policy regarding their development, and finally, the requester of the system, who has intimate knowledge of the system for which the proposed project is being developed. Each of these members must be available to review the progress of the project at proposed dates of delivery as discussed in the feasibility document. It is anticipated that there will be times when all committee members will not be able to be present at certain progress meetings, therefore individual review of documentation may be necessary in addition to the traditional group review of progress.
However, each member must be present for the final review, or defense, of the system scheduled for the first week in August.

In addition to periodic review of the proposed system by committee members, end users must also be available to clarify current system operation and to verify the desired nature of the eventual software. In particular, it is necessary to have communication with the two primary end users of the system as questions concerning procedures arise. Finally, before the system is operationally complete, the files of the current system must be entered into the automated system, requiring the time of one or more persons to perform data entry tasks. One of the end users has discussed the possibility that she may be able to do this task or that temporary help may be hired. Because of the lightened work load during the summer quarter at the University of Montana Physical Plant, it has been decided that this would be the most feasible time to enter the data if space is available on
the target hardware system. Training of users on the proposed system once it is complete will be performed by the two member development team.

Hardware Resources

Several hardware alternatives were discussed in the feasibility document produced earlier, and it was the final recommendation to develop the system for the DEC 2065, a mainframe computer used for University of Montana administrative programs, because of program interface possibilities and ease of future maintenance. Based on this recommendation, this section will discuss those hardware resources available for use with the DEC 2065, and those necessary.

As previously discussed, the DEC 2065 is available for University of Montana administrative programs for which disk space has been allotted. It is currently being decided both whether and when disk space may be allocated for the key issue control data base and associated programs. The
availability of disk space in time for full implementation of the program by August 1985 is as yet, unknown. Other necessary hardware includes a tape drive to create back up storage of data base information, which is currently available on the DEC 2065, and access to the program via terminals and printers. The Physical Plant currently owns several compatible terminals, and at least one printer to be used for this interface.

Software Resources

One of the main advantages of using the DEC 2065, as discussed in the feasibility study, is the opportunity for the key inventory control system to interface with existing administrative programs and data base information to provide for better control and tracking of key holders. In particular, access to both personnel and registration files may provide an earlier indication of key holders leaving the University system so that key returns may be requested. In addition,
presently there are no records of key holder social security numbers, and these may be obtained via information already stored in administrative data bases. Addresses of key holders are also not maintained by the current system, but may be obtained by interfacing with available information on the DEC 2065, in order to initiate a letter sending function of the system which is anticipated to increase the return rate of keys. Therefore, the existence of these data bases is one of the software resources available to the proposed project.

Another advantage of the DEC 2065 is the presence of a sophisticated data base system, 1022. This software allows for both storage of large amounts of data, and for query concerning this data. It also provides a programming language to access the data base. The proposed system must interface with the 1022 software. A less noticeable, but highly important software resource is the TOPS 20 operating system implemented on the
DEC 2065, which provides a user friendly environment for future use of the proposed project, and a rich setting for program development.

Availability Windows

The following diagram depicts the time frame when each human, hardware, and software resource will be necessary, and when each will be available.
<table>
<thead>
<tr>
<th>Hardware Resources</th>
<th>MAY</th>
<th>JUNE</th>
<th>JULY</th>
<th>AUGUST</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEC 2065</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disk Space</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Terminal</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Printer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tape Drive</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software Resources</th>
<th>MAY</th>
<th>JUNE</th>
<th>JULY</th>
<th>AUGUST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface Software</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1022</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOPS 20</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Human Resources</th>
<th>MAY</th>
<th>JUNE</th>
<th>JULY</th>
<th>AUGUST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development Team</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Review Committee</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>End-User Data Entry</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Unknown</th>
<th>Needed</th>
<th>Available</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Cost/Benefit Appraisal

Several categories of cost may be incurred during the development of a software product. These are directly related to the resources necessary, and those already available. As seen in the previous section, all hardware resources are available, therefore, under the assumption of development proceeding on the DEC 2065, no funds will be required for the purchase of additional hardware. If, however, transition is made from the DEC 2065 to a microcomputer, hardware purchases will have to be made. From a survey of market availability, this figure would be approximately $5,000.00 to $8,000.00, depending on the sophistication and capabilities of the desired computer system.

Software resources are also available on the DEC 2065, therefore, no software purchases must be made for the present development effort. The cost of data base and operating system software for
microcomputer development is included in the hardware cost figure above, should this become a future option.

The final cost to be discussed is that of human resources. Because of the nature of this project as a coursework requirement for the development team members, team development effort will require no funding. However, should the proposed system be converted to a microcomputer system, it will be necessary to fund persons for conversion. The cost of data entry can be estimated by the amount of time it will take to enter all current records into the data base system. With an estimation of approximately 20,000 records of key holders, and another 27,000 records concerning inventory and control records, and with the assumption that 500 records may be entered daily, the cost for data entry may approach $2,500.00.
The cost saved by the proposed system is harder to estimate. Time will be saved by using an on-line system, however, the major source of savings will be realized by the increased return rate of keys because of the accuracy and interface possibilities of the proposed system. The money saved in this area will be realized by the reduction of the necessity to rekey buildings because of the loss or non-return of keys, which, as discussed in the proposal document of this system, may approach $7,000.00 to rekey a building. In addition, more efficient use of the locksmith's time may be gained by prediction of key use and the necessity to make more keys as supplies get low. Furthermore, front office procedures may be speeded up to allow office personnel more efficient use of their time.
Conclusion

In summary, this document provides the reader with an understanding of the processes of the current system, both physical and logical. Requirements for the implementation of the proposed system have been discussed and constraints on development have been described. System functions and data within the system have been defined, and necessary resources delineated. From the information contained within this narrative, progress may continue on the product design phase of development.
Schedule

Following is a projected schedule for the completion of remaining phases in the proposed project. These dates are approximate, due to both the inexact art of estimation and the difficulty of scheduling review meetings at the end of each phase.

5/13/85  -  Software Plans and Requirements
5/15/85  -  Validation
6/3/85   -  Product Design
6/5/85   -  Verification
6/26/85  -  Detailed Design
6/28/85  -  Verification
7/10/85  -  Coding and Unit Testing
7/31/85  -  Integration Testing
8/9/85   -  Final Implementation and User Manual
Bibliography


Administrative Development, University of Montana Computer C
DESIGN DOCUMENT

KEY INVENTORY CONTROL

by

Michele Miley

Robin Fauntleroy

June 26, 1985
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>178</td>
</tr>
<tr>
<td>Design Decomposition</td>
<td>184</td>
</tr>
<tr>
<td>Data Flow Diagrams</td>
<td>187</td>
</tr>
<tr>
<td>Data Dictionary</td>
<td>203</td>
</tr>
<tr>
<td>Hierarchical Charts</td>
<td>215</td>
</tr>
<tr>
<td>Functions and Internal Interfaces</td>
<td>232</td>
</tr>
<tr>
<td>Algorithmic Descriptions</td>
<td>248</td>
</tr>
<tr>
<td>File Descriptions</td>
<td>292</td>
</tr>
<tr>
<td>Requirements Mapping</td>
<td>294</td>
</tr>
<tr>
<td>Requirements Table</td>
<td>294</td>
</tr>
<tr>
<td>Requirements Cross-Reference</td>
<td>296</td>
</tr>
<tr>
<td>User Interface</td>
<td>298</td>
</tr>
<tr>
<td>State Transition Diagram</td>
<td>299</td>
</tr>
<tr>
<td>Menu Descriptions</td>
<td>305</td>
</tr>
<tr>
<td>Memory Use</td>
<td>313</td>
</tr>
<tr>
<td>Hardware Considerations</td>
<td>316</td>
</tr>
<tr>
<td>Performance Considerations</td>
<td>317</td>
</tr>
<tr>
<td>Development Projections</td>
<td>319</td>
</tr>
<tr>
<td>Maintenance Considerations</td>
<td>320</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>323</td>
</tr>
<tr>
<td>References</td>
<td>324</td>
</tr>
</tbody>
</table>
Appendixes

A. Layout of DMD Files .................................. 325
B. Data Structure Layout ................................. 328
C. File Relationships ................................. 330
D. Standard Module Header .............................. 331
E. Standard Menu & Prompt Layout .................... 332
F. Sample Letter Format ................................. 334
G. Sample Reports ...................................... 335
H. File Naming Conventions ............................ 337
Introduction

The software design phase of the life-cycle attempts to determine 'how' to implement the 'what' that was specified in the requirements phase. This is considered the structural design of the program or system; that is, what are the appropriate subsystems, programs, or modules, and how are they interconnected? In other words, it describes an acceptable programming solution to the problem specified in the requirements document. This process will determine the major characteristics of the final system, establishes the upper bounds in performance and quality that the best implementation can achieve, and may even determine what the final cost may be. The overall technical objective function of the design phase is made up of varying amounts of emphasis on efficiency, maintainability, modifiability, generality, flexibility, and utility.

A good design specification shows the solution in two ways, in terms of function and logic. The functional description shows what the system is to do; the logic description shows how the sys-
tem is actually structured to provide those functions. Tools are used to describe to the reader how these requirements are to be fulfilled. A design document may include some or all of the following tools available:

flow chart - a diagram that combines symbols and abbreviated narrative to describe a sequence of operations in a program system.

HIPO - consists of (1) a set of diagrams which show the functional breakdown of a system in the form of traditional hierarchy charts and (2) separate diagrams which explode each box on the hierarchy chart into a set of three boxes showing inputs, processes, and outputs.

pseudo-code - a notation similar in look, form, and meaning to both spoken language and programming languages; a bridge between the two.

structured charts - a pictorial way of expressing program logic, or structure, in a rigorous way, readable from top to bottom.

data flow diagrams - describes the system functionally, with symbols and text to describe a sequence of operations, but has no regard for the actual system structure.

decision tables - a simple, convenient way of showing what action is to be taken if a given condition or set of
conditions exist.

coverage matrices - a means of showing the relationship between two kinds of information.

storage maps - pictures describing how the various storage devices are being utilized.

simulation models - the system being modeled is expressed in some computer language and the computer executes the resultant programs in imitation of the described system.

To establish criteria for evaluation of a 'good' design, the following guidelines are presented:

1) A design should exhibit a hierarchical organization that makes intelligent use of control among elements of software.

2) A design should be modular; that is, the software should be logically partitioned into elements that perform specific functions and subfunctions.

3) A design should lead to modules (e.g., subroutines or procedures) that exhibit independent functional characteristics.

4) A design should be derived using a repeatable method that is driven by information obtained during software requirements analysis.
This design document is written for the Key Inventory Control System for the Physical Plant at the University of Montana. For a complete description of this system as it exists today and the proposed system, the reader is referred to the proposal (written April 4, 1985) and the Feasibility Document (written April 18, 1985). However, a brief overview of the system will be presented here to refresh the reader with the system under design.

There are two major divisions within the current key inventory system control system. The first is the actual locksmith operation, in which not only are keys made, but records are kept of which keys open which locks. The locksmith must maintain records of key numbers currently in use, those available for use, those retired, and the length of retirement. Furthermore, an inventory must be maintained of key blanks and other lock components. Presently, the locksmith is notified as particular keys are disbursed and supplies are
depleted, thereby necessitating their duplication. The second major portion of the key inventory system is key issue control. Records must be kept of all persons possessing a given key, the deposits received, keys issued to an individual, keys returned by an individual and deposit refunds. Although the procedure for this is relatively straightforward, the amount of information that must be processed is quite extensive. Accuracy is essential for university security. The only account of key distribution is maintained by this office.

Recently, an unofficial audit was performed by State of Montana auditors on the key inventory control system, which suggested improvements might be in order. Because of the large number of keys involved, it has become extremely difficult and tedious to perform manual searches of records for desired information. Furthermore, much valuable employee time is consumed in this process. With the amount of transactions performed daily in the
office, the possibility of an error exists. In addition, prompt service is often requested and may not be possible due to the current manual system.
Design Decomposition

The principles of structured design were used to specify the design of the key inventory control system. Structured design has two main principles: modules should be weakly coupled to each other and should exhibit as strong a level of cohesion as possible.

Coupling is defined as a measure of the strength of interconnection between modules. Thus, "highly coupled" modules are joined by strong interconnections and "loosely coupled" by weak interconnections. In other words, the more that we must know about module B in order to understand module A, the more closely connected A is to B. The factors that influence coupling are:

type of connection between modules;
complexity of interface;
type of information flow along the connection;
and, binding time of the connection.
Cohesion of each module is defined as how tightly bound or related its internal elements are to one another. An uncohesive module is one that may perform several tasks, whereas a highly cohesive module performs only one task. There are seven levels of cohesion and are listed here in order of increasing strength of cohesion.

- coincidental cohesion
- logical cohesion
- temporal cohesion
- procedural cohesion
- communicational cohesion
- sequential cohesion
- functional cohesion

Cohesion and coupling are interrelated and generally, the greater the cohesion of individual modules in the system, the lower the coupling between modules will be. The major aim, then of structured design, is to produce modules which are functionally cohesive (execute one function only) and are loosely coupled (do their task with a minimum of information from other modules).
The method of transform analysis was used to derive an initial structural design. This method is a form of top-down strategy, which generally requires only a modest restructuring to arrive at a final design. Transform analysis consists of the following four steps:

restating the problem as a data flow graph (from Requirements document)

identifying the afferent and efferent data elements

first-level factoring

factoring of afferent, efferent, and transform branches

The following section of this design document specifies the design for the key inventory control system. Data flow diagrams follow (from the Requirements Document) with the afferent, efferent and transform elements identified. Next is included the data dictionary which defines all terms used in the data flow diagrams. The hierarchical charts are included next and are fol-
lowed by the overall description and interface specifications; and the algorithmic description. These two descriptions, written in both spoken English and pseudo-code should provide an unambiguous description of the modules. The last part of this section will include descriptions of the files necessary for storage of information within the system.

Data Flow Diagrams

The following pages contain a pictorial representation of the data flows present in the proposed system. They are marked too indicate the afferent, transform and efferent processes within the system.
LEVEL 2: DISTRIBUTE INFO

Distribute Info 1.1

Afferent
- INFO REQUEST
- IND_RECORD
- NAME
- LOCATION
- INVENTORY
- CHECKBOOK

Transform

Effferent
- INFO
LEVEL 2: FIND NAME & SSN

AFFERENT

PERSONNEL

REGISTRAR

NAME

FIND SSN

1.2

FIND NAME

1.9

TRANSFORM

SSN

NAME

EFFERENT
LEVEL 2: ORDER KEY

ORDER KEY

1.7

ORDER DATE

AFFERENT

INVENTORY

TRANSFORM

EFFERENT

ORDER KEY
LEVEL 3: GENERATE CORRESPONDENCE

AFFERENT

GENERATE LETTERS 1.6.1

GENERATE REPORTS 1.6.2

PERSONNEL

TERMINATION

LIST

TRANSFORM

QUERY

INVENTORY

LOCATION

LETTER

NAME

REPORT

KEYWAY
LEVEL 1: LOCKSMITH
CONTEXT DIAGRAM

LOCKSMITH PROCESS
1.2

SUPPLIER

FUND REQUEST

U OF M ADMINISTRATION

REKEY REQUEST

PART

MAINTENANCE REQUEST

INVENTORY FILE

LOCATION FILE

KEY ORDER FILE

KEY FILE

ORDER FILE
LEVEL 3: REKEY LOCK

INVENTORY FILE

TRANSFORM

GENERATE KEY NUMBER 2.3.1

DETERMINE LOCK NUMBER 2.3.2

PIN LOCK 2.3.3

INSTALL LOCK 2.3.4

PART FILE

AFFERENT

REKEY REQUEST

LOCK NUMBER

Pinned Lock

REKEY ORDER
Data Dictionary

Entries in the following pages provide a concrete definition for the flow of data between system processes. Each piece of data describes an item of information needed for a process to perform its function, or output desired by the users. Curly braces indicate that there may be a repetition of information within a data element, a plus sign indicates 'and', and a vertical bar indicates 'or'. Parentheses depict an item that is optional. Comments about legal data values are enclosed by a slash and an asterisk. Data flows for both the front office and locksmith procedures are defined together, and both the current and proposed system may be found in this section. Descriptions of files and file layouts along with data structures to be used may be found in the appendixes.

allow entry = /* flag returned from Password */
/* value is true or false */
amount = dollar amount of deposit |
        dollar amount of refund

availability = /* flag to indicate key issue
               availability */
               /* value is true or false */

bank deposit = dollar amount +
              date +
              account number
building = Aber Hall |
          Alumni Center |
          Art Annex |
          Botany |
          Brantly Hall |
          Business Administration |
          Chemistry-Pharmacy |
          Corbin Hall |
          Craig Hall |
          Duniway Hall |
          Elrod Hall |
          Field House |
          Fine Arts |
          Forestry |
          Health Science |
          Health Service |
          Heating Plant |
          Jesse Hall |
          Journalism |
          Knowles Hall |
          Law |
          Lecture Hall |
          Liberal Arts |
          Library |
          Lodge |
          Mathematics |
          Men's Gymnasium |
          Miller Hall |
          Music |
          North Corbin Hall |
          Pharmacy-Psychology |
          Physical Plant |
          Jeanette Rankin Hall |
          Science Complex |
          Social Sciences |
          Swimming Pool |
          Turner Hall |
          University Center |
          University Hall |
          McGill Hall |
          1010 Arthur |
          600 Beckwith |
720 Beckwith |
612 Eddy |
616 Eddy |
626 Eddy |
724 Eddy |
730 Eddy |
1414 Maurice |
600 University |
Forestry Science Lab |
Forestry Science Admin |
Forestry Biolab |
Clinical Psychology Center |
Performing Arts |
Married Student Housing

date = date of deposit received

delete name flag = /* flag to indicate no keys held */
   /* value is true or false */

deposit refunds = name +
   date +
   dollar amount

fund approval = fund request +
   signature of approval
fund request = needed part count +
estimated dollar amount

ID = name |
social security number

info = information

info request = building + room number |
key number |
ID

information = keyholder |
key number |
location |
bank deposit |
inventory |
keys held

information requests = room number |
building |
{key owners}
key = key blank +
    key number

key list = { key number }
    /* all entries refer to same building
        and room number */

keycard = ID +
    building number +
    room number +
    key number +
    date

key number = bit cut number +
    keyway number

key order = key number +
    amount needed
key request = ID +
  date +
  user's position +
  building +
  room number +
  approval signature +
  deposit amount

key return = ID +
  key number

letter = letter to terminated personnel | letter to graduating students

list = name +
  SSN +
  address
  /* list of graduating students */

location = building +
  room number
lock number = key number

maintenance request = building number +
    room number +
    description of problem

married student housing = Garnet Court |
    Bannack Court |
    Rimini Court |
    Yreka Court |
    Pioneer Court |
    Helena Court |
    Craighead

menu choice = /* flag to indicate user's choice
    of function */

name = /* name of keyholder */
    /* stored in Personnel, Registrar, or
        Name File */
name issue info = name +
    key number +
    deposit amount +
    date +
    (social security number)

name return info = name +
    key number +
    (social security number)

needed part count = part +
    part number +
    amount needed

new_key_number = key number

part = lock |
    pin |
    pinned lock |
    key blank
part count = part + 
   part number + 
   amount on hand

part number = lock number | 
   keyway number | 
   pin size

part order = needed part count + 
   fund approval

password = /* word given to gain access to 
   program portions */

pinned lock = lock + 
   lock number

query = keyholder report | 
   inventory report | 
   locksmith report | 
   location report
receipt = receipt info

receipt info = name +
   amount +
   date

record info = ID +
   key number +
   date +
   amount

refund = refund info +
   dollar amount of refund

refund info = name +
   amount +
   date

rekey order = key order +
   building +
   room number
rekey request = building name +
               (room number)

request info = info request

requested = /* flag to indicate user request for
            maintenance program entry */
            /* value is true or false */

report = keyholder  |
         inventory   |
         locksmith  |
         location

restore request = requested

room number = number assigned to given room
ssissue info = social security number +
key number +
deposit amount +
date

SSN = social security number

ssreturn info = social security number +
key number

temp list = SSN +
{  key number  }

user choice = menu choice

Hierarchical Charts

The following pages contain a pictorial representation of the modular structure of the proposed program solution. Both flow of data and
flow of control are indicated.
222

DISTRIBUTE INFO

DISPLAY INFO

DISPLAY KEYS HELD

DISPLAY INVENTORY

DISPLAY DEPOSIT

DISPLAY LOCATION

DISPLAY KEY NUMBER

DISPLAY KEYHOLDER

SSN

NAME

1.1.6

1.1.3.6

1.1.3.5

1.1.3.4

1.1.3.3

1.1.3.2

1.1.3.1

1.1.3.7
GENERATE CORRESPONDENCE

GENERATE LETTERS

TEMP LIST

GENERATE STUDENT LETTERS

NAME

SSN

1.1.4.1.1

TEMP LIST

GENERATE PERSONNEL LETTERS

1.1.4.1.2

PRINT LETTERS

1.1.4.1.3

TEMP LIST

GENERATE REPORTS

1.1.4.2

KEYHOLDER REPORT

1.1.4.2.1

INVENTORY REPORT

1.1.4.2.2

LOCATION REPORT

1.1.4.2.4

LOCKSMITH REPORT

1.1.4.2.3
BOOKKEEPING,
FIND SSN,
FIND NAME

BOOKKEEPING
1.1.3

FIND NAME
1.1.7

FIND SSN
1.1.6

AMOUNT

SSN

NAME

NAME

SSN
REKEY LOCK

KEY NUMBER 1 e r

1.2.3

NEW - KEY GENERATE NEW NUMBER

BLDG, ROOM NO

ACCEPT A

UPDATE FILES

GET NEW NUMBER

FOUND

NEW KEY

CHECK APPLICABILITY

GENERATE NEW NUMBER

1.2.3.1.1

1.2.3.1.2
Functions and Internal Interfaces

Main Driver and Password Modules

Module 1 of the Key Inventory Control System, Main, is the main driver module of the entire program. It allows the user to choose either the front office, locksmith, or maintenance functions of the system, and once a choice is entered, it calls the password routines to determine if the person attempting access should be allowed entry. Because there are different passwords to different portions of the system, the menu choice is passed to the password routines so they may determine if the correct password has been entered for that particular part of the system. Main receives a flag, allow_entry, back, and it is either true or false depending upon the correct entry of the appropriate password.

Module 1.0, Password, is the controlling module of the password routines. It calls Accept_Password, and receives the allow_entry flag
from this module. Depending on how this flag is set, it may or may not call Disallow_Access, and pass it the menu_choice. Accept_Password prompts the user for the appropriate password depending on the portion of the system to be accessed. If the user enters the password incorrectly the first time, he is allowed a second chance. However, a second mis-entry will result in setting the allow_entry flag to false, and Disallow_Access will be called by the main Password routine. Disallow_Access, module 1.0.2, calls Send_Mail, passing it the menu_choice, and Logout_User, both batch or system processes. Send_Mail will leave a message in the user area concerning the attempted access, and the portion of the system which was chosen, and Logout_User will allow the program to terminate normally, then log the illegal user off the system.

Front Office Modules
The front office portion of the program is controlled by module 1.1, Front Office Main. It displays the front office menu (shown in the user interface section), and calls the appropriate module depending on the user's choice of function. A password is required to enter this section of the system, however, once the user is in this area, any number of front office functions may be performed without re-entering a password. As with all menus in this program, if an incorrect choice is entered, an error message is displayed, and the user is asked to re-enter the desired choice.

Module 1.1.1, Issue Key, is the driver routine for entry of key issue information. When an individual requests a key from the Physical Plant, this program portion is used. Issue Key prompts the user for the necessary information, and then calls the appropriate Find Key Number, passing it the building and room number. Find Key Number, module 1.1.1.1, determines if the building and room number are valid, and if so, calls module
1.1.1.1.1.1, Check Inventory, passing it the list of possible key numbers. Check Inventory checks to make sure there is a key available for issue, of the key type requested, and returns the key number and availability status to Find Key Number. If a key is available for issue, Issue Key will then call module 1.1.1.2, Update Name Files, passing it record_info. Update Name Files calls Find SSN if the social security number has not been given, and enters the appropriate information in both the Ind_Record File and the Name File if necessary. In addition, Update Name Files calls the Bookkeeping module, passing it the amount of deposit paid by the individual requesting a key.

Module 1.1.2, Return Key, is called when a keyholder returns a key to the Physical Plant. It prompts the user for the keyholder's name and social security number, and if no social security number is given, it calls Find SSN to determine the correct number. It also requests the key number being returned. Check All Names, module
1.1.2.1, is then called, passing it the entire key return record. Check All Names checks the Ind_Record File and Name File for entries concerning the holding of that key, and if the correct entry is found, it calls Delete Files, module 1.1.2.1.1, passing it the keyholder ID, and a flag indicating whether the name may be deleted if the individual possesses no more keys. Increase Key Available is also called by Check All Names, and it increases the quantity on hand field of the Inventory File. It is passed the returned key id number.

Distribute Info, module 1.1.3, is called if the user requests the information access option of the front office portion of the program. It displays a menu allowing the user a choice of types of information that may be requested. Depending on the user choice, Display Keyholder, Display Key Number, Display Location, Display Deposit, Display Inventory, or Display Keys Held may be called. Display Keyholder, module 1.1.3.1,
displays a menu allowing the user to choose what information to enter to find the holders of a particular key. Building and room may be entered to find all holders of keys to that room, and key numbers may be entered to find all individuals possessing that particular combination of keys. All holders of a given key may be found by entering only one key number. This function calls Find Name in order to produce the name of an individual rather than a social security number.

Display Key Number, module 1.1.3.2, allows the user to enter a building and room number, and it will display all the key numbers fitting that particular lock. Module 1.1.3.3, Display Location, will use a key number to determine which door or doors it will open in which building. Module 1.1.3.4, Display Deposit, will display the balance of cash flow for that particular day, up to the time it is called. If it is being used for daily balancing purposes, the user has the option of clearing out the total when it is called. Display
Inventory, module 1.1.3.5, will display the number of keys on hand, the number out, and the total number, given a particular key number. Given a social security number or name, module 1.1.3.6, Display Keys Held, will list each key held by that individual, the date it was issued, and the deposit paid for each key.

Module 1.1.4, Generate Correspondence, is the controlling module for the generation of letters and reports. It displays a user menu, and dependent upon the user's choice, may call either Generate Letters or Generate Reports. Generate Letters will allow the user to generate letters to graduating students who still have keys, and to personnel leaving the University system. Module 1.1.4.1.1, Generate Student Letters, uses address labels to be provided by the Registrar's Office and manual input of the social security numbers on those labels to determine if a student possesses a key, and if so, it adds that student and the outstanding keys to a list. Generate Personnel Letters
accesses Personnel files to determine terminated employees and, if they have keys, puts their social security numbers and keys held in a list. It also prints address labels for those on the list. Module 1.1.4.1.3, Print Letters, uses the list produced by the previous two modules to print letters to each of the individuals on the list. Examples of the letter sent may be found in the appendix portion of this document.

Generate Reports, module 1.1.4.2, is the controlling module for the report generation function of the system. It displays a menu allowing the user to choose the desired report. Depending on user choice, it may call Keyholder Report, Inventory Report, Key Retirement Report, or Location Report. Module 1.1.4.2.1, Keyholder Report, produces a listing of all individuals holding keys, the keys in their possession, dates issued, and deposits paid. Inventory Report lists all key numbers, the quantity on hand, the quantity out, and the total number. Location Report, module
1.1.4.2.4, lists all key numbers and the buildings and room numbers that they will open. The Key Retirement module produces a report of all retired keys, the quantity on hand, quantity out, total number, and date of retirement. Sample reports may be found in the appendix section of this document.

Module 1.1.5, Bookkeeping, calls no other modules. It is passed an amount by either the Return Key or Issue Key functions, and it updates the current Checkbook balance. At the end of the day, this balance may be used to balance the cash drawer, and then cleared out for the following days transactions.

Module 1.1.6, Find SSN, uses a name to search Personnel and Registrar Files to find the matching social security number. It is called by a number of modules, including the support modules to be used during data entry. Find Name, module 1.1.7, performs the reverse function, in that, given a
social security number, it will search the appropriate files for the matching name.

Locksmith Modules

Module 1.2, Locksmith Main, is the controlling module for the locksmith portion of the program. This segment of the program is mostly menu run. A sample of the menu format can be seen in Appendix E. For further specification of the menus, the reader is referred to the section on the User Interface. The main locksmith module calls one of four modules, depending on the choice given by the user in response to the menu. These modules are: Complete Key Orders, Replace Key, Rekey Lock, and Inventory Control.

Module 1.2.1, Complete Key Orders allows the user flexibility with the Order Key file. It offers the user four choices. Module 1.2.1.1, List Orders, simply prints the list of orders, in order of priority and date, to the screen. This provides a means of determining what the keys to
be made are. Module 1.2.1.2, Change Orders, provides the user with a means of changing an existing order, whether it was input incorrectly or a need exists to change part of the order. This module calls Module 1.2.1.2.1, Number Order File, which takes the order file and lists it to the screen with numbers prefixing each order. To record the completion of an order, Module 1.2.1.3, Record Completion is called. This module also calls Number Order File, as it needs the orders listed to the screen. Finally, Print Orders, Module 1.2.1.4 prints to the line printer a copy of the Order Key file.

Module 1.2.2, Replace Key, is a means of acknowledging the fact that keys become broken and ineffective. It simply accesses the Inventory file and subtracts one from the total_amount and quantity_on_hand. (It assumes that as an individual returns a broken key, they will have that key replaced with another key with no paperwork involved in the transaction.)
Module 1.2.3, Rekey Lock, allows the user to either generate an entirely new number for a rekeying of a lock, or to determine if a key retrieved from the retired key file is eligible to be used. The user is prompted for the building and room number of the lock being rekeyed, and then Module 1.2.3.1, Get New Number is called. The user is prompted for the use of a new key or an existing, retired one. If the user requests a new key, Module 1.2.3.1.1, Generate New Number is called, and if an existing key is requested, the user is prompted for the existing key number. In either case, once a number has been requested for use, Module 1.2.3.1.2, Check Applicability, is called to determine whether that key number may be used. Once an eligible key number has been determined, the user is asked if he wishes to use that number. If so, Module 1.2.3.2, Update Files is called, which updates all the appropriate files.

Module 1.2.4, Inventory Control, is provided to allow the user access to the inventory files.
It provides the user five choices, dependent upon a choice made from a menu. Module 1.2.4.1, Change Inventory, will change the amount field in the Keyway file. Order_Parts, Module 1.2.4.4, will print a list of those items within the keyway file which need to be ordered (are below the threshold number). It calls Module 1.2.4.4.1, which checks each record in the keyway file for the amount and checks that against the threshold number. It will create an internal file of keyway numbers to be ordered, and passes that file back up to Order_Parts to be printed. Add_Item, Module 1.2.4.2, allows the user to add new items to the inventory list. It will prompt for the new keyway number and an amount if it exists. It then calls Update File, Module 1.2.4.2.1, which inserts a new record in the keyway file with the appropriate keyway number and amount. Module 1.2.4.3, Print Inventory, prints to the line printer a copy of the information stored in the Keyway file. Delete Item, Module 1.2.4.5, prompts the user for an item
which needs to be removed permanently from the Keyway file. It then deletes that record from the file.

Maintenance Modules

Module 1.3, Maintenance Main, is the controlling module for the maintenance portion of the program. For the most part, this portion of the program runs automatically, with the exception that a user may request that desired files stored on magnetic tape be retrieved. This module calls Backup Info, Order Key and Update Key Status, although the actual execution of these modules depends on the current system date and the date they were last executed. Backup Info is passed the flag "requested" to indicate whether the user has requested entry to the maintenance portion of the program or whether it is being called automatically. Backup Info controls the storage and retrieval of information on magnetic tape. It calls Save and Restore. The function of Save is to
back up current files weekly. Restore is a user requested process which will pull the desired files from magnetic tape and put them in the user area.

Process 1.3.2, Order Key, is designed to search the Inventory File daily, and if keys are below the threshold level, it will place a record of this in the Order Key File to be accessed by the locksmith portion of the program.

Module 1.3.3, Update Key Status, is executed periodically. Its purpose is to increment through the Inventory file, checking for keys which have been retired. If the key status is retired, but the date of retirement is greater than one year from that date, the status field is then set to available.

Data Entry Modules

Module 2, Data Entry Main, is the controlling module for the support processes to be used during
initial data entry. It calls Accept Name and Enter Info. Accept Name, module 2.1, prompts the user for the keyholder name, and calls Find SSN to determine the correct social security number. Both name and social security number are passed to the calling procedure. Enter Info accepts the name and social security number passed to it from Data Entry Main and enters these in the appropriate files. The user is then prompted for additional information to be entered and this information is also entered in the appropriate file.
Algorithmic Descriptions

Front Office Processing Narratives

Module Name: l Main
Parameters:
  In: none
  Out: menu_choice, allow_entry
  In/Out: none
Module Description:

display main menu
set allow_entry to false
prompt user for menu_choice
if not a valid choice, then
  repeat
    display error message
    prompt for choice
  until menu_choice is valid
set requested to false
call Maintenance_Main(requested)
if menu_choice is quit, then
terminate program run
call Password(menu_choice)
if allow_entry is true, then
  repeat
    if menu_choice is front office, then
      call Front_Office_Main
    else, if menu_choice is locksmith, then
      call Locksmith_Main
    else, if menu_choice is maintenance, then
      set requested to true
      call Maintenance_Main(requested)
    until menu_choice is quit
else, terminate program run.
Module Name: 1.0 Password
Parameters:
    In: menu_choice
    Out: allow_entry
    In/Out: none
Module Description:

if menu_choice is for front office, then
    prompt for front_office_password
    call Accept_Password(front_office_password)
else, if menu_choice is locksmith, then
    prompt for locksmith_password
    call Accept_Password(locksmith_password)
else, if menu_choice is maintenance, then
    prompt for maintenance_password
    call Accept_Password(maintenance_password)
if allow_entry is false,
    print incorrect password message
    prompt user for password re-entry
    call Accept_Password(password)
if allow_entry is false,
    call Disallow_Entry(menu_choice).
Module Name: 1.01 Accept Password
Parameters:
  In: none
  Out: password, allow_entry
  In/Out: none
Module Description:

if menu_choice is front office
  and front_office_password is correct,
  OR menu_choice is locksmith
  and locksmith_password is correct,
  OR menu_choice is maintenance
  and maintenance_password is correct,
    then set allow_entry to true
else,
  set allow_entry to false.

Module Name: 1.0.2 Disallow Access
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

call Send_Mail
call Logout_User
Module Name: 1.0.2.1 Send Mail
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:
Send mail about system entry attempt.

Module Name: Logout User
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:
delay for program termination
log user off system.
Module Name: 1.1 Front Office Main
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

repeat
  display Front Office Menu
  /* see user interface section */
  prompt for menu_choice
  if menu_choice is not valid, then
    repeat
      display error message
      prompt for menu_choice
    until menu_choice is valid
  if menu_choice is issue key, then
    call Issue_Key
  else, if menu_choice is return key, then
    call Return_Key
  else, if menu_choice is distribute info, then
    call Distribute_Info
  else, if menu_choice is generate correspondence, then
    call Generate_Correspondence
  until menu_choice is quit.
Module Name: 1.1.1 Issue Key
Parameters:
    In: none
    Out: none
    In/Out: none
Module Description:

set available to false
prompt for name
prompt for social security number
if social security number is unknown, then
    set social security number to negative number
prompt for building abbreviation
prompt for room number
call Find Key Number(building, room)
if available is true, then
    call Update_Name_Files(record_info)
else,
    display insufficient supply message.

Module Name: 1.1.1.1 Find Key Number
Parameters:
    In: building, room number
    Out: key number, availability
    In/Out: none
Module Description:

open Location File
find all entries matching building
find all entries matching room number
if found, then
    call Check_Inventory(key_list)
else,
    display invalid building/room message
close Location File.
Module Name: 1.1.1.1.1 Check Inventory
Parameters:
   In: key list
   Out: availability, key number
   In/Out: none
Module Description:

open Inventory File
prompt user for key_type requested
repeat
   find record matching top entry on key list
   if key_type matches, then
      if quantity_on_hand greater than 0, then
         set found to true
         set key_number to key_id_number field
      else,
         discard top item on key_list
   until found or key_list is empty
if found, then
   decrease quantity_on_hand by one
else,
   display insufficient supply message
   set available to false
close Inventory File.
Module Name: 1.1.1.2 Update Name Files
Parameters:
   In: record_info
   Out: none
   In/Out: none
Module Description:

open Ind_Record File and Name File
if social security number is less than 0, then
call Find_SSN(name)
if social security number is less than 0, then
   add name and social security number to Name File
add social security number, key_number, deposit, to Ind_Record File
call Bookkeeping(deposit)
close Ind_Record and Name Files.

Module Name: 1.1.2 Return Key
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

prompt for keyholder name
prompt for social security number
if social security number is unknown, then
call Find_SSN(name)
if social security number is unknown, then
   open Name File
   find name in Name File
   set social security number to SSN in Name File
   close Name File
prompt for key_number
call Check_All_Names(key_number).
Module Name: 1.1.2.1 Check All Names

Parameters:

In: key_return
Out: refund_info
In/Out: none

Module Description:

open Ind_Record File
find all_entries matching ID
if Synrec is 1, then
   set delete_name_flag to true
repeat
   match key_id_number in file to key_number in top record
   if they match, then
      set found to true
      with refund_info record, do
      set amount to (-1 * deposit)
      set date to Sysdate
      call Bookkeeping(amount)
   until found or no records left
if found, then
   call Delete_Files(ID, delete_name_flag)
else,
   display no record found message
close Ind_Record File
call Increase_Key_Available(key_number).
Module Name: 1.1.2.1.1 Delete Files
Parameters:
   In: ID, delete_name_flag
   Out: none
   In/Out: none
Module Description:

in Ind_Record File  /* file is already open */
   delete current record
if delete_name_flag is true, then
   open Name File
   find matching name or social security number
   delete that record
   close Name File.

Module Name: 1.1.2.1.2 Increase Key Available
Parameters:
   In: key_number
   Out: none
   In/Out: none
Module Description:

open Inventory File
find record matching key_number
increment quantity_on_hand by one
close Inventory File.
Module Name: 1.1.3 Distribute Info
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

display Information Access Menu
   /* see user interface section */
repeat
   prompt for menu_choice
   if menu_choice is keyholder, then
       call Display_Keyholder
   else, if menu_choice is key numbers, then
       call Display_Key_Number
   else, if menu_choice is building and room number, then
       call Display_Location
   else, if menu_choice is cash balance, then
       call Display_Deposit
   else, if menu_choice is key inventory, then
       call Display_Inventory
   else, if menu_choice is keys held, then
       call Display_Keys_Held
   else, if menu_choice is quit, then
       do nothing for now
   else,
       display invalid choice message
until menu_choice is quit.
Module Name: Display Keyholder
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

display Keyholder Menu
repeat
   prompt for menu_choice
   if menu_choice is building/room, then
      prompt for building and room number
      open Ind_Record File
      find all records matching building
      and room
      make temporary list of social security
      numbers
      close Ind_Record File
      for each social security number on the list,
      call Find_Name(social_security_number)
      put name in list
      display name_list
   else, if menu_choice is key_numbers, then
      repeat
         prompt for key_number
         put key_number in list
      until no key_numbers given (CR)
      open Ind_Record File
      find all entries matching each key_number
      in list
      put social security numbers in temporary
      list
      close Ind_Record File
      for each social_security_number in list,
      call Find_Name(social_security_number)
      put name in name_list
      display name_list
   else, if menu_choice is quit, then
      do nothing
   else,
      display invalid choice message
until menu_choice is quit

Module Name: 1.1.3.2 Display Key Number
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

prompt user for building
prompt user for room number
open Location File
find all records matching building and room
for each record,
  display key_id_number
close Location File.

Module Name: 1.1.3.3 Display Location
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

prompt user for key_number
open Location File
find all records matching key_number
for each record,
  display building
  display room number
close Location File.
Module Name: 1.1.3.4 Display Deposit
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

open Checkbook File
display balance in file
ask user if end of day
if yes, then
   set balance to 0
close Checkbook File.

Module Name: 1.1.3.5 Display Inventory
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

prompt user for key_number
open Inventory File
find all records matching key_number
for each record /* there should only be one */
   display quantity_on_hand
   set total_out to
      (total_number - quantity_on_hand)
   display total_out
display total_number
close Inventory File.
Module Name: 1.1.3.6 Display Keys Held
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

    prompt for social_security_number
   if none (CR), then
      prompt for keyholder name
      call Find_SSN(name)
      set social_security_number to SSN returned
   open Ind_Record_File
   find all records matching social_security_number
   for each record,
      display key_id_number
      display date_out
      display deposit
   close Ind_Record_File.
Module Name: 1.1.4 Generate Correspondence
Parameters:
  In: none
  Out: none
  In/Out: none

Module Description:

repeat
  prompt user for letter or report
  if choice is letter, then
    call Generate_Letters
  else, if choice is report, then
    call Generate_Report
  else, if (CR), then
    do nothing
  else,
    display invalid choice message
until choice is quit (CR).
Module Name: 1.1.4.1 Generate Letters
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

repeat
  prompt user for student or personnel letters
  if choice is student, then
    call Generate_Student_Letters
    call Print_Letters(temp_list)
  else, if choice is personnel, then
    call Generate_Personnel_Letters
    call Print_Letters(temp_list)
  else, if choice is quit (CR), then
    do nothing
  else,
    display invalid choice message
until choice is quit (CR).
Module Name: 1.1.4.1.1 Generate Student Letters
Parameters:
   In: none
   Out: temp_list
   In/Out: none
Module Description:

open Ind_Record File
repeat
    prompt user for social security number
    find all records matching social security number
    if no records, then
        display no keys outstanding message
    else,
        put social security number in temp_list
        for each record,
            put key_id_number in temp_list
        put end_of_keys flag in temp_list
        display found outstanding keys message
    until no more social security numbers (CR)
close Ind_Record File.
Module Name: 1.1.4.1.1 Generate Personnel Letters

Parameters:
- In: none
- Out: temp_list
- In/Out: none

Module Description:

open Terminated File
for each entry in Terminated File,
    put social security number in ssn_list
close Terminated File
open Ind_Record File
for each entry in ssn_list,
    find matching record in Ind_Record File
    if none, then
        delete social security number from ssn_list
    else,
        put social security number in temp_list
        for each record,
            put key_number in temp_list
        put end_of_keys flag in temp_list
close Ind_Record File
open Personnel File
for each entry on ssn_list
    find matching entry in Personnel File
    with name and address fields,
        print address label
close Personnel File.
Module Name: 1.1.4.1.3 Print Letters
Parameters:
   In: temp_list
   Out: none
   In/Out: none
Module Description:

for each social security number in temp_list,
   print letter heading
   for each key_number under ssn in temp_list,
      print key_number
   print letter closing
/* see Appendix for sample letter format */
Module Name: 1.1.4.2 Generate Reports

Parameters:
  In: none
  Out: none
  In/Out: none

Module Description:

display Generate Report Menu
  /* see user interface section */
repeat
  prompt user for menu_choice
  if menu_choice is keyholder, then
    call Keyholder_Report
  else, if menu_choice is inventory, then
    call Inventory_Report
  else, if menu_choice is locksmith, then
    call Locksmith_Report
  else, if menu_choice is location, then
    call Location_Report
  else, if menu_choice is quit, then
    do nothing
  else,
    display invalid choice message
until menu_choice is quit.
Module Name: 1.1.4.2.1 Keyholder Report
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

open Ind_Record File
print keyholder report heading
sort file by social security number
for each unique social security number in file
   call Find_Name(ssn)
   print name
   for each record with same social security number,
      print key number
      print date issued
      print deposit paid
close Ind_Record File.

Module Name: 1.1.4.2.2 Inventory Report
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

print inventory report heading
open Inventory File
sort by key number
for each entry,
   print key number
   print quantity on hand
   print (total_number - quantity_on_hand)
   print total number
close Inventory File.
Module Name: 1.1.4.2.3 Key Retirement Report
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

print key retirement report heading
open Inventory File
sort by key number
for each retired key,
   print key number
   print quantity on hand
   print (total_number - quantity_on_hand)
   print total number
   print date retired
close Inventory File.

Module Name: 1.1.4.2.4 Location Report
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

print location report heading
open Location File
sort by key number
for each unique key number,
   print key number
   for each entry with same key number,
      print building
      print room number
close Location File.
Module Name: 1.1.5 Bookkeeping
Parameters:
   In: amount
   Out: none
   In/Out: none
Module Description:

open Checkbook File
add amount to balance in Checkbook File
close Checkbook File.

Module Name: 1.1.6 Find SSN
Parameters:
   In: name
   Out: SSN
   In/Out: none
Module Description:

open Personnel File
find record matching name
if none, then
   close Personnel File
   open Registrar File
   find record matching name
   if none, then
      display not found message
   else,
      set social security number to SSN field
      in Registrar record
   close Registrar File
else,
   set social security number to SSN in
   Personnel File
close Personnel File.
Module Name: 1.1.7 Find Name
Parameters:
  In: SSN
  Out: name
  In/Out: none
Module Description:

if social security number is less than zero, then
  open Name File
  find record matching social security number
  if none, then
    display not found record
  else,
    set name to name in Name File
else,
  open Personnel File
  find record matching social security number
  if none, then
    close Personnel File
    open Registrar File
    find record matching social security number
    if none, then
      display not found message
    else,
      set name to name in Registrar File
      close Registrar File
    else,
      set name to name in Personnel File
  close Personnel File.
Module Name: 1.2 Locksmith Main
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

display Locksmith Main Menu
   /* see user interface section */
repeat
   prompt for menu_choice
   if menu_choice is Complete Key Orders, then
      call Complete Key Orders
   else if menu_choice is Replace Key, then
      call Replace Key
   else if menu_choice is Rekey Lock, then
      call Rekey Lock
   else if menu_choice is Inventory Control, then
      call Inventory Control
   else, if menu_choice is quit, then
      do nothing
   else, display invalid choice message
until menu_choice is quit
Module Name: 1.2.1 Complete Key Orders
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

display Complete Key Orders Menu
repeat
   prompt for menu_choice
   if menu_choice is List Orders, then
call List Orders
   else if menu_choice is Change Orders, then
call Change Orders
   else if menu_choice is Record Completion, then
call Record Completion
   else if menu_choice is Print Orders, then
call Print Orders
   else if menu_choice is quit, then
do nothing
   else display invalid choice message
until menu_choice is quit

Module Name: 1.2.1.1 List Orders
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

sort order file on date
sort order file on priority
list all orders that currently exist (on screen)
Module Name: 1.2.1.2 Change Orders
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

Number_Order_File
while not done
  prompt user for order number or symbol to quit
  if quit symbol
    then done set equal to true
  else if proper order number
    then prompt for new priority
      replace existing priority with new priority
      prompt for new amount
      replace existing amount with new amount
    else print error message

Module Name: 1.2.1.2.1 Number Order File
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

set counter to 1
open key order file
sort on date
sort on priority
while there are records in the file
  list the counter (to the screen)
  list the corresponding record (to the screen)
  increment the counter
Module Name: 1.2.1.3 Record Completion
Parameters:
In: none
Out: none
In/Out: none
Module Description:

Number_Order_File
while not done
prompt for order number completed or symbol to quit
if a correct order number
then map to Inventory file via key ID number
   increase total_amount by amount
   increase quantity_on_hand by amount
   return to order file
   delete record
else if quit symbol
then done set equal to true
else display error message

Module Name: 1.2.1.4 Print Orders
Parameters:
In: none
Out: none
In/Out: none
Module Description:

sort order file on date
sort order file on priority
print all (line printer)
Module Name: 1.2.2 Replace Key
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

   prompt for key number
   find key number in Inventory file
   subtract 1 from quantity_on_hand
   subtract 1 from total_amount

Module Name: 1.2.3 Rekey Lock
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

   prompt for building
   prompt for room number
   call Get New Number
   if accept_number is true, then
       call Update Files
Module Name: 1.2.3.1 Get New Number
Parameters:
  In: building, room number
  Out: accept_number, new_key_number, amount, order_amount, key_number
  In/Out: none
Module Description:

open Location File
find record matching building and room number
if it exists, then
  let key_number equal KIN
else,
  print message,
  prompt for key_number
prompt user for new or existing key
repeat
  if new then
    call Generate_New_Number(key_number, new_key)
  else,
    prompt for new_key
    call Check_Applicability(new_key, found)
until found
display new_number
prompt user for acceptance
if acceptance equal true, then
  if existing, then
    prompt for amount on hand
  prompt for amount needed
Module Name: 1.2.3.1.1 Generate New Number
Parameters:
  In: key_number
  Out: new_key_number
  In/Out: none
Module Description:

in Inventory file
find all records with corresponding keyway number
within that selection set
while not done
choose first record
  increment the key number by 2
  check new number with all numbers in
    selection set
  if not duplicated, then found
else choose next record

Module Name: 1.2.3.1.2 Check Applicability
Parameters:
  In: new_key_number
  Out: found
  In/Out: none
Module Description:

in Inventory file
find all records with corresponding keyway number
if number is not present or
  number is found and status is available
then return found
Module Name: 1.2.3.2 Update Files
Parameters:
   In:  new_key_number, amount, order_amount, key_number
       Out: none
       In/Out: none
Module Description:

open Inventory File
retire old key number
find new key number
if it exists, then
   "unretire" it,
else,
   change quantities to amount
   add record
open Location File
find record matching key_number, building, and room
if it exists, then
   change to new_key_number,
else,
   add new record
open Order File
   add new record
Module Name: 1.2.4 Inventory Control
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

display Inventory Control Menu
repeat
   prompt for menu_choice
   if menu_choice Is Change Inventory, then
      call Change Inventory
   else if menu_choice is Order Parts, then
      call Order Parts
   else if menu_choice is Add Item, then
      call Add Item
   else if menu_choice is Print Inventory, then
      call Print Inventory
   else if menu_choice is Delete Item, then
      call Delete Item
   else if menu_choice is quit, then
      do nothing
   else display invalid choice message
until menu_choice is quit
Module Name: 1.2.4.1 Change Inventory
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

  prompt for keyway number
  access keyway file
  find keyway number
  if number exists
    prompt for new_amount
    substitute new_amount for amount
    display changed entry
  else display error message

Module Name: 1.2.4.2 Add Item
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

  while there are more entries
    prompt for keyway number
    prompt for new_amount
    Update_file
Module Name: 1.2.4.2.1 Update File
Parameters:
  In: key_number, new_amount
  Out: none
  In/Out: none
Module Description:

access Keyway File
find keyway number
if number exists
  add new_amount to amount
else request verification of number
  if verified
    then add new record with keyway number
        and new_amount for amount

Module Name: 1.2.4.3 Print Inventory
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

access Keyway File
print all
Module Name: 1.2.4.4 Order Parts
Parameters:
   In: none
   Out: none
In/Out: none
Module Description:

Check_File
Print Order_Parts_File

Module Name: 1.2.4.4.1 Check File
Parameters:
   In: none
   Out: none
In/Out: none
Module Description:

for each record in keyway file
if amount is less than threshold number
then add keyway number to Order_Parts_File
Module Name: 1.2.4.5 Delete Item
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

while there are more items to delete
  prompt for keyway number
  find number in keyway file
  if it exists
    verify number
    delete record
  else print error message

Module Name: 1.3 Maintenance Main
Parameters:
  In: requested
  Out: none
  In/Out: none
Module Description:

call Backup_Info(requested)
call Order_Key
call Update_Key_Status.
Module Name: 1.3.1 Backup Info
Parameters:
  In: requested
  Out: none
  In/Out: none
Module Description:

if not requested, then
  open Last_Save File
  if date last saved is less than sysdate - 7 days, then
    call Save
else,
  display File Restoration Menu
repeat
  prompt user for menu_choice
  if valid menu_choice,
    call Restore(menu_choice)
  else,
    display invalid choice message
until menu_choice is quit.

Module Name: 1.3.1.1 Save
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

save Ind_Record File on tape
save Name File on tape
save Location File on tape
save Inventory File on tape
save Keyway File on tape
save Order Key File on tape.
Module Name: 1.3.1.2 Restore
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

if menu_choice is Ind_Record File, then
   copy Ind_Record File from tape to user area
else, if menu_choice is Name File, then
   copy Name File from tape to user area
else, if menu_choice is Location File, then
   copy Location File from tape to user area
else, if menu_choice is Inventory File, then
   copy Inventory File from tape to user area
else, if menu_choice is Keyway File, then
   copy Keyway File from tape to user area
else, if menu_choice is Order Key File, then
   copy Order Key File from tape to user area
else, if menu_choice is All Files, then
   copy all files from tape to user area.
Module Name: 1.3.2 Order Key
Parameters:
  In: none
  Out: none
  In/Out: none
Module Description:

open Order Date File
if last_order_date does not equal sysdate, then
  set last_order_date to sysdate
open Inventory and Order Key Files
find all quantity_on_hand less than 6 in
  Inventory File
for each record,
  find matching key_number in Order Key File
  if found, then
    if (6 - quantity_on_hand) is greater
        than amount, then
        change amount to
          (6 - quantity_on_hand)
    else,
      do nothing
  else,
    add priority, key_id_number, amount
    record to Order Key File
close Order Key and Inventory Files
close Order Date File.
Module Name: 1.3.3 Update Key Status
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

for each record in Inventory file
   if status is unavailable
      then if date is older than one year
         then change status to available

Module Name: 2 Data Entry Main
Parameters:
   In: none
   Out: none
   In/Out: none
Module Description:

repeat
   call Accept_Name
   call Enter_Info
until user_choice is quit.
Module Name: 2.1 Accept Name
Parameters:
   In: none
   Out: name, SSN
   In/Out: none
Module Description:

repeat
   prompt user for name or quit (CR)
   call Find_SSN(name)
   set social security number to ssn found
until user choice is quit
if social security number is not found, then
   open Dummy_SS File
   set social security number to dummy_ssn
   decrement dummy_ssn by one
   close Dummy_SS File.
Module Name: 2.2 Enter Info
Parameters:
   In: name, SSN
   Out: user_choice
   In/Out: none
Module Description:

if social security number is less than zero, then
   open Name File
   enter name in name field
   enter social security number in ssn field
   close Name File
open Ind_Record File
enter social security number in ssn field
repeat
   prompt for key_number
   prompt for deposit_amount
   if deposit_amount (CR), then
      set deposit_amount to 2
   display key_number and deposit_amount
   ask user if correct
until correct
enter key_number in key_id_number field
enter deposit_amount in deposit field
close Ind_Record File.
File Descriptions

The following files have been determined as being sufficient to store the data required to maintain the key inventory control system. There are six external files used throughout the system. The keyway file is used exclusively in the locksmith operation, the Ind_Record file and Name file are used exclusively in the front office operation, and the remaining files; Inventory file, Order Key file, and Location file are used by both operations.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security Number</td>
<td>integer</td>
<td>9</td>
</tr>
<tr>
<td>Key ID Number</td>
<td>text</td>
<td>10</td>
</tr>
<tr>
<td>Date Out</td>
<td>date</td>
<td>8</td>
</tr>
<tr>
<td>Deposit</td>
<td>integer</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security Number</td>
<td>integer</td>
<td>9</td>
</tr>
<tr>
<td>Name</td>
<td>text</td>
<td>30</td>
</tr>
</tbody>
</table>
### Inventory File:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key ID</td>
<td>text</td>
<td>10</td>
</tr>
<tr>
<td>Type</td>
<td>text</td>
<td>1</td>
</tr>
<tr>
<td>Quantity on Hand</td>
<td>integer</td>
<td>3</td>
</tr>
<tr>
<td>Total Number</td>
<td>integer</td>
<td>3</td>
</tr>
<tr>
<td>Status</td>
<td>text</td>
<td>1</td>
</tr>
<tr>
<td>Date</td>
<td>date</td>
<td>8</td>
</tr>
</tbody>
</table>

### Location File:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key ID</td>
<td>text</td>
<td>10</td>
</tr>
<tr>
<td>Building</td>
<td>text</td>
<td>3</td>
</tr>
<tr>
<td>Lock/Room</td>
<td>text</td>
<td>4</td>
</tr>
</tbody>
</table>

### Keyway File:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyway Number</td>
<td>text</td>
<td>4</td>
</tr>
<tr>
<td>Quantity On Hand</td>
<td>integer</td>
<td>3</td>
</tr>
</tbody>
</table>

### Order Key File:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>text</td>
<td>1</td>
</tr>
<tr>
<td>Key ID</td>
<td>text</td>
<td>10</td>
</tr>
<tr>
<td>Amount</td>
<td>integer</td>
<td>3</td>
</tr>
</tbody>
</table>

Requirements Mapping
A requirements mapping is included here to indicate to the reader and the user that all requirements have been addressed. For each requirement specified in the requirement document, an entry is made and a cross-reference is indicated to the appropriate modules within the design. The purpose of this cross-reference matrix is to (1) establish that all functional requirements (listed in the left hand column) are satisfied by the software design and indicate which modules (listed across the top row) are critical to the implementation of specific requirements.

Requirements Table

The following is a listing of all the requirements specified in the Requirements Document. The reader is referred to the Requirements Document in case of questions, as some of the requirements specified below are not automated and therefore will have no requirements cross-
reference.

1.1 Distribute Information
1.2 Find SSN
1.3 Issue Key
1.3.1 Find Key Number
1.3.2 Update Name
1.3.3 Check Key Quantity
1.3.4 Decrease Key Quantity Available
1.4 Return Key
1.4.1 Flag Record
1.4.2 Check All Names
1.4.3 Delete Files
1.4.4 Increase Key Quantity
1.5 Bookkeeping
1.5.1 Generate Receipt
1.5.2 Generate Deposit
1.5.3 Refund Check
1.5.4 Update Checkbook
1.6 Generate Correspondence
1.6.1 Generate Letters
1.6.2 Generate Reports
1.7 Order Keys
1.8 Backup Information
2.1 Make Key
2.2 Service Locks
2.3 Rekey Lock
2.3.1 Generate Key Number
2.3.2 Determine Lock Number
2.3.3 Pin Lock
2.3.4 Install Lock
2.4 Control Inventory
2.4.1 Store Part
2.4.2 Check Supply
2.4.3 Estimate Need
2.4.4 Order Part
2.5 Recycle Keys
## REQUIREMENTS CROSS REFERENCE

### DESIGN MODULES

<table>
<thead>
<tr>
<th>Requirements</th>
<th>1.1</th>
<th>1.2</th>
<th>1.3</th>
<th>1.3.1</th>
<th>1.3.2</th>
<th>1.3.3</th>
<th>1.3.4</th>
<th>1.4</th>
<th>1.4.1</th>
<th>1.4.2</th>
<th>1.4.3</th>
<th>1.4.4</th>
<th>1.5</th>
<th>1.5.1</th>
<th>1.5.2</th>
<th>1.5.3</th>
<th>1.5.4</th>
<th>1.6</th>
<th>1.6.1</th>
<th>1.6.2</th>
<th>1.7</th>
<th>1.8</th>
<th>1.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Design Specifications

- 1.1: Initial design specifications
- 1.2: Functional requirements
- 1.3: Performance specifications
- 1.3.1, 1.3.2, 1.3.3, 1.3.4: Detailed design modules
- 1.4: Interface design
- 1.4.1: Hardware interfaces
- 1.4.2: Software interfaces
- 1.4.3: Communication interfaces
- 1.4.4: Control interfaces
- 1.5: Security design
- 1.5.1: Access control
- 1.5.2: Encryption
- 1.5.3: Authentication
- 1.5.4: Integrity
- 1.6: Testing design
- 1.6.1: Regression testing
- 1.6.2: Performance testing
- 1.7: Maintenance
- 1.8: Documentation
- 1.9: Configuration management

### Design Considerations

- 2.1: Hardware considerations
- 2.2: Software considerations
- 2.3: Network considerations
- 2.4: Security considerations
- 2.5: Maintenance considerations

### Design Specifications

- 3.0: General specifications
- 4.0: Summary specifications
User Interface

The following state transition diagram shows the various states that the Key Inventory Control system will undergo as various menu options are chosen. This should provide the user with the ability to trace the movements of the working program throughout the user interface.
(FRONT OFFICE MENU)

- ISSUE KEY
- KEY ISSUE
- EXIT
- RETURN KEY
- KEY RETURN
- EXIT
- GENERATE CORRESPONDENCE
- REPORT OR LETTER GENERATION
- EXIT

(INFORMATION ACCESS MENU)

- KEYHOLDER
- INFORMATION
- DISPLAY KEYHOLDER
- EXIT
- KEY NUMBER INFORMATION
- DISPLAY KEY NUMBER
- EXIT
- BUILDING AND ROOM NUMBER INFO
- DISPLAY LOCATION
- EXIT
- INFORMATION ON KEYS HELD
- KEY INVENTORY INFO
- KEY INVENTORY
- DISPLAY INVENTORY
- DAILY CASH BALANCE
- DISPLAY DEPOSIT
- EXIT
- EXIT
- EXIT

FRONT OFFICE MAIN

INFORMATION ACCESS
STATE TRANSITION DIAGRAM
LOCKSMITH MAIN

(LOCKSMITH MAIN MENU)

LOCKSMITH MAIN
- REPLACE KEY
- COMPLETE KEY ORDERS
- COMPLETE KEY ORDERS
- REKEY LOCK
- INVENTORY CONTROL
- EXIT
- EXIT
- EXIT
- EXIT
- EXIT

302
STATE TRANSITION DIAGRAM
INVENTORY CONTROL

(INVENTORY CONTROL MENU)

INVENTORY CONTROL

- CHANGE INVENTORY
- ORDER PARTS
- ADD ITEM
- DELETE ITEM

- PRINT INVENTORY
- ORDER PARTS
- DELETE ITEM

EXIT
Menu Descriptions

The following is a brief overview of the user interface and proposed menu selections.

Front Office Menus

The first menu the Key Inventory Control System user will encounter is the main menu, giving the choices of subprograms that the user may choose to enter. Each of these expects a separate password, however, so indiscriminate access to all portions of the program is not allowed.

***************************************************
*    *
*  MAIN MENU  *
*    *
***************************************************
*    *
* Enter program choice: *
*   A  Front Office Procedures *
*   B  Locksmith Procedures *
*   C  Maintenance Procedures *
*   D  Exit Program *
*    *
***************************************************
If the Front Office Procedure was chosen at the main program level and the password for that section was entered correctly, the next menu the user will encounter is that of the Front Office. It allows the user of front office function choices. These functions may be chosen in any order, and any number of times until the user indicates a desire to exit the front office portion of the program.

********************************************************************************
* *
* FRONT OFFICE MENU *
* *
********************************************************************************
* *
Enter function choice: *
*
A Key Issue *
B Key Return *
C Information Access *
D Report or Letter Generation *
E Exit *
* *
********************************************************************************

If information access was the function chosen from the front office menu, then the Information
Access menu will be displayed. This allows the user to choose the type of information he would like to obtain.

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

INFORMATION ACCESS MENU

* * *

Enter information topic option:

A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

* * *

Selection of the option of Keyholder Information in the Information Access menu will result in the display of the Keyholder Information Menu. This menu will allow the user to choose which information to enter in order to obtain information concerning key holders.
If the user had entered the Report or Letter Generation function at the Front Office Menu, and then indicated (from a single prompt) the desire to generate reports, then the Report Generation Menu would appear. This menu indicates which reports the user may produce.
If, at the Main Menu level, the user had chosen the Maintenance Program option, and entered the password correctly, this would allow access to the file restoration function. This function allows the user to pull previously stored files from magnetic tape to be accessed via interactive 1022 commands. The menu provided for this function allows the user to choose which files he wishes to restore.
Locksmith Menus

The following menu is presented at the onset of the locksmith portion of the program. It controls all the options available at the high level. The user can choose any one of the four options.
The following menu is presented to the user with the choice of 'A' above.

Choose Option:
A List Orders
B Change Orders
C Record Completion
D Print Orders
E Exit
The following menu is displayed following the choice of 'D', (Inventory Control) in the main menu. The choice of 'B' or 'C', does not provide the user with a menu, as there is only one function to perform and no menu is needed.

******************************************************************************
* *
* INVENTORY CONTROL MENU *
* *
******************************************************************************
* *
* Choose Option: *
* A Change Inventory *
* B Order Parts *
* C Add Item *
* D Delete Item *
* E Print Inventory *
* F Exit *
* *
******************************************************************************
Memory Use

The memory usage of this system has been estimated at 400 pages. Needless to say, at this time the exact number of pages required to implement this system cannot be established. Although the amount of space needed for storage of files can be fairly accurately estimated, only a rough estimate of program length can be arrived at. This can be attributed to the inexact art of estimation (especially of lines of code), but also to the inexperience of the developers in the fields of large program development and estimation skills.

Memory usage is composed of two separate categories. These are file storage and program length (lines of code). Currently, the following figures have been calculated for the amount of storage required for the various files needed for information storage. The reader is referred to the section on File Descriptions for a complete
description and field breakdown of all the files listed here.

Ind_Record File length: 31
19,872 records x 31 bytes = 616,032 bytes
616,032 bytes / 2560 bytes/page = 231 pages

Name File length: 39
1,000 records x 39 bytes = 39,000 bytes
39,000 bytes / 2560 bytes/page = 16 pages

Inventory File length: 26
7500 records x 26 bytes = 195,000 bytes
195,000 bytes / 2560 bytes/page = 77 pages

Location File length: 17
5000 records x 17 bytes = 85,000 bytes
85,000 bytes / 2560 bytes/page = 34 pages
Order Key File length: 14
50 records X 14 bytes = 700 bytes
700 bytes / 2560 bytes/page = 1 page

Keyway File length: 7
15 records X 7 bytes = 105 bytes
105 bytes / 2560 bytes/page = 1 page

The total number of pages required for file storage is therefore estimated to be 360 pages. The estimate for program storage is estimated to be 40 pages. This figure was arrived by taking the number of modules in the program (calculated from the hierarchical charts) and multiplying this figure by an average figure of 70 lines of code per module. Obviously, this figure is not, and will not follow throughout the program, but is only an average. Following the concepts of structured design, the smaller the module is, the easier to modify and to understand. We, therefore,
will strive to keep module size as small as possible, however, as this system is highly user interactive and must include error checking, we estimate that the average module size will be larger than non-user-interactive programs. The approximate lines of code necessary is:

\[ 71 \text{ (modules)} \times 70 \text{ (statements/module)} = 4970 \]

The memory use needed by this program will consist of a combination of the amount needed for file storage and that needed for program storage. This figure is estimated to be:

\[ 360 \text{ (files)} + 40 \text{ (program)} = 400 \text{ (total)} \]

Hardware Considerations

The Physical Plant has currently several terminals and printers which will provide them with access to the Dec 2065, upon which the Key Inventory Control system will be developed.
Access to the Dec 2065 is imperative to the development of this system. Currently, space has been guaranteed for the development of the program and the files necessary. However, the amount of space necessary to store all the information within the data files has been calculated to be approximately 360 pages. This amount of space cannot be guaranteed at this time, by the Computer Center.

Performance Considerations

The Dec 2065 was chosen as host computer for several reasons. The Dec 2065 is operated by the University of Montana Computer Center and is the host machine for many of the campus administrative programs. Access has been provided to several of these files, providing not only for less storage space within the files, but for access to additional information, hopefully allowing for greater traceability of keys. Additionally, support of these programs and the Key Inventory Control System is provided by the Computer Center. Also, due
to the amount of information contained within the data base files, accessing information could be somewhat time consuming. Use of the Dec 2065, a mainframe, will provide a satisfactory response time.

Use of structured design principles and documented testing procedures will provide the Physical Plant with a reliable program. This is essential, as the data base will contain information on the whereabouts of the keys and thus, the security of the University.
Development Projections

Following is a projected schedule for the completion of the remaining phases in the proposed project. These dates are approximate, due to both the inexact art of estimation and the difficulty of scheduling review meetings at the end of each phase.

6/26/85 - Detailed Design
6/28/85 - Verification
7/10/85 - Coding and Unit Testing
7/31/85 - Integration Testing
8/9/85 - Final Implementation and User Manual
Maintenance Considerations

Maintainability is the ease with which a software system can be corrected when errors or deficiencies occur, and can be expanded or contracted to satisfy new requirements. This is one of several characteristics sought in a high-quality software system. Maintainability can be thought of as a product of the following characteristics:

testable
understandable
modifiable
portable
reliable
efficient
usable

Of these characteristics, perhaps understandability is the most important, and if a program is understandable it tends to naturally have some of the other characteristics. If a program is not understood it is virtually impossible to maintain
in any sort of efficient or effective manner. Understandable programs have many of the following properties:

modularity
consistency of style
avoidance of obscure code
use of meaningful data and procedure names
structuredness

Use of structured design leads to these properties. Structuredness and modularity are concepts of structured design and direct the systems towards a more easily maintained and modified system. It also provides for a means of generating test cases and, it follows, greater testing and higher reliability.

The reader is referred to the Appendixes where several standardizations have already been specified. This helps to provide the consistency of style mentioned above. As this enhances readability, and understandability, it therefore,
enhances maintainability.
Acknowledgements

At this point we would like to express our appreciation to Mr. Philip Bain of the Registrar's Office and Ms. Linda Brown of the Personnel Office, whose cooperation has been most helpful in establishing the interfaces between the various existing university database programs and the creation of the Physical Plant Key Inventory Control system. Without their support, many of the proposed functions would not be possible to implement.

We would also like to thank those people in the Physical Plant who have donated their time and expertise to help with the development of the Key Inventory Control system. Harry Simon, locksmith, Patty Gibson and Marlice McMahon, administrative aide, have offered invaluable assistance, without which our task would have been much more difficult.
References


Appendix A: Layout of DMD Files

The following file formats are those to be used within the actual program in order to store the necessary data base information. All fields are laid out in the order shown, and these will be used as the 1022 definition files.

IND_RECORD.DMD

SOCIAL_SECURITY_NUMBER SSN INTEGER KEYED 9
KEY_ID_NUMBER KIN TEXT KEYED 10
DATE_OUT DO DATE-OF-ENTRY 8
DEPOSIT DEP INTEGER 4

NAME.DMD

SOCIAL_SECURITY_NUMBER SSN INTEGER KEYED 9
LAST_NAME LNM TEXT 17
FIRST_NAME FNM TEXT 13
INVEN.DMD

KEY_ID_NUMBER KIN TEXT KEYED 10
KEY_TYPE KY TEXT 1
QUANTITY_ON_HAND QON INTEGER 3
TOTAL_NUMBER TN INTEGER 3
STATUS ST TEXT 1
DATE_RETIRED DR DATE 8

LOCATE.DMD

KEY_ID_NUMBER KIN TEXT KEYED 10
BUILDING BLD TEXT 3
ROOM_NUMBER RN TEXT 4

KEYWAY.DMD

KEYWAY_NUMBER KYN TEXT KEYED 4
QUANTITY_ON_HAND QON INTEGER 4
! ORDER.DMD

!! PRIORITY PR TEXT 1
KEY_ID_NUMBER KID TEXT KEYED 10
AMOUNT AMT INTEGER 3
ENTRY_DATE ED DATE OF ENTRY 8
Appendix B: Data Structure Layout

This section describes the layout of internally used files rather than the external ones previously defined. These files are not included in the file relationship diagram because they are used only to store information from one program run to the next. They do not map to any other files, and for the most part have only one field and one entry in that field.

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
! DUMMY_SS.DMD
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
SOCIAL_SECURITY_NUMBER SSN INTEGER KEYED 9

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
! CHECKBOOK.DMD
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
BALANCE BAL INTEGER KEYED 6
ORDER_DATE.DMD

ORDER_DATE OD DATE-OF-ENTRY 8

LAST_SAVE.DMD

DATE_LAST_SAVED DLS DATE OF ENTRY 8

KEY_ORDER_FILE.DMD

KEYWAY_NUMBER KYN INTEGER 4
AMOUNT_QON INTEGER 4

HOLDER.DMD

SOCIAL_SECURITY SSN INTEGER KEYED 9
KEY_NUMBER KIN TEXT 10
APPENDIX C:
FILE RELATIONSHIPS

PERSONNEL

REGISTRAR

INVENTORY

IND_RECORD

NAME

LOCATION

KEYWAY

ORDER KEY
Appendix D: Standard Module Heading Information

The following is a pictorial representation of the module heading information that will be included within the code of the program to identify each module, its parameters, and its function. It is included to make future maintenance work easier in identifying particular parts of the entire program.

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
! Module Name: 
!
! Parameters: 
! In Only: 
! Out Only: 
! In/Out: 
! Coded By: 
! Date Last Modified: 
! Reason Modified: 
! 
! Module Description: 
! (functional description) 
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

The length of the module heading may vary dependent upon the amount of information to be included within the functional description.
Appendix E: Standard Menu Format and Prompt Layout

The following is a pictorial representation of the standard menu layout to be used within the entire key inventory control system. It is standardized so that all menus encountered by the user will be of the same format, and require the same type of user response form one section of the program to another. The width of the menu display will remain constant, however, dependent upon the number of choices available to the user, the length may vary. In addition, a prompt will be included at the end of the menu to remind the user to make a choice, and any additional instructions necessary will be included here.
In addition to the standardized user menu, a standardized form of prompt will be used to obtain information from the user other than that obtained from the menu. Whenever a user must enter information into the system, a prompt indicating the type of information will appear on the screen, and blanks over which the user may type the information. If there are a finite number of responses possible to a question such as Yes/No, the user will be given a list of the possible choices. The format for the user prompt is as follows.

**Instruction/Question (possible answers):**
Appendix F: Sample Letter

The following letter shows the content of letters sent out to personnel and students leaving the University system if they still possess keys. The letter itself will be kept in a separate file so that it will be easily changed should another format be desirable in the future.

Date

Dear Student/University Employee,

It has come to our attention that you will soon be leaving the University system, and that you still possess one or more keys from the Physical Plant. Please return these to the Physical Plant prior to leaving, and your deposit will be refunded. The keys we show that you have are:

Key Number 1
Key Number 2
...
Key Number n

Thank you for your cooperation!

University of Montana Physical Plant
Appendix G: Sample Reports

Several kinds of reports may be generated by the report generation function of the Key Inventory Control System. The format of these reports is shown below. Each sample is a greatly abbreviated representation of the actual report, but may be used to understand the layout of each.

Keyholder Report

<table>
<thead>
<tr>
<th>Keyholder Name</th>
<th>Key Number</th>
<th>Date Issued</th>
<th>Deposit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doe, John E.</td>
<td>3K68179</td>
<td>2/12/85</td>
<td>2.00</td>
</tr>
<tr>
<td></td>
<td>3K65132</td>
<td>3/15/84</td>
<td>2.00</td>
</tr>
<tr>
<td>Smith, Mary L.</td>
<td>3K63128</td>
<td>6/18/75</td>
<td>2.00</td>
</tr>
</tbody>
</table>
Inventory Report

<table>
<thead>
<tr>
<th>Key Number:</th>
<th>Quantity on Hand:</th>
<th>Quantity Out:</th>
<th>Total Number:</th>
</tr>
</thead>
<tbody>
<tr>
<td>3K68179</td>
<td>10</td>
<td>25</td>
<td>35</td>
</tr>
<tr>
<td>2F43289</td>
<td>3</td>
<td>17</td>
<td>20</td>
</tr>
<tr>
<td>3K63219</td>
<td>18</td>
<td>5</td>
<td>23</td>
</tr>
</tbody>
</table>
### Location Report

<table>
<thead>
<tr>
<th>Key Number</th>
<th>Building</th>
<th>Room Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>3K68179</td>
<td>UH</td>
<td>315</td>
</tr>
<tr>
<td>3K47328</td>
<td>UH</td>
<td>22</td>
</tr>
<tr>
<td>2F53728</td>
<td>LA</td>
<td>103</td>
</tr>
</tbody>
</table>

### Key Retirement Report

<table>
<thead>
<tr>
<th>Key Number</th>
<th>Quantity on Hand</th>
<th>Quantity Out</th>
<th>Total Number</th>
<th>Date Retired</th>
</tr>
</thead>
<tbody>
<tr>
<td>3K68179</td>
<td>33</td>
<td>2</td>
<td>35</td>
<td>1/15/85</td>
</tr>
<tr>
<td>2F34287</td>
<td>12</td>
<td>17</td>
<td>29</td>
<td>5/14/85</td>
</tr>
</tbody>
</table>
APPENDIX H:
FILE NAMING CONVENTIONS:
RESTORE

<FILE> → HIST<FILE> → TAP<FILE>

RESTORE: LOADS TAP<FILE> TO TMP<FILE>
Appendix E: Test Run

(RECORDING INITIATED AT 21:38:10)
LINK FROM CS.GRAD.MILEY, TTY 25

TOPS-20 Command processor 5(712)
@1022
7/30/85
System 1022A 116A(436)
* use keys

TOPS-20 Command processor 5(712)
@H$J

**************************************************

                  KEY INVENTORY CONTROL MENU

**************************************************

Enter Program Choice:
*  A Front Office Procedures
*  B Locksmith Procedures
*  C Maintenance Procedures
*  D Exit Program

Enter Choice: c'H$Kx
Invalid Choice

TOPS-20 Command processor 5(712)
@H$J

**************************************************

                  KEY INVENTORY CONTROL MENU

**************************************************

Enter Program Choice:
*  A Front Office Procedures
*  B Locksmith Procedures
*  C Maintenance Procedures
*  D Exit Program

Enter Choice: a

Enter Password:
1234449

TOPS-20 Command processor 5(712)
@H$J
Incorrect Password
Please Re-enter Password
1234567'H$K9

TOPS-20 Command processor 5(712)
@H$J

TOPS-20 Command processor 5(712)
@1:20h MATT, Queen, Request-TN 161, Limit 0:05:001
Enter Program Choice:
- A Front Office Procedures
- B Locksmith Procedures
- C Maintenance Procedures
- D Exit Program

Enter Choice: a

Please enter 9-digit social security number, if not available, simply type a 0 (zero).
SSN: 516745486

SSN: 516745486
Is this correct? (Y/N) y

There is no corresponding record for that SSN.

Please enter issuee's last name: buck

Please enter issuee's first name and middle initial: david f.

LAST NAME: buck
FIRST NAME: david f.
Is this correct? (Y/N) v
Please enter building abbreviation, (maximum 3 characters).
Bldg: lib

Please enter room number, (maximum 4 characters).
Room No: 110

Bldg: lib
Room No: 110

Is this correct? (Y/N) y

Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return:

The following record will be entered into the files:
Name: david f. buck
Bldg & Room: lib 110
Key Number: 3644444
Deposit Amt: $2.00
Is this correct? (Y/N) y
That record has been entered.
Do you have another entry for this individual (Y/N) n

TOPS-20 Command processor 5(712)
@SHSJ

Enter Choice: a

Please enter 9-digit social security number, if not available, simply type a 0 (zero).
SSN: 728418899

Is this correct? (Y/N) y

NAME: GLENDA BARNES

Please enter building abbreviation, (maximum 3 characters).
Bldg: uh

Please enter room number, (maximum 4 characters).
Room No: 118

Bldg: uh
Room No: 118
Is this correct? (Y/N)  n
Please enter building abbreviation, (maximum 3 characters).
Bldg: ma

Please enter room number, (maximum 4 characters).
Room No: 118

Bldg: ma
Room No: 118

Is this correct? (Y/N)  y
You have an invalid building and room combination.
Please check your numbers.
(Type a carriage return <cr> to continue)

Do you have another entry for this individual (Y/N)  y
Please enter building abbreviation, (maximum 3 characters).
Bldg: ma

Please enter room number, (maximum 4 characters).
Room No: 'HSK'HSKoo'HSK'HSK001

Bldg: ma
Room No: 001

Is this correct? (Y/N)  y
Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return: 8800

The following record will be entered into the files:
Name: GLENDA BARNES
Bldg & Room: ma 001
Key Number: 2F246834
Deposit Amt: ***.**

Is this correct? (Y/N)  n
Do you have another entry for this individual (Y/N)  y
Please enter building abbreviation, (maximum 3 characters).
Bldg: ma

Please enter room number, (maximum 4 characters).
Room No: 001

Bldg: ma
Room No: 001

Is this correct? (Y/N)  y
Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return: 800

The following record will be entered into the files:
Name: GLENDA BARNES
Bldg & Room: ma 001
Key Number: 2F246834
Deposit Amt: $800.00
Is this correct? (Y/N) y
That record has been entered.

Do you have another entry for this individual (Y/N) n

TOPS-20 Command processor 5(712)
@$HSJ
*

FRONT OFFICE MENU

Enter Function Choice;
A Key Issue
B Key Return
C Information Access
D Report or Letter Generation
E Exit Program

Enter Choice: c

TOPS-20 Command processor 5(712)
@$HSJ
*

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: a

TOPS-20 Command processor 5(712)
@$HSJ
*

KEYHOLDER INFORMATION MENU

Enter option for which you have information:
A Building and room number
B Key numbers
C Exit
Please enter menu choice. a

Please enter the building abbreviation
(maximum 3 characters):
Bldg: ma

Please enter the room number
(maximum 4 characters):
Room No: oo"H$K"H$KOO1

Bldg: ma
Room No: 001
Is this correct? (Y/N) y
NAME: RHEA ASHMORE
NAME: GERRY BAERTSCH
NAME: GLENDA BARNES

Do you have another request? (Y/N) y

Please enter the building abbreviation
(maximum 3 characters):
Bldg: f

Please enter the room number
(maximum 4 characters):
Room No: 301

Bldg: f
Room No: 301
Is this correct? (Y/N) n

Please enter the building abbreviation
(maximum 3 characters):
Bldg: la

Please enter the room number
(maximum 4 characters):
Room No: 103

Bldg: la
Room No: 103
Is this correct? (Y/N) y
NAME: WILLIAM DERRICK
NAME: JOHN SMITH
NAME: KEN MILLER

Do you have another request? (Y/N) y"H$Kn

TOPS-20 Command processor 5(712)
@SH$J

KEYHOLDER INFORMATION MENU

Enter option for which you have information:
A Building and room number
B Key numbers
C Exit

Please enter menu choice. c
Enter Choice: F
Deposit Total
$4949.00
Do you wish to have the total cleared? (Y/N) Y
Deposit total is now $0.00.
Type a carriage return <cr> to continue

Enter Choice: D
Please enter the individual's SSN. (9 digits).
| or a carriage return <cr> if none |
Please enter the individual's last name.
Last Name: BUCK
Please enter the individual's first name.
First Name: DAVID
LAST NAME: BUCK
FIRST NAME: DAVID
Is this correct? (Y/N) Y
Key Number  Date Out  Deposit Amount
36H444444  7/30/1985  $2.00

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@SH$J

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: E
Please enter the Key Number.
Key Number: 3K123460

KEY NUMBER  QUANTITY ON HAND  TOTAL OUT  TOTAL NUMBER
3K123460  32  8  40

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@SH$J

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: A

TOPS-20 Command processor 5(712)
@SH$J

KEYHOLDER INFORMATION MENU
Enter option for which you have information:
A Building and room number
B Key numbers
C Exit

Please enter menu choice. D
Invalid choice. Please try again.

TOPS-20 Command processor 5(712)
$HSJ

KEYHOLDER INFORMATION MENU

Enter option for which you have information:
A Building and room number
B Key numbers
C Exit

Please enter menu choice. B

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 36H66666

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 36H666666

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 36H444444

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 3K1232$HK460

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 3K123462

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q

Do you have another request? (Y/N) Y

NAME: UTCKT ANDRE
Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 62K123464

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 62K123464?

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 666666666666666666666666666666666

?1022 INPUT ERROR, FIELD 1; PLEASE RETYPE LINE...
555555555

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 3

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q

There is no one who has all those keys issued to them

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 2F246824

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q

NAME: SALLY SMITH
NAME: DAVID ALT
NAME: FRANK ANDERSON
NAME: STEPHEN BALOGH
NAME: BRUCE BARRETT
NAME: ROBERT REAM
NAME: LORETTA EDWARDS
NAME: CHARLES EYER

Do you have another request? (Y/N) Y
KEYHOLDER INFORMATION MENU

Enter option for which you have information:
A Building and room number
B Key numbers
C Exit

Please enter menu choice. C

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: G

FRONT OFFICE MENU

Enter Function Choice:
A Key Issue
B Key Return
C Information Access
D Report or Letter Generation
E Exit Program

Enter Choice: D

The following types of correspondence are available:

Letter
Please enter choice: L

The following types of letters are available:
S: Student
P: Personnel
Q: Quit

Please enter choice: S
NOT IMPLEMENTED YET
IN PRINT LETTERS.

The following types of letters are available:
S: Student
P: Personnel
Q: Quit

Please enter choice: P
IN GEN PERS LETTERS.
NOT IMPLEMENTED YET
IN PRINT LETTERS.

The following types of letters are available:
S: Student
P: Personnel
Q: Quit

Please enter choice: Q
Leaving letter generation

TOPS-20 Command processor 5(712)
@$SHSJ

The following types of correspondence are available:
L: Letter
R: Report
Q: Quit

Please enter choice: R

TOPS-20 Command processor 5(712)
@$SHSJ

******************************************************************************
| REPORT GENERATION MENU |
******************************************************************************

Enter report option:
A Keyholder Report
B Inventory Report
C Location Report
D Key Retirement Report
E Exit

******************************************************************************

Please enter option choice: C
Please choose one of the following:
T: Total Location Report
   (sorted by either keyway number or building number)
I: Individual Report
   (report by individual keyways or building numbers)
Q: Quit

Please enter your choice: I

Would you like a report based upon a Keyway Number
or Building Number? (K/B): B

Please enter building abbreviation for report.
Building Number: MA

Your report has been sent to the line printer
and can be picked up at the computer center
Type a carriage return <cr> to continue:

TOPS-20 Command processor 5(712)
@$H$J

REPORT GENERATION MENU

Enter report option:
A Keyholder Report
B Inventory Report
C Location Report
D Key Retirement Report
E Exit

Please enter option choice: A

Please choose one of the following:
T: Report on Total Keyholders
I: Individual Report
   (report by individual keyways)
Q: Quit

Please enter your choice: I

Please enter keyway number for report.
Keyway Number: 88

Your report has been sent to the line printer
and can be picked up at the computer center
Type a carriage return <cr> to continue:

TOPS-20 Command processor 5(712)
@$H$J
Please enter option choice: B

Please choose one of the following:
  T: Report on Total Inventory
  I: Individual Report
      (report by individual keyways)
  Q: Quit

Please enter your choice: T

Your report has been sent to the line printer and can be picked up at the computer center.
Type a carriage return <cr> to continue:

TOPS-20 Command processor 5(712)
@$H$J

REPORT GENERATION MENU

Enter report option:
  A Keyholder Report
  B Inventory Report
  C Location Report
  D Key Retirement Report
  E Exit

Please enter option choice: E

The following types of correspondence are available:

L: Letter
R: Report
Q: Quit

Please enter choice: Q

TOPS-20 Command processor 5(712)
@$H$J

FRONT OFFICE MENU

Enter Function Choice:
  A Key Issue
  R Key Return
Enter Choice: C

TOPS-20 Command processor 5(712)
@SHSJ

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: F

Deposit Total 5.00

Do you wish to have the total cleared? (Y/N) N

Type a carriage return <cr> to continue

TOPS-20 Command processor 5(712)
@SHSJ

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: A

TOPS-20 Command processor 5(712)
@SHSJ

KEYHOLDER INFORMATION MENU

Enter option for which you have information!
A Building and room number

B Key numbers

C Exit

Please enter menu choice. B

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.

Key Number: Q

There is no one who has all those keys issued to them.

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.

Key Number: 6666666

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.

Key Number: Q

There is no one who has all those keys issued to them.

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.

Key Number: 999999999

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.

Key Number: Q

There is no one who has all those keys issued to them.

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.

Key Number: 62K123464

?1022 INPUT ERROR, FIELD 1; PLEASE RETYPE LINE... Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.

Key Number: 62K123464

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.

Key Number: Q

NAME: HARRY RAY

NAME: JULIETTE CRUMP

NAME: KEN MILLER

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.

Key Number: 36H444666

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
There is no one who has all those keys issued to them.

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@SHSJ

**KEYHOLDER INFORMATION MENU**

Enter option for which you have information:

A Building and room number
B Key numbers
C Exit

Please enter menu choice. C

TOPS-20 Command processor 5(712)
@SHSJ

**INFORMATION ACCESS MENU**

Enter information topic option:

A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: D

Please enter the individual's SSN. ( 9 digits ).
( or a carriage return <cr> if none ) 516545186

That individual does not have any keys issued to him.

Do you have another request? (Y/N) Y

Please enter the individual's SSN. ( 9 digits ).
( or a carriage return <cr> if none ) 516745186

That individual does not have any keys issued to him.

Do you have another request? (Y/N) 251457199

?1022 INPUT ERROR, FIELD 1: PLEASE RETYPE LINE...

Y

Please enter the individual's SSN. ( 9 digits ).
( or a carriage return <cr> if none ) 251457199

NAME: DFT.ORES REN7
Key Number  Date Out  Deposit Amount
3K123458   7/17/1985   $2.00
62K123462   7/17/1985   $2.00
4K3F864292   7/17/1985   $2.00
3K123464   7/17/1985   $2.00

Do you have another request? (Y/N)  N

TOPS-20 Command processor 5(712)
#5656

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice:  E
Please enter the Key Number.
Key Number:  62K123464

KEY NUMBER  QUANTITY ON HAND  TOTAL OUT  TOTAL NUMBER
---------  -------------  ----------  -----------
62K123464     9            4         13

Do you have another request? (Y/N)  B
Please enter the Key Number.
Key Number:  888888888

Invalid key number given.

Do you have another request? (Y/N)  G
Please enter the Key Number.
Key Number:  777777777

Invalid key number given.

Do you have another request? (Y/N)  TYHUJ
?1022 INPUT ERROR, FIELD 1; PLEASE RETYPE LINE...
^
?1022 INPUT ERROR, FIELD 1; PLEASE RETYPE LINE...
Y
Please enter the Key Number.
Key Number:  666666

Invalid key number given.

Do you have another request? (Y/N)  H
Please enter the Key Number.
Key Number:  LKJUYH7
Invalid key number given.

Do you have another request? (Y/N) N
Key Number: 000000000

Invalid key number given.

Do you have another request? (Y/N) K
Key Number: 8UY8YUY7

Invalid key number given.

Do you have another request? (Y/N) F
Key Number: N

Invalid key number given.

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)

Enter Choice: G

TOPS-20 Command processor 5(712)

Enter Choice: A

Please enter 9-digit social security number,
If not available, simply type a 0 (zero).
S S N: n
SSN: 0
Is this correct? (Y/N) Y
Please enter 9-digit social security number, if not available, simply type a 0 (zero).
SSN: 0
SSN: 0
Is this correct? (Y/N) Y
There is no corresponding record for that SSN.
Please enter issuee’s last name: BUCK
Please enter issuee’s first name and middle initial: DAVID F.
LAST NAME: BUCK
FIRST NAME: DAVID F.
Is this correct? (Y/N) Y
Please enter building abbreviation, (maximum 3 characters).
Bldg: MA
Please enter room number, (maximum 4 characters).
Room No: 001
Bldg: MA
Room No: 001
Is this correct? (Y/N) Y
Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return: 896
The following record will be entered into the files:
Name: DAVID F. BUCK
Bldg & Room: MA 001
Key Number: 2F246834
Deposit Amt: $896.00
Is this correct? (Y/N) Y
That record has been entered.
Do you have another entry for this individual (Y/N) N

*S$H$J

******************************************************
*      FRONT OFFICE MENU                              *
******************************************************
*  Enter Function Choice:                          *
*    A  Key Issue                                    *
*    B  Key Return                                   *
*    C  Information Access                          *
*    D  Report or Letter Generation                 *
*
Enter Choice: B

Please enter SSN or a 0 <zero> if not available: 0

SSN: 0
Is this correct? (Y/N) Y

Please enter the individual's last name.
Last Name: MILEY

Please enter individual's first name and middle initial if known.
First Name: MICHELE NONE

Last Name: MILEY
First Name: MICHELE NONE
Is this correct? (Y/N) Y

The matching SSN could not be found.

There is no record for that individual.
This operation is aborted.
(Type carriage return <cr> to continue) .

TOPS-20 Command processor 5(712)
$H$J

FRONT OFFICE MENU

Enter Function Choice:
A Key Issue
B Key Return
C Information Access
D Report or Letter Generation
E Exit Program

Enter Choice: B

Please enter SSN or a 0 <zero> if not available: 516745486

SSN: 516745486
Is this correct? (Y/N) Y

Please enter the individual's last name.
Last Name: BUCK

Please enter individual's first name and middle initial if known.
First Name: DAVID F.

Last Name: BUCK
First Name: DAVID F.
Is this correct? (Y/N) Y
Enter the key number of the returned key.
Key Number: 8

Key Number: 8
Is this correct? (Y/N) N

Enter the key number of the returned key.
Key Number: 8

Key Number: 8
Is this correct? (Y/N) Y
The record corresponding to that individual and
key number could not be located.
(type carriage return <cr> to continue)

Do you have more keys to return for this individual? (Y/N) Y

Enter the key number of the returned key.
Key Number: 62K123464

Key Number: 62K123464
Is this correct? (Y/N) Y
The record corresponding to that individual and
key number could not be located.
(type carriage return <cr> to continue)

Do you have more keys to return for this individual? (Y/N) Y

Enter the key number of the returned key.
Key Number: 0

Key Number: 0
Is this correct? (Y/N) Y
The record corresponding to that individual and
key number could not be located.
(type carriage return <cr> to continue)

Do you have more keys to return for this individual? (Y/N) N

TOPS-20 Command processor 5(712)
8SHSJ
8

******************************************************************************


FRONT OFFICE MENU

******************************************************************************

Enter Function Choice:
A Key Issue
B Key Return
C Information Access
D Report or Letter Generation
E Exit Program

******************************************************************************

Enter Choice: A

Please enter 9-digit social security number,
If not available, simply type a 0 (zero).
SSN: 0

SSN: 0
Is this correct? (Y/N) Y
There is no corresponding record for that SSN.

Please enter issuee's last name: MILEY

Please enter issuee's first name and middle initial: MICHELE NONE

LAST NAME: MILEY
FIRST NAME: MICHELE NONE

Is this correct? (Y/N)  Y

The matching SSN could not be found

Please enter building abbreviation, (maximum 3 characters).
Bldg: MA

Please enter room number, (maximum 4 characters).
Room No: 001

Bldg: MA
Room No: 001

Is this correct? (Y/N)  Y

Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return: 99

The following record will be entered into the files:
Name: MICHELE NONE MILEY
Bldg & Room: MA 001
Key Number: 2F246834
Deposit Amt: $99.00

Is this correct? (Y/N)  Y

That record has been entered.

Do you have another entry for this individual (Y/N)  N

TOPS-20 Command processor 5(712)
@SH$J

************************************************************************************
*                                                                                   *
*                                              FRONT OFFICE MENU                      *
*                                                                                   *
*                                                                                   *
*          Enter Function Choice:                                                     *
*          A Key Issue                                                                  *
*          B Key Return                                                                 *
*          C Information Access                                                        *
*          D Report or Letter Generation                                                *
*          E Exit Program                                                              *
*                                                                                   *
************************************************************************************

Enter Choice: C

TOPS-20 Command processor 5(712)
@SH$J
@
INFORMATION ACCESS MENU

Enter information topic option:
A  Keyholder Information
B  Key Number Information
C  Building and Room Number Information
D  Information on Keys Held
E  Key Inventory Information
F  Daily Cash Balance
G  Exit

Enter Choice: A

TOPS-20 Command processor 5(712)
@SHSJ

KEYHOLDER INFORMATION MENU

Enter option for which you have information:
A  Building and room number
B  Key numbers
C  Exit

Please enter menu choice. A

Please enter the building abbreviation (maximum 3 characters):
Bldg: MA

Please enter the room number (maximum 4 characters):
Room No: 001

Bldg: MA
Room No: 001
Is this correct? (Y/N) Y

NAME: RHEA ASHMORE
NAME: GERRY BAERTSCH
NAME: GLENDA BARNES
NAME: MICHELE NONE MILEY

Do you have another request? (Y/N) N
Please enter menu choice. B

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 62K123464

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q
NAME: BARBARA BAIN
NAME: HARRY RAY
NAME: JULIETTE CRUMP
NAME: KEN MILLER
Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@SH$J

********************************************************************

KEYHOLDER INFORMATION MENU

********************************************************************

Enter option for which you have information:
* A Building and room number
* B Key numbers
* C Exit

********************************************************************

Please enter menu choice. C

TOPS-20 Command processor 5(712)
@SH$J

********************************************************************

INFORMATION ACCESS MENU

********************************************************************

Enter information topic option:
* A Keyholder Information
* B Key Number Information
* C Building and Room Number Information
* D Information on Keys Held
* E Key Inventory Information
* F Daily Cash Balance
* G Exit

********************************************************************

Enter Choice: A

TOPS-20 Command processor 5(712)
@SH$J

********************************************************************

KEYHOLDER INFORMATION MENU

********************************************************************
Enter option for which you have information:
A Building and room number
B Key numbers
C Exit

Please enter menu choice. C

TOPS-20 Command processor 5(712)
@SHSJ

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information ★
B Key Number Information ★
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information ★
F Daily Cash Balance ★
G Exit

Enter Choice: E
Please enter the Key Number.
Key Number:

Invalid key number given.

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@SHSJ

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: F
Deposit Total
$995.00

Do you wish to have the total cleared? (Y/N) N
Type a carriage return <cr> to continue

TOPS-20 Command processor 5(712)
@S$SHJ

*******************************************************

INFORMATION ACCESS MENU

*******************************************************

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: G

TOPS-20 Command processor 5(712)
@S$SHJ

*******************************************************

FRONT OFFICE MENU

*******************************************************

Enter Function Choice:
A Key Issue
B Key Return
C Information Access
D Report or Letter Generation
E Exit Program

Enter Choice: A

Please enter 9-digit social security number.
If not available, simply type a 0 (zero).
SSN: 516745486

SSN: 516745486
Is this correct? (Y/N) Y

There is no corresponding record for that SSN.

Please enter issuee's last name: BUCK

Please enter issuee's first name and middle initial: DAVID F.

LAST NAME: BUCK
FIRST NAME: DAVID F.

Is this correct? (Y/N) Y

Please enter building abbreviation, (maximum 3 characters).
Abbrev: MA 001
Please enter room number, (maximum 4 characters).
Room No: 001
Bldg: MA
Room No: 001
Is this correct? (Y/N) Y

Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return: 8

The following record will be entered into the files:
Name: DAVID F. BUCK
Bldg & Room: MA 001
Key Number: 2F246814
Deposit Amt: $8.00
Is this correct? (Y/N) Y

That record has been entered.

Do you have another entry for this individual (Y/N) Y

Please enter building abbreviation, (maximum 3 characters).
Bldg: LIB

Please enter room number, (maximum 4 characters).
Room No: 114
Bldg: LIB
Room No: 114
Is this correct? (Y/N) Y

Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return: 4

The following record will be entered into the files:
Name: DAVID F. BUCK
Bldg & Room: LIB 114
Key Number: 36H666666
Deposit Amt: $4.00
Is this correct? (Y/N) Y

That record has been entered.

Do you have another entry for this individual (Y/N) N
Enter Choice: C

TOPS-20 Command processor 5(712)
$0SHSJ

INFORMATION ACCESS MENU

Enter information topic option:

A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: B

Please enter the building abbreviation (maximum 3 characters)
Bldg: LIB

Please enter the room number (maximum 4 characters)
Room No: 114

LIB 114 36H666666

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
$0SHSJ

INFORMATION ACCESS MENU

Enter information topic option:

A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: D

Please enter the individual's SSN (9 digits).
(or a carriage return <cr> if none) 516745486
<table>
<thead>
<tr>
<th>Key Number</th>
<th>Date Out</th>
<th>Deposit Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>36H444444</td>
<td>7/30/1985</td>
<td>$2.00</td>
</tr>
<tr>
<td>2F246834</td>
<td>7/30/1985</td>
<td>$896.00</td>
</tr>
<tr>
<td>2F246834</td>
<td>7/30/1985</td>
<td>$8.00</td>
</tr>
<tr>
<td>36H666666</td>
<td>7/30/1985</td>
<td>$4.00</td>
</tr>
</tbody>
</table>

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
#$HSJ
@

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: F

Deposit Total
$1007.00

Do you wish to have the total cleared? (Y/N) N

Type a carriage return <cr> to continue

TOPS-20 Command processor 5(712)
#$HSJ
@

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information

Enter Choice: A

TOPS-20 Command processor 5(712)
#$HSJ
@

KEYHOLDER INFORMATION MENU
Please enter menu choice. A

Enter the building abbreviation
(maximum 3 characters):
Bldg: LIB

Please enter the room number
(maximum 4 characters):
Room No: 114

Bldg: LIB
Room No: 114
Is this correct? (Y/N) Y
NAME: SHARON BARRETT
NAME: HARRY RAY
NAME: GLEN READ
NAME: EVAN DENNY
NAME: SURESH VADHVA

Do you have another request? (Y/N) K

Please enter the building abbreviation
(maximum 3 characters):
Bldg: MA

Please enter the room number
(maximum 4 characters):
Room No: 001

Bldg: MA
Room No: 001
Is this correct? (Y/N) Y
NAME: RHEA ASHMORE
NAME: GERRY BAERTSCH
NAME: GLENDA BARNES
NAME: MICHELE NONE MILEY

Do you have another request? (Y/N) M

Please enter the building abbreviation
(maximum 3 characters):
Bldg: K

Please enter the room number
(maximum 4 characters):
Room No: 99

Bldg: K
Room No: 99
Is this correct? (Y/N) N

Please enter the building abbreviation
(maximum 3 characters):
R1ñq: ☐
Please enter the room number
(maximum 4 characters):
Room No: ;
Bldg: 0
Room No: ;
Is this correct? [Y/N] Y
You have an invalid building, and room number
Do you have another request? [Y/N] N

TOPS-20 Command processor 5(712)
@$H$J
*
* KEYHOLDER INFORMATION MENU *
* *
* Enter option for which you have information: *
* A Building and room number *
* B Key numbers *
* C Exit *
* *

Please enter menu choice. B

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: 2F246834

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: 36H666666

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: 36H444444

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: Q
Do you have another request? [Y/N] N

TOPS-20 Command processor 5(712)
@$H$J
*
* KEYHOLDER INFORMATION MENU *
* *
* Enter option for which you have information: *
* A Building and room number *
* B Key numbers *
* C Exit *
Please enter menu choice. B

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: 36H666666

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: Q
NAME: SHARON BARRETT
NAME: HARRY RAY
NAME: GLEN READ
NAME: EVAN DENNY
NAME: SURESH VADHVA

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: 2F246834

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: Q
NAME: RHEA ASHMORE
NAME: GERRY BAERTSCH
NAME: GLENDA BARNES
NAME: MICHELE NONE MILEY

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: 36H444444

Please enter a key number or a 'q' to
indicate no more keys. If no names appear
then no one has been issued that group of keys.
Key Number: Q
NAME: SALLY SMITH
NAME: HOWARD JONES
NAME: DAVID ALT
NAME: AARON ANDREASON
NAME: GERRY BAERTSCH
NAME: BARBARA BAIN
NAME: GREGORY BARRETT
NAME: JOAN RANNEY
NAME: JULIA RAY
NAME: HARRY RAY
NAME: MARLYN EGGE
NAME: IRENE EVERS
NAME: BETTINA ESCUDERO
NAME: JULIETTE CRUMP
NAME: ANN SMITH
NAME: SURESH VADHVA

Do you have another request? (Y/N) N

TOPS-20 Command processor 5/7/72
Please enter menu choice. C

TOPS-20 Command processor 5(712)

Please enter menu choice. C

Enter information topic option:

A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: D

Please enter the individual's SSN. (9 digits).
( or a carriage return <cr> if none ) 516745486

Key Number    Date Out   Deposit Amount
--------------    ----------   -----------
36H444444        7/30/1985   $2.00
2F246834         7/30/1985   $896.00
2F246834         7/30/1985   $8.00
36H666666        7/30/1985   $4.00

Do you have another request? (Y/N) H

Please enter the individual's SSN. (9 digits).
( or a carriage return <cr> if none ) 777888999

That individual does not have any keys issued to him.

Do you have another request? (Y/N) F

Please enter the individual's SSN. (9 digits).
( or a carriage return <cr> if none ) MMMMMMMMMM

?1022 INPUT ERROR. FIELD 1: PLEASE RETYPE LINE...

?1022 INPUT ERROR. FIELD 1: PLEASE RETYPE LINE...
That individual does not have any keys issued to him.

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@SHSJ

--------------------------------------------------------------------------------

INFORMATION ACCESS MENU

--------------------------------------------------------------------------------

Enter information topic option:
* A Keyholder Information
* B Key Number Information
* C Building and Room Number Information
* D Information on Keys Held
* E Key Inventory Information
* F Daily Cash Balance
* G Exit

Enter Choice: A

TOPS-20 Command processor 5(712)
@SHSJ

--------------------------------------------------------------------------------

KEYHOLDER INFORMATION MENU

--------------------------------------------------------------------------------

Enter option for which you have information:
* A Building and room number
* B Key numbers
* C Exit

Please enter menu choice. B

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q

There is no one who has all those keys issued to them

Do you have another request? (Y/N) Y

TOPS-20 Command processor 5(712)
@SHSJ

--------------------------------------------------------------------------------

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 2F246834

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q
NAME: RHEA ASHMORE
NAME: GERRY BAERTSCH
NAME: GUINNA RARNFS
NAME: MICHELE NONE MILEY

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@SHSJ
@

KEYHOLDER INFORMATION MENU

Enter option for which you have information:
A Building and room number
B Key numbers
C Exit

Please enter menu choice. C

TOPS-20 Command processor 5(712)
@SHSJ
@

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: G

TOPS-20 Command processor 5(712)
@SHSJ
@

FRONT OFFICE MENU

Enter Function Choice:
A Key Issue
B Key Return
C Information Access
D Report or Letter Generation
E Exit Program

Enter Choice: B

Please enter SSN or a 0 <zero> if not available: 516745486
SSN: 516745486
Is this correct? (Y/N)  Y

Please enter the individual's last name.
Last Name: BUCK

Please enter individual's first name
and middle initial if known.
First Name: DAVID F.

Last Name: BUCK
First Name: DAVID F.
Is this correct? (Y/N) Y

buck    david f.    516745486
BUCK    DAVID F.    516745486
BUCK    DAVID F.    516745486

With the information given the correct
individual cannot be determined. All the
above records apply. Do you have additional
information to distinguish between entries (Y/N) Y

Enter the corresponding SSN. 516745486
SSN: 516745486
Is this correct? (Y/N): Y

Enter the key number of the returned key.
Key Number: 2F246834
Key Number: 2F246834
Is this correct? (Y/N) Y

The following record will be deleted from the files:
Name: DAVID F.    BUCK
Key Number: 2F246834
Bldg: MA
Room Num: 001
Deposit Amount: $896.00
Deposit Amount: $8.00
Is this correct? (Y/N) Y
The record has been deleted.

Do you have more keys to return for this individual? (Y/N) N

TOPS-20 Command processor 5(712)
@SHSJ

***************************************************************************
*                           FRONT OFFICE MENU                            *
***************************************************************************
*   Enter Function Choice:                                          *
*   A  Key Issue                                                   *
*   B  Key Return                                                  *
*   C  Information Access                                         *
*   D  Report or Letter Generation                                *
*   E  Exit Program                                               *
***************************************************************************
Enter Choice:  A

Please enter 9-digit social security number, If not available, simply type a 0 (zero).
SSN:  0

SSN:  0
Is this correct? (Y/N)  Y

There is no corresponding record for that SSN.

Please enter issuee's last name:  BUCK

Please enter issuee's first name and middle initial:  DAVID F.
LAST NAME:  BUCK
FIRST NAME:  DAVID F.
Is this correct? (Y/N)  Y

buck  david f.  516745486
BUCK  DAVID F.  516745486
BUCK  DAVID F.  516745486

With the information given the correct individual cannot be determined. All the above records apply. Do you have additional information to distinguish between entries (Y/N)  N

Please enter building abbreviation, (maximum 3 characters).
Bldg:  MA

Please enter room number, (maximum 4 characters).
Room No:  001

Bldg:  MA
Room No:  001
Is this correct? (Y/N)  Y

Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return:  666

The following record will be entered into the files:
Name:  DAVID F.  BUCK
Bldg & Room:  MA  001
Key Number:  2P246834
Deposit Amt:  $666.00
Is this correct? (Y/N)  Y

That record has been entered.

Do you have another entry for this individual (Y/N)  Y?

?1022 INPUT ERROR, FIELD 1; PLEASE RETYPE LINE...
Y

Please enter building abbreviation, (maximum 3 characters).
Bldg:  LTR
Please enter room number, (maximum 4 characters).
Room No.: 110
Bldg: LIB
Room No: 110
Is this correct? (Y/N) Y

Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return: 2

The following record will be entered into the files:
Name: DAVID F. BUCK
Bldg & Room: LIB 110
Key Number: 36H444444
Deposit Amt: $2.00
Is this correct? (Y/N) Y

That record has been entered.
Do you have another entry for this individual (Y/N) N

TOPS-20 Command processor 5(712)
@$H$J

**********************************************
FRONT OFFICE MENU
**********************************************
Enter Function Choice:
* A Key Issue
* B Key Return
* C Information Access
* D Report or Letter Generation
* E Exit Program

Enter Choice: E

TOPS-20 Command processor 5(712)
@$H$J

**********************************************
KEY INVENTORY CONTROL MENU
**********************************************
Enter Program Choice:
* A Front Office Procedures
* B Locksmith Procedures
* C Maintenance Procedures
* D Exit Program

Enter Choice: B

Enter Password: 7146479
**LOCKSMITH MENU**

Choose Option:
* A Complete Key Orders
* B Replace Key
* C Rekey Lock
* D Inventory Control
* E Exit

Enter choice: D

**INVENTORY CONTROL MENU**

Choose Option:
* A Change Inventory
* B Order Parts
* C Add Item
* D Delete Item
* E Print Inventory
* F Exit

Enter choice: E

<table>
<thead>
<tr>
<th>Keyway Number</th>
<th>Quantity on Hand</th>
</tr>
</thead>
<tbody>
<tr>
<td>3K</td>
<td>332</td>
</tr>
<tr>
<td>2F</td>
<td>95</td>
</tr>
<tr>
<td>3GH</td>
<td>89</td>
</tr>
<tr>
<td>62K</td>
<td>206</td>
</tr>
<tr>
<td>4K3F</td>
<td>73</td>
</tr>
</tbody>
</table>

Type carriage return <cr> to continue:
Choose Option:
A Change Inventory
B Order Parts
C Add Item
D Delete Item
E Print Inventory
F Exit

Enter choice: F

TOPS-20 Command processor 5(712)
@$HSJ

LOCKSMITH MENU

Choose Option:
A Complete Key Orders
B Replace Key
C Rekey Lock
D Inventory Control
E Exit

Enter choice: E

TOPS-20 Command processor 5(712)
@$HSJ

KEY INVENTORY CONTROL MENU

Enter Program Choice:
A Front Office Procedures
B Locksmith Procedures
C Maintenance Procedures
D Exit Program

Enter Choice: C

Enter Password:
999000

TOPS-20 Command processor 5(712)
@$HSJ
@ Incorrect Password
Please Re-enter Password
3456789

TOPS-20 Command processor 5(712)
@$HSJ
@
FILE RESTORATION MENU

Enter file to be restored:
A Individual Record File
B Name File
C Inventory File
D Location File
E Keyway File
F Order File
G All Files
H Exit

Please enter the file to be restored: B

TOPS-20 Command processor 5(712)
@ Unrecognized command - File not found - "NAME.CTL"
@A Unrecognized command - Ambiguous - "A"
@USE KEYS
Unrecognized command - Does not match switch or keyword - "USE"
@1 "HSK022"
7/30/85
System 1022A 116A(436)
* USE KEYS

TOPS-20 Command processor 5(712)
@$H$J

KEY INVENTORY CONTROL MENU

Enter Program Choice:
A Front Office Procedures
B Locksmith Procedures
C Maintenance Procedures
D Exit Program

Enter Choice: A

Enter Password: 1234567

TOPS-20 Command processor 5(712)
@$H$J

TOPS-20 Command processor 5(712)
@$H$J

FRONT OFFICE MENU
Enter Function Choice:
A Key Issue
B Key Return
C Information Access
D Report or Letter Generation
E Exit Program

Enter Choice: E

TOPS-20 Command processor 5(712)
@SHSJ

Enter Program Choice:
A Front Office Procedures
B Locksmith Procedures
C Maintenance Procedures
D Exit Program

Enter Choice: A

Enter Password:
1234567

TOPS-20 Command processor 5(712)
@SHSJ

Enter Function Choice:
A Key Issue
B Key Return
C Information Access
D Report or Letter Generation
E Exit Program

Enter Choice: C

TOPS-20 Command processor 5(712)
@SHSJ

Enter information topic option:
Enter Choice: B

Please enter the building abbreviation
(maximum 3 characters)
Bldg: LIB

Please enter the room number
(maximum 4 characters)
Room No: 110

LIB 110 36H444444

Do you have another request? (Y/N) Y

Please enter the building abbreviation
(maximum 3 characters)
Bldg: MA

Please enter the room number
(maximum 4 characters)
Room No: 001

MA 001 2F246834

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@$H$J
@

***********************************************************************

INFORMATION ACCESS MENU

***********************************************************************

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

***********************************************************************

Enter Choice: B

Please enter the building abbreviation
(maximum 3 characters)
Bldg: LIB

Please enter the room number
(maximum 4 characters)
Room No: 110

LIB 110 36H444444
Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@SH$J
@

******************************************************************************

INFORMATION ACCESS MENU

******************************************************************************

Enter information topic option:

A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

******************************************************************************

Enter Choice: A

TOPS-20 Command processor 5(712)
@SH$J
@

******************************************************************************

KEYHOLDER INFORMATION MENU

******************************************************************************

Enter option for which you have information:

A Building and room number
B Key numbers
C Exit

******************************************************************************

Please enter menu choice. B

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 2F246834

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q

NAME: RHEA ASHMORE
NAME: GERRY BAERTSCH
NAME: GLENDAS BARNES
NAME: MICHELE NONE MILEY

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 36H666666

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number:  
NAME:  SHARON BARRETT
NAME:  HARRY RAY
NAME:  GLEN READ
NAME:  EVAN DENNY
NAME:  SURESH VADHVA

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 36H444444

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q
NAME:  SALLY SMITH
NAME:  HOWARD JONES
NAME:  DAVID ALT
NAME:  AARON ANDREASON
NAME:  GERRY BAERTSCH
NAME:  BARBARA BAIN
NAME:  GREGORY BARRETT
NAME:  JOAN RADANDT
NAME:  JULIA RADTKE
NAME:  JAMES RANNEY
NAME:  HARRY RAY
NAME:  MARLYN EGGE
NAME:  IRENE EVERS
NAME:  BETTINA ESCUDERO
NAME:  JULIETTE CRUMP
NAME:  ANN SMITH
NAME:  SURESH VADHVA

Do you have another request? (Y/N) Y

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: 2F246834

Please enter a key number or a 'q' to indicate no more keys. If no names appear then no one has been issued that group of keys.
Key Number: Q
NAME:  RHEA ASHMORE
NAME:  GERRY BAERTSCH
NAME:  GLENDA BARNES
NAME:  MICHELE NONE MILEY

Do you have another request? (Y/N) N

TOPS-20 Command processor 5(712)
@SHSJ

******************************************************************************
* KEYHOLDER INFORMATION MENU
******************************************************************************
* Enter option for which you have information:
*   A Building and room number
*   B Key numbers
Please enter menu choice. C

TOPS-20 Command processor 5(712)

INFORMATION ACCESS MENU

Enter information topic option:
A Keyholder Information
B Key Number Information
C Building and Room Number Information
D Information on Keys Held
E Key Inventory Information
F Daily Cash Balance
G Exit

Enter Choice: G

TOPS-20 Command processor 5(712)

FRONT OFFICE MENU

Enter Function Choice:
A Key Issue
B Key Return
C Information Access
D Report or Letter Generation
E Exit Program

Enter Choice: E

TOPS-20 Command processor 5(712)

KEY INVENTORY CONTROL MENU

Enter Program Choice:
A Front Office Procedures
B Locksmith Procedures
C Maintenance Procedures
D Exit Program

Enter Choice: C
Enter Password:
3456789

TOPS-20 Command processor 5(712)
@SH$J
@

TOPS-20 Command processor 5(712)
@SH$J
@

FILE RESTORATION MENU

Enter file to be restored:
* A Individual Record File
* B Name File
* C Inventory File
* D Location File
* E Keyway File
* F Order File
* G All Files
* H Exit

Please enter the file to be restored: B

TOPS-20 Command processor 5(712)
@
?Unrecognized command - File not found - "NAME.CTL"
@BUCK, D
?Unrecognized command - Does not match switch or keyword - "BUCK"
@POP

TOPS-20 Command processor 5(712)
@SH$J
@

FILE RESTORATION MENU

Enter file to be restored:
* A Individual Record File
* B Name File
* C Inventory File
* D Location File
* E Keyway File
* F Order File
* G All Files
* H Exit

Please enter the file to be restored: BUCK, DAVID F.
71022 INPUT ERROR, FIELD 1; PLEASE RETYPE LINE...
BUCK
71022 INPUT ERROR, FIELD 1; PLEASE RETYPE LINE...
C
Unrecognized command - File not found - "INVENTORY.CTL"

SYSTEM 1022A 116A(436)
* EXIT

FILE RESTORATION MENU

Enter file to be restored:
A Individual Record File
B Name File
C Inventory File
D Location File
E Keyway File
F Order File
G All Files
H Exit

Please enter the file to be restored: H

KEY INVENTORY CONTROL MENU

Enter Program Choice:
A Front Office Procedures
B Locksmith Procedures
C Maintenance Procedures
D Exit Program

Enter Choice: D

* POP
? (CS24) Invalid command
* EXIT

EXIT
* POP

FILE RESTORATION MENU

Enter file to be restored:
A Individual Record File
B Name File
C Inventory File
D Location File
E Keyway File
F Order File
G All Files
H Exit

Please enter the file to be restored: H

KEY INVENTORY CONTROL MENU

Enter Program Choice:
A Front Office Procedures
B Locksmith Procedures
C Maintenance Procedures
D Exit Program

Enter Choice: D

* POP
? (CS24) Invalid command
* EXIT

EXIT
* POP
Please enter the file to be restored: G

TOPS-20 Command processor 5(712)
@ Unrecognized command - File not found - "RESTORE.CTL"
@ POP

TOPS-20 Command processor 5(712)
@$H$J
@

Please enter the file to be restored: H

TOPS-20 Command processor 5(712)
@$H$J
@

Enter Program Choice:
A Front Office Procedures
B Locksmith Procedures
C Maintenance Procedures
D Exit Program

Enter Choice: D
EXIT
@CONN <CS. GRAD. ROBMICH>
?No such directory or structure not mounted
@POP

[RECORDING TERMINATED AT 23:57:54]
@
Test Files
<table>
<thead>
<tr>
<th>SECURITY</th>
<th>KEY</th>
<th>DATE</th>
<th>DEPOSIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>3K123460</td>
<td>3K123460</td>
<td>7/18/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>3K123462</td>
<td>3K123462</td>
<td>7/17/1985</td>
<td>200</td>
</tr>
<tr>
<td>Date</td>
<td>Code</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123458</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>4K3F864290</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>4K3F864288</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>4K3F864294</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246824</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>2F246826</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>3K123456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1985-07-17</td>
<td>62K123462</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KEY NUMBER</td>
<td>TYPE</td>
<td>HAND</td>
<td>QUANTITY</td>
</tr>
<tr>
<td>------------</td>
<td>------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>C 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2F246824</td>
<td>C</td>
<td>10</td>
<td>23</td>
</tr>
<tr>
<td>2F246826</td>
<td>C</td>
<td>12</td>
<td>24</td>
</tr>
<tr>
<td>2F246828</td>
<td>C</td>
<td>89</td>
<td>90</td>
</tr>
<tr>
<td>2F246830</td>
<td>C</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>2F246832</td>
<td>C</td>
<td>20</td>
<td>25</td>
</tr>
<tr>
<td>2F246834</td>
<td>M</td>
<td>15</td>
<td>17</td>
</tr>
<tr>
<td>2F246836</td>
<td>C</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>36H444444</td>
<td>M</td>
<td>15</td>
<td>33</td>
</tr>
<tr>
<td>36H446666</td>
<td>M</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>36H666666</td>
<td>M</td>
<td>9</td>
<td>15</td>
</tr>
<tr>
<td>36H888888</td>
<td>M</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>3K123456</td>
<td>C</td>
<td>16</td>
<td>44</td>
</tr>
<tr>
<td>3K123458</td>
<td>C</td>
<td>4</td>
<td>27</td>
</tr>
<tr>
<td>3K123460</td>
<td>C</td>
<td>31</td>
<td>40</td>
</tr>
<tr>
<td>3K123462</td>
<td>C</td>
<td>15</td>
<td>25</td>
</tr>
<tr>
<td>3K123462</td>
<td>C</td>
<td>26</td>
<td>27</td>
</tr>
<tr>
<td>3K123464</td>
<td>C</td>
<td>14</td>
<td>20</td>
</tr>
<tr>
<td>3K222222</td>
<td>M</td>
<td>5</td>
<td>14</td>
</tr>
<tr>
<td>3K654322</td>
<td>C</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>3K654324</td>
<td>C</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>3K654326</td>
<td>C</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>3K654328</td>
<td>C</td>
<td>12</td>
<td>24</td>
</tr>
<tr>
<td>4K3F864286</td>
<td>M</td>
<td>16</td>
<td>18</td>
</tr>
<tr>
<td>4K3F864288</td>
<td>C</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>4K3F864290</td>
<td>C</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>4K3F864292</td>
<td>M</td>
<td>12</td>
<td>20</td>
</tr>
<tr>
<td>4K3P64294</td>
<td>C</td>
<td>19</td>
<td>22</td>
</tr>
<tr>
<td>62K123456</td>
<td>C</td>
<td>10</td>
<td>25</td>
</tr>
<tr>
<td>62K123458</td>
<td>C</td>
<td>7</td>
<td>32</td>
</tr>
<tr>
<td>62K123460</td>
<td>C</td>
<td>16</td>
<td>30</td>
</tr>
<tr>
<td>62K123462</td>
<td>C</td>
<td>16</td>
<td>35</td>
</tr>
<tr>
<td>62K123464</td>
<td>C</td>
<td>8</td>
<td>13</td>
</tr>
</tbody>
</table>
LOCATE.FIL.3

KEY    BLDG   ROOM
NUMBER  NMBR  NMBR
        -----   ----

LA      103
2P246824 F 206
2P246826 F 301
2P246828 MA 305
2P246830 MA 304
2P246832 MA 214
2P246834 MA 001
36H444444 LIB 110
36H666666 LIB 114
36H888888 LIB 118
3K123456 UH 315
3K123458 UH 214
3K123460 UH 22
3K123462 UH 2A
3K222222 UH UH
3K654324 LA 11
3K654326 LA 203
3K654328 LA 304
4K3F864286 UC CC
4K3F864288 UC WRC
4K3F864290 UC OPC
4K3F864292 UC BS
62K123456 JRH 201
62K123458 JRH 203
62K123460 SS 340
62K123462 SS 352

NAME.FIL.3

SOCIAL SECURITY LAST NAME FIRST NAME
NUMBER NAME
--------- ------------------
-123456799 SMITH JOHN
-123456796 VADHVA SURESH
-123456795 ANDERSON ROBIN
-123456794 FAUNTLEROY ROBIN
-123456793 MILLER KEN
-123456792 ANDERSON JOE
-123456791 SMITH MARY
-123456790 SMITH ANN
-123456789 SMITH JOHN
796349876 JACKSON ROBINSON

PSMAST.FIL.3
<table>
<thead>
<tr>
<th>SOCIAL SECURITY NUMBER</th>
<th>FIRST NAME</th>
<th>LAST NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>123456789</td>
<td>FRED</td>
<td>SMITH</td>
</tr>
<tr>
<td>251450699</td>
<td>LEROY</td>
<td>ANDERSON</td>
</tr>
<tr>
<td>251515999</td>
<td>ROBERT</td>
<td>BANAUGH</td>
</tr>
<tr>
<td>273229199</td>
<td>STEPHANIE</td>
<td>ANDERSEN</td>
</tr>
<tr>
<td>273294699</td>
<td>RHEA</td>
<td>ASHMORE</td>
</tr>
<tr>
<td>273296099</td>
<td>AARON</td>
<td>ANDREASON</td>
</tr>
<tr>
<td>321456789</td>
<td>JAMES</td>
<td>SMITH</td>
</tr>
<tr>
<td>542019599</td>
<td>CHARLOTTE</td>
<td>BARBER</td>
</tr>
<tr>
<td>542256399</td>
<td>BRUCE</td>
<td>BARRETT</td>
</tr>
<tr>
<td>543564599</td>
<td>BARBARA</td>
<td>BAIN</td>
</tr>
<tr>
<td>543617799</td>
<td>FRANK</td>
<td>ANDERSON</td>
</tr>
<tr>
<td>543781899</td>
<td>ELDON</td>
<td>BAKER</td>
</tr>
<tr>
<td>543781999</td>
<td>DAN</td>
<td>BALLAS</td>
</tr>
<tr>
<td>543888999</td>
<td>DAVID</td>
<td>ALT</td>
</tr>
<tr>
<td>549260399</td>
<td>GREGORY</td>
<td>BARRETT</td>
</tr>
<tr>
<td>549597999</td>
<td>WILLIAM</td>
<td>BALLARD</td>
</tr>
<tr>
<td>678123987</td>
<td>HOWARD</td>
<td>JONES</td>
</tr>
<tr>
<td>678456987</td>
<td>LINDA</td>
<td>JONES</td>
</tr>
<tr>
<td>678901234</td>
<td>SALLY</td>
<td>SMITH</td>
</tr>
<tr>
<td>721012999</td>
<td>SUSAN</td>
<td>BARR</td>
</tr>
<tr>
<td>721115699</td>
<td>GERRY</td>
<td>BAERTSC</td>
</tr>
<tr>
<td>721263899</td>
<td>STEPHEN</td>
<td>BALOGH</td>
</tr>
<tr>
<td>721369599</td>
<td>SHARON</td>
<td>BARRETT</td>
</tr>
<tr>
<td>721369899</td>
<td>BETSY</td>
<td>BACH</td>
</tr>
<tr>
<td>728194599</td>
<td>ROBERT</td>
<td>ANDERSON</td>
</tr>
<tr>
<td>728329599</td>
<td>MARLENE</td>
<td>BACKMAN</td>
</tr>
<tr>
<td>728418899</td>
<td>GLENDRA</td>
<td>BARNES</td>
</tr>
<tr>
<td>728592699</td>
<td>PHILIP</td>
<td>BAIN</td>
</tr>
<tr>
<td>728620899</td>
<td>VICKI</td>
<td>ANDRE</td>
</tr>
<tr>
<td>728650899</td>
<td>ROBERT</td>
<td>BALCH</td>
</tr>
<tr>
<td>728675199</td>
<td>I</td>
<td>BALL</td>
</tr>
<tr>
<td>878454567</td>
<td>ERNEST</td>
<td>BROWN</td>
</tr>
<tr>
<td>878456789</td>
<td>ERNEST</td>
<td>SMITH</td>
</tr>
<tr>
<td>987654321</td>
<td>MIKE</td>
<td>MILLER</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SSW NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>241406699</td>
</tr>
<tr>
<td>251207299</td>
</tr>
<tr>
<td>251402999</td>
</tr>
<tr>
<td>251457199</td>
</tr>
<tr>
<td>519300699</td>
</tr>
<tr>
<td>542269599</td>
</tr>
<tr>
<td>543320899</td>
</tr>
<tr>
<td>543329499</td>
</tr>
<tr>
<td>543431899</td>
</tr>
<tr>
<td>543440399</td>
</tr>
<tr>
<td>543493499</td>
</tr>
<tr>
<td>543706899</td>
</tr>
<tr>
<td>543892799</td>
</tr>
<tr>
<td>549525199</td>
</tr>
<tr>
<td>549684999</td>
</tr>
<tr>
<td>549703599</td>
</tr>
<tr>
<td>721193599</td>
</tr>
</tbody>
</table>
KEY INVENTORY CONTROL SYSTEM

USER MANUAL

by

Robin Fauntleroy

Michele Miley
Table of Contents

Introduction ............................................ 402
System Description ..................................... 404
"Logging On" to the System ............................... 405
The Key Inventory Control Menu ....................... 406
The Front Office Menu .................................. 409
The Key Issue Option ................................... 410
The Return Key Function ................................ 413
The Information Access Menu ............................ 414
The Keyholder Information Menu ....................... 416
The Key Number Information Function ................ 418
Building and Room Number Information ............... 418
The Information on Keys Held Function .............. 419
The Key Inventory Information Function ............. 419
The Daily Cash Balance Function ...................... 419
The Report or Letter Generation Function .......... 420
The Locksmith Menu ........................................ 425
The Complete Key Orders Menu ............................. 426
The List Orders Function .................................. 428
The Change Orders Function ................................ 428
The Record Completion Function ......................... 429
The Print Orders Function .................................. 430
The Replace Key Function .................................. 430
The Rekey Lock Function .................................. 430
The Inventory Control Menu ................................. 433
The Change Inventory Function ............................ 435
The Order Parts Function ................................... 436
The Add Item Function ...................................... 436
The Delete Item Function ................................... 437
The Print Inventory Function ............................... 438
The Maintenance Function .................................. 439
The File Restoration Menu .................................. 439
"Logging Off" the System .................................... 441
Appendix A: Sample Letters ................................. 442
Appendix B: Sample Reports ................................. 443
Introduction

The Key Inventory Control System is a "menu-driven" program, meaning that when you run the program, and you are at a point that you may make one of many choices, those choices will be listed on the screen with a brief description of each choice, and a letter corresponding to that choice. To choose one of the displayed choices, simply type its corresponding letter followed by a carriage return. Should the user inadvertently type an invalid choice, the menu will be redisplayed and another opportunity will be given to enter the desired option. There are parts of the program that will require typing a response, but no menu will be displayed. This happens when the program needs some information from you in order to carry out the function that you have selected from the menu. In each of these cases, a "prompt" will be shown on the screen, or a question concerning the data the program needs. If the response must be entered in a special format, the "prompt" will
explain this also. Furthermore, if there are a limited number of responses to be made, such as "yes or no", these responses will be listed after the prompt.

In response to a prompt, there may be a time when you inadvertently type the wrong answer. After each prompt or series of prompts, the information you have typed in will be redisplayed, and you will be asked if the response you typed is the correct one. If it is not, simply answer "N" for no, and you will be given another chance to retype the information. If the information is correct, type "Y" for yes, and the program will carry out the function you requested, or if needed, will ask you for more information.

This manual is designed to walk through the program with you in the order that menu choices are displayed. Each function will be discussed and explained, and examples of the information you will see on the screen are shown. Each section is
titled according to the function of that particular menu choice.

System Description

The following pages provide a walkthrough of each of the available functions within the Key Inventory Control program. You may either read this section while working at the terminal, or prior to experimenting with the program. Examples of what you will encounter in the program are provided so that if you are not working at the terminal, you will know what to expect of each section.

The Key Inventory Control System is divided into three major parts, the Front Office Procedures, Locksmith Procedures, and Maintenance Procedures. The functions under the Front Office Procedures include Key Issue, Key Return, Information Access, and Report or Letter Generation. Under the Locksmith portion of the system you will find Key Order Completion, Key Replacement, Lock Rekeying, and Inventory Control. Finally, the
Maintenance subsystem contains the File Restoration function in addition to several automatic program maintenance procedures.

"Logging On" to the System

When you first turn on the terminal, you will see a blank screen. When you type a carriage return, the line:

```
WELCOME TO THE UNIVERSITY OF MONTANA
enter class
```

will appear on the screen. The class refers to the particular computer you have an account on, and for this program, that is "deca", so at this time type "deca" (without the quotation marks) and type a carriage return. Another message will be displayed on the terminal followed by the "@" sign. Next to the @ sign, type:

```
LOG CS.GRAD.ROBMICH KICS
```
When you type KICS, it will not show up on the screen. This is the password to enter the system, and should not be given to those not authorized to use the Key Inventory Control System. Several messages will be displayed, and finally an asterisk ("**") will appear on the screen. At the asterisk, type "use keys", and the program will begin running.

The Key Inventory Control Menu

Once you are in the Key Inventory Control System, the main menu will be displayed on the screen. It looks like:

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* *
* KEY INVENTORY CONTROL MENU *
* *
* *
* Enter program choice: *
*   A Front Office Procedures *
*   B Locksmith Procedures *
*   C Maintenance Procedures *
*   D Exit Program *
* *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```
To choose the Front Office Procedures, simply type A in either upper or lower case followed by a carriage return. Likewise, to choose any of the other options, type the letter preceding it followed by a carriage return. If you choose option D, Exit Program, you will leave the program, and may then log off of the system. Choosing option A will result in the prompt:

Enter Password:

At this point you should enter the password for the Front Office Procedures, which is currently "1234567", however, you may change that to any password as long as it is seven letters/characters or less. If the wrong password is entered at this point, the message will appear:

Incorrect Password
Please Re-enter Password:
If the password is entered incorrectly again, you will not be allowed access to the program. Mail will be sent to the account administrator concerning the unauthorized attempted access. The password is displayed on the screen as you type it in, then the screen is cleared, however, this may be a slow process at times, therefore you should take care that unauthorized personnel do not witness this process.
The Front Office Menu

Once the password has been correctly entered for the Front Office Procedures, the Front Office Menu will be displayed. The menu appears as:

************************************************************
    FRONT OFFICE MENU
    Enter Function Choice:
    A  Key Issue
    B  Key Return
    C  Information Access
    D  Report or Letter Generation
    E  Exit

************************************************************

This menu lists the major functions within the Front Office portion of the system. Key Issue is chosen when an individual requests a key, whereas Key Return is the choice when an individual returns a key. The function of Information Access is to provide the user with a means of accessing a variety of information stored within the system.
Report of Letter Generation is utilized when the user wishes to generate letters to keyholders leaving the University system or when a specific report is desired. Exit will return you to the Main Menu.

The Key Issue Option

If the Key Issue function was chosen from the Front Office Menu, you will be prompted for the social security number of the individual requesting a key issue. If this number is not known, then you will be prompted to type in a zero. The prompt is as follows:

Please enter 9-digit social security number, If not available, simply type a 0 (zero). Social Security Number:

You will be asked if this is the correct entry and provided with an opportunity to change the number if it is not correct. Once you are satisfied that the entry is accurate, the program will attempt to
find the name associated with that social security number. If it is found, then it will be displayed on the screen and you may verify that with the individual, however, if it is not found you will be prompted to enter the last name, first name, and middle initial. The prompt will appear as follows:

Please enter issuee's last name:

Please enter issue's first name and middle initial:

In the case that there are several individuals with the same last name within the University records, these names and social security numbers will be displayed for the user to attempt to distinguish between the various records. If the user can determine the correct record, the program will prompt for the corresponding social security number. Otherwise, a unique identification number will be assigned to that individual by the program for the exclusive use of the Key Inventory Control
System. Following correct entry of the issuee's name, or verification of the name displayed by the program, you will be prompted for the building and room number that the individual wishes to be issued a key for. This will be displayed as:

Please enter building abbreviation
(maximum 3 characters):

Please enter room number
(maximum 4 characters):

Finally, you will be prompted for the deposit amount. If the deposit amount is the standard $2.00, no response is required except a carriage return <cr>. However, any other amount must be entered at this point. Entry must be an integer value. For example, $1.00 should be entered as 1. This prompt will appear as:

Enter deposit amount
If amount is other than standard, type in new amount, otherwise type a carriage return <cr>: 
This completes the key issue process for that particular key. You will be asked if there is another key to issue for that particular individual. If so, you will be asked to enter the building and room number and deposit amount. This will continue until you respond 'N', there are no more keys to issue to that individual and you will then be returned to the Front Office Menu.

The Return Key Function

If choice "B", Key Return, was selected from the Front Office Menu, information about the individual returning the key must be obtained. This procedure will follow that which was discussed previously under Issue Key regarding the issuee's social security number and/or name. Following this procedure, rather than requesting a building and room number, the program will request the number of the key being returned. This will be displayed as:
Enter the key number of the returned key:

After completing this process, you will again have the opportunity to continue returning keys for that individual. You will then be returned to the Front Office Menu.

The Information Access Menu

If Information Access was the function chosen from the Front Office Menu, then the Information Access Menu will be displayed. This allows you to choose the type of information you would like to obtain.
The selection of option "A", Keyholder Information, is used for obtaining a list of persons possessing keys for a particular building and room number or key number. Option "B", Key Number Information, requires the input of building and room number and will provide you with the corresponding key number. Selection "C", Building and Room Number Information, will display the corresponding building and room number, once a key number is entered. Option "D", Information on Keys Held will list all keys possessed by a given
individual. Key Inventory Selection, selection "E", provides you with a summary of the number of keys on hand, the number issued, and the total number. Option "F", Daily Cash Balance, will print out the current balance of cash received and refunded. It may be cleared upon request. The choice of option "G", Exit, will return you to the Front Office Menu.

The Keyholder Information Menu

Selection of the option of Keyholder Information in the Information Access Menu will result in the display of the Keyholder Information Menu. This menu will allow the user to choose which information to enter in order to obtain information concerning key holders.
Depending upon the information available to you, you may determine persons possessing a particular key or keys. Selection of option "A" will result in the prompt:

Please enter the building abbreviation (maximum 3 characters):

Please enter the room number (maximum 4 characters):

Those persons possessing the key for that building and room will then be displayed. However, should you have a key number or numbers, choice of option
"B" will then prompt you for those numbers as follows:

Please enter a key number
or a 'q' to indicate there are no more keys:

This will result in a list of all persons who have been issued that group of keys. Choice "C" will return you to the Information Access Menu.

The Key Number Information Function

Selection of this option, "B", will provide you with the key number which corresponds to a particular building and room number. You will be prompted for this information as option "A" of the Key Holder Information Menu above.

Building and Room Number Information

Option "C", will display the building and room number corresponding to a given key number. You will be prompted for the key number in the
following manner:

Please enter the key number:

The Information on Keys Held Function

This function, "D", within the Information Access Menu, will display all keys held by a particular individual. You will be prompted, as in Issue Key, for a social security number, or, if not available a last and first name. Please see the section under Issue Key for more specific information on these prompts.

The Key Inventory Information Function

Option "E", given a valid key number, will display a summary of the quantity on hand, number issued, and total number of keys. This prompt will be identical to that seen in Option "C".

The Daily Cash Balance Function
This function, "F", on the Information Access Menu, will display the current balance of all cash received and refunded throughout the chosen time period. This may be used to balance the cash on hand for the daily deposit. An option is provided within this function to clear the total. Choice "G", Exit will return you to the Front Office Menu.

The Report or Letter Generation Function

If the user had entered the Report or Letter Generation function at the Front Office Menu, option "D", and then indicated (from a single prompt) the desire to generate reports, then the Report Generation Menu would appear. This menu indicates which reports the user may produce.
The report generation function allows the user to choose among a variety of reports available. The keyholder report will list keyholders and the corresponding keys that they have issued to them. The inventory report is a summary of the inventory available, and includes such information as total number of keys, amount issued, amount available, etc. The location report is a cross reference between the key numbers and the building and room numbers. Finally, the key retirement report is a list of all keys which have been 'retired' and the date of retirement.
The Keyholder Report Option

This option will allow you to choose whether you would like a report consisting of the entire information, (a large report), or a "individual report", which consists of a subset of the information. Should you choose the "Total" report at the prompt, you will be asked whether you would like the information sorted by the social security number of the individuals or the key numbers. After responding to this prompt, the report will be generated and you will be returned to the menu. The final report will be stored in a file called "Keyholder.txt". The individual reports are subdivided by keyways. Should you choose this option, you will be prompted for the keyway number which you want the report based upon. As there are many keys under a particular keyway number, you will then have the same option of sorting on either social security number of key number.
The Inventory Report Option

This report option will also provide you with the opportunity of subdividing the report into smaller divisions. You will see the same prompt for either a "Total" report or an "Individual" report. If you choose the "Individual" report, you will be asked for the keyway number you wish the report to be based upon. The final report will be stored in a file called "Inven.txt".

The Location Report Option

This report will provide a cross reference between key numbers and building and room numbers. It also can be divided into particular keyway numbers, to allow for a more concise report. The final report may be found in a file called "Locat.txt".

The Key Retirement Option

The key retirement report will simply print a report of all keys that have been put on the
"retired" list and the date of retirement. The final report will be located in a file called "Retire.txt".

The Letter Generation Function

The letter generation function is not implemented at this time, however, sample letters can be found in the Appendix. A choice of any of the options, except "Q", will display a "Not Implemented Yet" message. Choice of "Q", will return you to the Front Office Menu.

If you chose option, "E", from the Front Office Menu, you will return to the Key Inventory Control Menu.
The Locksmith Menu

If selection "B", Locksmith Procedures is chosen at the Key Inventory Control Menu, the program will prompt you for a password in exactly the same manner encountered in the Front Office Procedures. The present password for this portion of the program is "2345678", however this may be changed if desired. Once the password has been entered correctly, the following menu is presented at the onset of the locksmith portion of the program.

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* * *
* LOCKSMITH MENU *
* * *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* * *
* Choose option: *
* *
* A Complete Key Orders *
* B Replace Key *
* C Rekey Lock *
* D Inventory Control *
* E Exit *
* *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
The Complete Key Orders function, option "A", provides the user with access to the orders received for keys from the Front Office Procedures. Orders may be viewed, completed, printed, and changed from this selection. Selection "B", Replace Key, records the return and subsequent replacement of a damaged or ineffective key. Option "C", Rekey Lock is designed to record the retirement of a key, and to generate the new key number to be used for that lock. Inventory Control, selection "D", allows the viewing, manipulation, and recording of keyway inventory levels. Option "E" will return you to the Key Inventory Control Menu.

The Complete Key Orders Menu

The following menu is presented to the user with the choice of 'A' on the Locksmith Menu.
Option "A", List Orders, will display all the orders needed by the Front Office to maintain an adequate supply of keys available for issue for each building and room. Change Orders, selection "B", allows you to change the priority and/or amount of keys in the key order list. Selection "C", Record Completion, will adjust key inventory levels as orders are completed. Print Orders, option "D", will produce a printed copy of the order list. Finally, option "E", Exit, will return you to the Locksmith Menu.
The List Orders Function

No input is required from you when this choice is selected from the Complete Key Orders Menu. The list of orders produced by the Front Office Procedures and the Rekey Lock Function will be displayed on the screen at this time.

The Change Orders Function

When this option is chosen from the Complete Key Orders Menu, a list of orders is displayed as in the List Orders Function, however, each order is preceded by a number to indicate its position in the list. You will be prompted for the order that you wish to change. The prompt is as follows:

Please type the number of the record you wish to change. (Enter a 0 (zero) if you wish to exit this portion of the program.)
Order Number:

What is the new priority you would like to assign? Priority:

What is the new amount to be assigned? Amount:
As with Front Office Procedures, if an incorrect entry is made, you will be given the opportunity to re-enter the correct data. The new amount and priority will be entered into the order file, replacing the existing values.

The Record Completion Function

Selection of choice "C", Record Completion, will display the order file as in the Change Order Function, with the preceding number. Once all keys have been made for a particular order, you may choose this option to delete that entry from the list. This will update the Front Office inventory file and the Locksmith keyway inventory file. You will again be prompted for the order number as follows:

Please type the number of the record which has been completed. (Enter a 0 (zero) if you wish to exit this portion of the program.)
Order Number:
The Print Orders Function

Choice "D" of the Complete Orders Menu will print a paper copy of the order list. To get a printout, simply type the 'print' key on the keyboard.

The Replace Key Function

Selection "B" from the Locksmith Menu, Replace Key, allows you to record the return of an irreparably damaged key. This will update the appropriate files. You will be prompted for the key number in the following way:

Enter the key number of the returned key. (If you wish to exit without entering the key number, type E for Exit.)
Key Number:

The Rekey Lock Function

Option "C" of the Locksmith Menu, the Rekey Lock Function, is used when a particular room is
being rekeyed. It will determine the new bit cut number to be used, whether an already existing and retired key, or an entirely new key. Upon entry to this portion of the program, you will be prompted as follows:

Do you wish to continue with the lock rekeying process? (Y/N)

At this point you may exit this section by typing "N", or proceed by typing "Y". If you exit, you will be returned to the Locksmith Menu, and if you continue, you will be prompted for the building and room number as follows:

Please enter the building and room number that are being rekeyed.
Building:
Room Number:

As in other parts of the system, the program will ask you if this information is correct before performing any functions. If it is correct, the fol-
Do you wish to use a new key number or an existing one?
Type N for New, and E for Existing:

The choice of "E" is used when you have a retired key that you would like to use to rekey the lock, but are unsure if it has been in retirement long enough. Otherwise, you may enter "N", and the program will generate a new number with the same keyway that is two bit cuts away from any other number and is not in use. If you have decided to choose the "E" option, the following prompt will appear:

Please enter the old key number you wish to use.
Old Key Number:

Again, the information you have entered will be verified, and if correct, will check whether that key is available for use. If it is available for
use, the following prompt will appear:

Enter the amount you have on hand.
Amount:

In response to this, you should enter the number of the retired keys you have on hand. An order will automatically be generated and placed on the order list as described in the Complete Key Orders Menu section of this document. In addition, the old key will be listed as retired, and the new key number to be used will be listed as the key opening that particular building and door.

The Inventory Control Menu

If you selected choice "D", Inventory Control, from the Locksmith Menu, the Inventory Control Menu will be displayed. It appears as:
Option "A", Change Inventory, is used to manually change the quantity on hand of key blanks listed in the inventory records, rather than rely on the automatic generation of order amounts. Order Parts, option "B", will generate the order amounts needed for each key blank and print these out on the printer. Selection "C", Add Item, is chosen when you receive an order and wish to increase the inventory levels listed. Delete Item, selection "D", is used to delete the record for a keyway that is no longer being used. Option "E", Print
Inventory, will produce a paper copy of the amounts on hand of each key blank. Finally, option "F", Exit, will return you to the Locksmith Menu.

The Change Inventory Function

When you enter the Change Inventory Function, you will be prompted with:

Do you wish to continue with the inventory change process? (Y/N)

If you choose "N", you will be returned to the Inventory Control Menu, however, if you choose "Y", the following prompt will appear:

Please enter the keyway number for which you wish to make a change.
Keyway Number:

Once you have entered the keyway number and verified that it is correct, you will be prompted with:
What is the new amount you would like to enter?
Amount:

Again, once you've entered the amount and verified that it is correct, both the keyway number and the new amount listed will be displayed on the screen.

The Order Parts Function

The Order Parts Function, selection "B" on the Inventory Control Menu, is used to produce a listing of the number of key blanks to be ordered to bring the inventory levels of each blank to the threshold level.

The Add Item Function

If you chose option "C" at the Inventory Control Menu, this is the function you will be performing. This is used when an order is received and inventory levels must be increased to reflect that. You will be prompted as follows:
Please enter the keyway number for which you have received inventory.
Keyway Number:

What is the amount received?
Amount Received:

The amount you type in as amount received will be added to the current quantity on hand once you have indicated that it is correct. You will next be prompted with:

Do you have more entries to add? (Y/N)

A choice of "Y" will repeat the process you have just completed, while the choice of "N" will return you to the Inventory Control Menu.

The Delete Item Function

If a keyway number is no longer in use, it may be deleted from the files with this function. You will be prompted with:
Please enter the keyway number for the record to be deleted.
Keyway Number:

If there are no records with that keyway number, a message will appear on the screen informing you of this, and no change will be made to the files, however, if that number does exist, it will be deleted from the files.

The Print Inventory Function

The Print Inventory function prints a paper copy of the amounts on hand for each keyway number. To activate the printer, simply type the ′print′ key.
The Maintenance Function

If, at the Key Inventory Control Menu level, you had chosen the Maintenance Program option, and entered the password correctly, this would allow access to the file restoration function. This function allows you to pull previously stored files from magnetic tape to be accessed via interactive 1022 commands. The menu provided for this function allows you to choose which files you wish to restore. In addition to the file restoration function, the Maintenance Function performs other maintenance routines automatically, such as weekly saving of all files on magnetic tape, and checking threshold levels of Front Office keys and placing those needed into the Locksmith order list.

The File Restoration Menu

If you selected the Maintenance Program at the Key Inventory Control Menu, you would see the following menu:
Each of the choices listed represents a file that has been automatically saved weekly by the Maintenance Function, and which, should anything happen to your current files, could be restored so that a minimum of data would be lost. You may choose to only restore one file, perhaps to view historical data, or you may choose to restore them all. Because there is no current tape at the Computer Center for the Physical Plant, it is suggested that you do not run these modules at present,
although they are implemented and tested.

"Logging Off" of the System

When you are finished running the Key Inventory Control System, and you have selected the Exit Program option from the Key Inventory Control Menu, you will see an asterisk ("*") displayed on the screen. At this point type "exit" and a "@" sign will appear. Type "logout" at the "@" sign, and this will disengage your connection to the computer. Now you may turn off your terminal and Gandalf switch.
Appendix A: Sample Letters

The following letter shows the content of letters sent out to personnel and students leaving the University system if they still possess keys. The letter itself will be kept in a separate file so that it will be easily changed should another format be desirable in the future.

Date

Dear Student/University Employee,

It has come to our attention that you will soon be leaving the University system, and that you still possess one or more keys from the Physical Plant. Please return these to the Physical Plant prior to leaving, and your deposit will be refunded. The keys we show that you have are:

- Key Number 1
- Key Number 2
- ...
- Key Number n

Thank you for your cooperation!

University of Montana Physical Plant
Appendix B: Sample Reports

Several kinds of reports may be generated by the report generation function of the Key Inventory Control System. The format of these reports is shown below. Each sample is a greatly abbreviated representation of the actual report, but may be used to understand the layout of each.

Keyholder Report

<table>
<thead>
<tr>
<th>Keyholder Name:</th>
<th>Key Number:</th>
<th>Date Issued:</th>
<th>Deposit:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doe, John E.</td>
<td>3K68179</td>
<td>2/12/85</td>
<td>2.00</td>
</tr>
<tr>
<td></td>
<td>3K65132</td>
<td>3/15/84</td>
<td>2.00</td>
</tr>
<tr>
<td>Smith, Mary L.</td>
<td>3K63128</td>
<td>6/18/75</td>
<td>2.00</td>
</tr>
</tbody>
</table>

Inventory Report

<table>
<thead>
<tr>
<th>Key Number:</th>
<th>Quantity on Hand:</th>
<th>Quantity Out:</th>
<th>Total Number:</th>
</tr>
</thead>
<tbody>
<tr>
<td>3K68179</td>
<td>10</td>
<td>25</td>
<td>35</td>
</tr>
<tr>
<td>2F43289</td>
<td>3</td>
<td>17</td>
<td>20</td>
</tr>
<tr>
<td>3K63219</td>
<td>18</td>
<td>5</td>
<td>23</td>
</tr>
</tbody>
</table>
Location Report

<table>
<thead>
<tr>
<th>Key Number:</th>
<th>Building:</th>
<th>Room Number:</th>
</tr>
</thead>
<tbody>
<tr>
<td>3K68179</td>
<td>UH</td>
<td>315</td>
</tr>
<tr>
<td>3K47328</td>
<td>UH</td>
<td>22</td>
</tr>
<tr>
<td>2F53728</td>
<td>LA</td>
<td>103</td>
</tr>
</tbody>
</table>

Key Retirement Report

<table>
<thead>
<tr>
<th>Key Number:</th>
<th>Quantity on Hand:</th>
<th>Quantity Out:</th>
<th>Total Number:</th>
<th>Date Retired:</th>
</tr>
</thead>
<tbody>
<tr>
<td>3K68179</td>
<td>33</td>
<td>2</td>
<td>35</td>
<td>1/15/85</td>
</tr>
<tr>
<td>2F34287</td>
<td>12</td>
<td>17</td>
<td>29</td>
<td>5/14/85</td>
</tr>
</tbody>
</table>
Module Name: 10 PASSWORD MAIN
Parameters:
In Only:
Out Only:
In/Out: GL_ALLOW_ENTRY
Coded By: Robin FauntLeRoy
Date Last Modified: July 9, 1985
Reason Modified:

Module Description: This module controls the password functions. It calls ACCEPT_PASSWORD and receives the GL_ALLOW_ENTRY flag from this module. Depending on the flag it may or may not call DISALLOW_ACCESS.

DEFINE TEXT 7 PASSWORD.

IF GL_MAIN_MENU_CHOICE EQ (\'A\', \'a\', \'B\', \'b\', \'C\', \'c\') THEN
   PRINT \"Enter Password:\".
   ACCEPT PASSWORD.
   PUSH USING BL
   POP END.
   CALL ACCEPT_PASSWORD.
ENDIF.

IF GL_ALLOW_ENTRY EQ \"F\" THEN
   PRINT \"Incorrect Password\".
   PRINT \"Please Re-enter Password\".
   ACCEPT PASSWORD.
   PUSH USING BL
   POP END.
   CALL ACCEPT_PASSWORD.
ENDIF.

IF GL_ALLOW_ENTRY EQ \"F\" THEN
   CALL DISALLOW_ACCESS.
ENDIF.
RETURN.

ACCEPT PASSWORD: @R101_ACCEPT_PASSWORD.DMC
DISALLOW_ACCESS: @R102_DISALLOW_ACCESS.DMC
Module Name: 101 ACCEPT PASSWORD

Parameters:
  In Only: PASSWORD
  Out Only:
  In/Out: G1_MAIN_MENU_CHOICE
  G1.Allow_Entry

Coded By: Robin FauntLeRoy
Date Last Modified: July 9, 1985

Reason Modified:

Module Description: This module compares the value of the password accepted with the appropriate password for the system requested and sets the G1.Allow_Entry flag accordingly.

DEFINE TEXT 7 FRONT_PASSWORD
TEXT 7 LOCK_PASSWORD
TEXT 7 MAINT_PASSWORD.

LET FRONT_PASSWORD EQ "1234567".
LET LOCK_PASSWORD EQ "2345678".
LET MAINT_PASSWORD EQ "3456789".

IF G1.MAIN_MENU_CHOICE EQ ( A', 'a' ) AND PASSWORD EQ FRONT_PASSWORD OR G1.MAIN_MENU_CHOICE EQ ( B', 'b' ) AND PASSWORD EQ LOCK_PASSWORD OR G1.MAIN_MENU_CHOICE EQ ( C', 'c' ) AND PASSWORD EQ MAINT_PASSWORD THEN LET G1.Allow_Entry EQ T''.
ELSE LET G1.Allow_Entry EQ F''.
ENDIF.

RETURN.
Module Name: 102 Disallow_Access
Parameters:
In Only:
Out Only:
In/Out:
Coded By: Robin FauntLeRoy
Date Last Modified: July 9, 1985
Reason Modified:

Module Description:
This module is the controller module for the processes that execute with an attempt by someone who doesn't know the correct password to enter the system.

CALL SEND_MAIL.
CALL LOGOUT_USER.
RETURN.

SEND_MAIL: @R1021_SEND_MAIL.DMC
LOGOUT_USER: @R1022_LOGOUT_USER.DMC
Module Name: 1021 SEND_MAIL
Parameters:
   In Only:
   Out Only:
   In/Out:
Coded By: Robin PauntleRoy
Date Last Modified: July 9, 1985
Reason Modified:

Module Description:
This module simply pushes out to the monitor level, submits a batch job, which sends mail to the user notifying them of an unauthorized entry attempt and then pops to return to 1022.

PUSH USING SUBMIT MAIL/OUTPUT:ERRORS
POP END.
RETURN.
CALL PROGRAM_END.

Module Name: 1022 Logout User
Parameters:
  In Only: none
  Out Only: none
  In/Out: none
Coded By: Robin FauntLeRoy
Date Last Modified: July 17, 1985
Reason Modified:

Module Description:
  This module is called if someone attempts to use the keys
  system and does not have the proper passwords. It simply calls
  the program end which closes all the files and exits 1022.

CALL PROGRAM_END.
Module Name: 11 Front Office Main
Parameters:
   In Only: none
   Out Only: Gll_LAST_NAME
   Gll_FIRST_NAME
   Gll_NAME
   Gll_SS_NUMBER
   In/Out: none
Coded By: Robin FauntLeRoy
Date Last Modified: July 9, 1985
Reason Modified:

Module Description:

  The Main module is the controlling module for the front office and displays a menu asking the user to choose between the functions provided. It passes no parameters and accesses no files.

DEFINE TEXT 1 LI_FRONT_OFFICE_MENU_CHOICE
  TEXT 17 Gll_LAST_NAME
  TEXT 30 Gll_NAME
  TEXT 13 Gll_FIRST_NAME.

DEFINE INTEGER Gll_SS_NUMBER.

REPEAT
  PERFORM
  POP END.

****
**
.*
**
**
****
PRINT "FRONT OFFICE MENU"

****
PRINT "Enter Function Choice:
**
PRINT "A Key Issue"
**
PRINT "B Key Return"
**
PRINT "C Information Access"
**
PRINT "D Report or Letter Generation"
**
PRINT "E Exit Program"
**
PRINT "" END.
ACCEPT L1_FRONT_OFFICE_MENU_CHOICE.
IF L1_FRONT_OFFICE_MENU_CHOICE NEQ ('A', 'a', 'B', 'b', 'C', 'c', 'D', 'd', 'E', 'e') THEN
   PRINT "Invalid Choice, please try again."
ENDIF.
UNTIL L1_FRONT_OFFICE_MENU_CHOICE EQ ('A', 'a', 'B', 'b', 'C', 'c', 'D', 'd', 'E', 'e').
IF L1_FRONT_OFFICE_MENU_CHOICE EQ ('A', 'a') THEN
   CALL ISSUE_KEY.
ELSEIF L1_FRONT_OFFICE_MENU_CHOICE EQ ('B', 'b') THEN
   CALL RETURN_KEY.
ELSEIF L1_FRONT_OFFICE_MENU_CHOICE EQ ('C', 'c') THEN
   CALL INFORMATION_ACCESS.
ELSEIF L1_FRONT_OFFICE_MENU_CHOICE EQ ('D', 'd') THEN
   CALL GENERATE_CORRESPONDENCE.
ELSE
   PRINT " ".
ENDIF.
UNTIL L1_FRONT_OFFICE_MENU_CHOICE EQ ('E', 'e').
RETURN.

ISSUE_KEY: @R111_ISSUE_KEY.DMC
RETURN_KEY: @R112_RETURN_KEY.DMC
INFORMATION_ACCESS: @R113_INFORMATION_ACCESS.DMC
GENERATE_CORRESPONDENCE: @R114_GEN_CORRESP.DMC

0
Module Name: G11 ISSUE KEY
Parameters:
In Only:
Out Only: G11_KEY_NUMBER
         G11_AVAILAEBLE
         G11_FOUND
         G11_BLD_NUMBER
         G11_ROOM_NUMBER
         G11 DEPOSIT
In/Out: G11_SS_NUMBER
         G11_Last_Name
         G11_First_Name
Coded By: Robin Fauntleroy
Date Last Modified: July 10, 1985
Reason Modified:
Module Description:
This module will collect the information necessary to issue a key and then update the files to reflect this information.

!!!!!!!

DEFINE TEXT 1 G11_AVAILAEBLE
TEXT 1 L111 ANSWER
TEXT 1 L111 FILLER
TEXT 1 G111 FOUND
TEXT 1 L111 DONE
TEXT 10 G111 KEY NUMBER
TEXT 3 G111 BLD NUMBER
TEXT 4 G111 ROOM NUMBER
TEXT 5 L111 DEPOSIT.

DEFINE INTEGER L111 AMOUNT
INTEGER G111 DEPOSIT.

LET L111 FILLER EQ " ".
LET G111 DEPOSIT EQ 2.
LET G111 FOUND EQ "F".
LET G111 DONE EQ "F".
LET G111 AVAILABLE EQ "F".
REPEAT
PRINT " ".
PRINT "Please enter 9-digit social security number,".
PRINT "If not available, simply type a 0 (zero).".
PRINT " ".
PRINT "Is this correct? (Y/N) " END.
ACCEPT G111 SS NUMBER.
PRINT " ".
PRINT "SSN: " G111 SS NUMBER.
PRINT "Is this correct? (Y/N) " END.
ACCEPT L111 ANSWER.
UNTIL L111 ANSWER EQ ( 'Y','y').
IF G111 SS NUMBER EQ 0 THEN
  LET G111 FOUND EQ "F".
ELSE
  CALL FIND_NAME.
ENDIF.
IF G111 FOUND EQ "F" THEN
PRINT " ".
PRINT "There is no corresponding record for that SSN.".
REPEAT
PRINT " ".
.
PRINT FMT $ "Please enter issuee's last name: " END.
ACCEPT G11_LAST_NAME.
PRINT " ".
PRINT FMT $ "Please enter issuee's first name and middle initial
: " END.
ACCEPT G11_FIRST_NAME.
PRINT " ".
PRINT "LAST NAME: " G11_LAST_NAME.
PRINT "FIRST NAME: " G11_FIRST_NAME.
PRINT " ".
PRINT FMT $ "Is this correct? (Y/N) " END.
ACCEPT L111_ANSWER.
UNTIL L111_ANSWER EQ ( 'Y', 'y' ) .
LET G11_NAME EQ STRIM(G11_FIRST_NAME) + L111_FILLER + STRIM(G11_LAST_NAME).
ENDIF.
IF G11_SS_NUMBER EQ 0 THEN
CALL FIND^SSN.
IF G11_SS_NUMBER NEQ 0 THEN
LET G111_FOUND EQ "T".
ENDIF.
ENDIF.
REPEAT
REPEAT
PRINT "Please enter building abbreviation,".
PRINT "(maximum 3 characters)."
PRINT FMT $ "Bldg: " END.
ACCEPT G111_BLD_NUMBER.
PRINT "Please enter room number,".
PRINT "(maximum 4 characters)."
PRINT FMT $ "Room No: " END.
ACCEPT G111_ROOM_NUMBER.
PRINT "Bldg: " G111_BLD_NUMBER.
PRINT "Room No: " G111_ROOM_NUMBER.
PRINT " ".
PRINT FMT $ "Is this correct? (Y/N) " END.
ACCEPT L111_ANSWER.
UNTIL L111_ANSWER EQ ( 'Y', 'y' ) .
CALL FIND^KEY_NUMBER.
IF G111_AVAILABLE EQ "T" THEN
PRINT "Enter deposit amount".
PRINT "If amount is other than standard, type in ".
PRINT FMT $ "new amount, otherwise type a carriage return: " END.
ACCEPT L111_DEPOSIT.
LET L111_AMOUNT EQ $INT(L111_DEPOSIT).
IF L111_AMOUNT GT 0 THEN
LET G111_DEPOSIT EQ L111_AMOUNT.
ENDIF.
PRINT " ".
PRINT "The following record will be entered into the files: ".
PRINT "Name: " G111_NAME.
PRINT "Bldg & Room: " G111_BLD_NUMBER G111_ROOM_NUMBER.
PRINT "Key Number: " G111_KEY_NUMBER.
PRINT "Deposit Amt: " G111_DEPOSIT.
PRINT FMT $ "Is this correct? (Y/N) " END.
ACCEPT L111_ANSWER.
IF L111_ANSWER EQ ( 'Y', 'y' ) THEN
CALL UPDATE_NAME_FILES.
ENDIF.
ENDIF.
PRINT " ".

PRINT FMT $ "Do you have another entry for this individual (Y/N) " END.
ACCEPT L11_DONE.
UNTIL L11_DONE EQ "N".
RETURN.

FIND_KEY_NUMBER: @R1111_FIND_KEY_NUMBER.DMC
UPDATE_NAME_FILES: @R1112_UPDATE_NAME_FILES
FIND_NAME: @R117_FIND_NAME.DMC
@
Module Name: L111 Find Key Number
Parameters:
   In Only: G111_BLD_NUMBER
   G111_ROOM_NUMBER
   Out Only:
   In/Out: G111_AVAILABILITY
   G111_Key_Number
Coded By: Robin FauntLeRoy
Date Last Modified: 7/10/85

Module Description:
This module will check the locate file to determine if there
is a key that fits the appropriate room and building number given
by the user who wishes to be issued a key.
If an appropriate key number is found, then Check_InVENTORY is
called to determine if there is a key available to be issued.

DEFINE TEXT lL111_WAIT.

@R11111_CHECK_INVENTORY.DMC
Module Name: 11111 Check Inventory

Parameters:

Out Only:

In/Out: 11111_Available

In/Out: 11111_Key_Number

Coded By: Robin FauntLeRoy

Date Last Modified: 7/10/85

Reason Modified:

Module Description:

This module takes the key number found by Find.Key.Number and checks to see if there is a key available to be issued. If there is, it decreases the quantity_on_hand by one and sets the available flag to true.

DEFINE TEXT 1 L11111 FOUND
TEXT 1 L11111_WAIT.

LET 11111_AVAILABLE EQ "F".
MAP TO INVEN VIA KIN.
IF SYSNREC EQ 1 THEN
    IF QON GT 0 THEN
        LET L11111_FOUND EQ "T".
        LET 11111_KEY_NUMBER EQ KIN.
        CHANGE QON QON-1.
        LET 11111_AVAILABLE EQ "T".
    ELSE
        PRINT "  ".
        PRINT "There is an insufficient supply of that key ".
        PRINT "to issue one. The key number will be sent to ".
        PRINT "the locksmith to order some additional made.".
    ENDIF.
ELSE
    PRINT "  ".
    PRINT "That key has no record in the inventory file.".
    PRINT FMT S "(Type a carriage return <cr> to continue) "END.
    ACCEPT L11111_WAIT.
ENDIF.
RETURN.
Module Name: 1112 Update Name Files
Parameters:
In Only: G111_Key_Number
Out Only:
In/Out: G111_Ind_Name
G111_Ss_Number
Coded By: Robin FauntLeRoy
Date Last Modified: 7/11/85

Reason Modified:

Module Description:
This module will add information to the Ind Record file and the Name file (if necessary) to reflect a recent key issue.

IF G111_Ss_Number LT 0 THEN
CALL FIND_SSN.
ENDIF.
IF G111_FOUND NEQ "T" THEN
  IF G111_Ss_Number EQ 0 THEN
    DBSET DUMMY_SS.
    FIND ALL.
    LET G111_Ss_Number EQ SSN.
    CHANGE SSN SSN-1.
  ENDIF.
  DBSET NAME.
  ADD SSN G111_Ss_NUMBER LNM G111_LAST_NAME FNM G111_FIRST_NAME.
ENDIF.
DBSET IND_RECORD.
ADD SSN G111_Ss_NUMBER KIN G1111_KEY_NUMBER DEP G1111 DEPOSIT.
CALL BOOKKEEPING.
RETURN.

BOOKKEEPING: @R115_BOOKKEEPING.DMC
FIND_SSN: @R116 FIND_SSN.DMC
Module Name: 112 Return Key

Parameters:

In Only:

In/Out:  Gll_SS_Number
    Gll_First_Name
    Gll_Last_Name

Out Only: Gll_Key_Number

Coded By:  Robin FauntLeRoy

Date Last Modified: 7/11/85

Reason Modified:

Module Description:

This module checks the files for the ssn and key number for an individual who is returning a key.

DEFINE TEXT 10 Gll2_KEY_NUMBER
TFEXT 1 L112_DONE=
TEXT 1 L112_ANSWER.

LET SYSCASE EQ 1.
LET Glll_FOUND EQ "F".
LET Gll_SS_NUMBER EQ 0.
LET L112_DONE EQ "Y".
REPEAT
    PRINT ",,
    PRINT FMT $ "Please enter SSN or a 0 <zero> if not available: " END.
    ACCEPT Gll_SS_NUMBER.
    PRINT ",,
    PRINT "SSN: " Gll_SS_NUMBER.
    PRINT FMT $ "Is this correct? (Y/N) " END.
    ACCEPT L112_ANSWER.
    UNTIL L112_ANSWER EQ ('Y', 'y').
    IF Gll_SS_NUMBER NEQ 0 THEN
        PRINT ",,
        CALL FIND_NAME.
        ENDIF.
    IF Glll_FOUND NEQ "T" THEN
        LET Gll_SS_NUMBER EQ 0.
        REPEAT
            PRINT ",,
            PRINT "Please enter the individual's last name. ",
            PRINT FMT $ "Last Name: " END.
            ACCEPT Gll_LAST_NAME.
            PRINT ",,
            PRINT "Please enter individual's first name ",
            PRINT FMT $ "First Name: " END.
            ACCEPT Gll_FIRST_NAME.
            PRINT ",,
            PRINT "Last Name: " Gll_LAST_NAME.
            PRINT "First Name: " Gll_FIRST_NAME.
            PRINT FMT $ "Is this correct? (Y/N) " END.
            ACCEPT L112_ANSWER.
            UNTIL L112_ANSWER EQ ('Y', 'y').
            CALL FIND_SS.
        ENDIF.
    IF Gll_SS_NUMBER EQ 0 THEN
        PRINT ",
        PRINT "Please enter SSN or a 0 <zero> if not available: " END.
        ACCEPT Gll_SS_NUMBER.
        PRINT ",
        PRINT "SSN: " Gll_SS_NUMBER.
        PRINT FMT $ "Is this correct? (Y/N) " END.
        ACCEPT L112_ANSWER.
        UNTIL L112_ANSWER EQ ('Y', 'y').
        CALL FIND_SS.
    ENDIF.
    IF Glll_FOUND NEQ "T" THEN
        LET Gll_SS_NUMBER EQ 0.
        REPEAT
            PRINT ",,
            PRINT "Please enter the individual's last name. ",
            PRINT FMT $ "Last Name: " END.
            ACCEPT Gll_LAST_NAME.
            PRINT ",,
            PRINT "Please enter individual's first name ",
            PRINT FMT $ "First Name: " END.
            ACCEPT Gll_FIRST_NAME.
            PRINT ",,
            PRINT "Last Name: " Gll_LAST_NAME.
            PRINT "First Name: " Gll_FIRST_NAME.
            PRINT FMT $ "Is this correct? (Y/N) " END.
            ACCEPT L112_ANSWER.
            UNTIL L112_ANSWER EQ ('Y', 'y').
            CALL FIND_SS.
        ENDIF.
    IF Gll_SS_NUMBER EQ 0 THEN
        PRINT ",
        PRINT "Please enter SSN or a 0 <zero> if not available: " END.
        ACCEPT Gll_SS_NUMBER.
        PRINT ",
        PRINT "SSN: " Gll_SS_NUMBER.
        PRINT FMT $ "Is this correct? (Y/N) " END.
        ACCEPT L112_ANSWER.
        UNTIL L112_ANSWER EQ ('Y', 'y').
        CALL FIND_SS.
    ENDIF.
PRINT "There is no record for that individual".
PRINT "This operation is aborted."
PRINT FMT $ "(Type carriage return <cr> to continue) END."
ACCEPT L112_ANSWER.
ELSE
REPEAT
  PRINT " ".
  PRINT "Enter the key number of the returned key. ".
  PRINT FMT $ "Key Number: " END.
  ACCEPT G112_KEY_NUMBER.
  PRINT " ".
  PRINT "Key Number: " G112_KEY_NUMBER.
  PRINT FMT $ "Is this correct? (Y/N) " END.
  ACCEPT L112_ANSWER.
  UNTIL L112_ANSWER EQ ('Y','y').
ENDIF.
RETURN.

CHECK_ALL_NAMES: @R1121_CHECK_ALL_NAMES.DMC
Module Name: 1121 CHECK ALL NAMES
Parameters:
In Only:
Out Only: G112I_DELETE_FLAG
In/Out:
Coded By: ROBIN FAUNTLEROY
Date Last Modified: JULY 17, 1985
Reason Modified:
Module Description:
This module will check to see if that individual who is
returning a key has only one key, if so it will set a flag to check
to delete the corresponding record in the name file, if it exists
there. Also, it will find the appropriate record to delete in the
individual record file.

DEFINE TEXT 1 G112I_DELETE_FLAG
TEXT 1 L112I_WAIT
DEFINE INTEGER G112I_DEPOSIT.

LET L112I_FOUND EQ "F".
DBSET IND_RECORD.
FIND ALL.
FIND SSN EQ G11_SS_NUMBER.
IF SYSNREC EQ 1 THEN
IF KIN EQ G112_KEY_NUMBER THEN
LET G112I_DELETE_FLAG EQ "T".
LET L112I_FOUND EQ "T".
ENDIF.
ELSE
SEARCH KIN EQ G112_KEY_NUMBER.
IF SYSNREC NEQ 0 THEN
LET L112I_FOUND EQ "T".
ENDIF.
ENDIF.
IF L112I_FOUND EQ "T" THEN
LET G111 DEPOSIT EQ -1*DEP.
CALL BOOKKEEPING.
CALL DELETE_FILES.
CALL INCREASE_KEY_AVAILABLE.
ELSE
PRINT "The record corresponding to that individual and ".
PRINT "key number could not be located.".
PRINT FMT S "(type carriage return <cr> to continue) "END.
ACCEPT L112I_WAIT.
ENDIF.
RETURN.

DELETE_FILES: @R11211 DELETE_FILES.DMC
INCREASE_KEY_AVAILABLE: @R11212_INCREASE_KEY_AVAILABLE.DMC
Module Name: 11211 Delete Files

Parameters:
- In Only: G112_key_number
- Out Only:
- In/Out:

Coded By: Robin Fauntleroy

Date Last Modified: 7/18/85

Reason Modified:

Module Description:
This module will delete the appropriate files, after an individual has returned a key.

DEFINE TEXT 1 L11211_RESPONSE
TEXT 3 L11211_BLD_NUMBER
TEXT 4 L11211_RM_NUMBER.

DBSET LOCATE.
FIND ALL.
FIND KIN EQ G112_KEY_NUMBER.
LET L11211_BLD_NUMBER EQ BLD.
LET L11211_RM_NUMBER EQ RN.
DBSET IND_RECORD.
PRINT "The following record will be deleted from the files:"
PRINT "Name: "GL1_NAME.
PRINT "Key Number: "G112_KEY_NUMBER.
PRINT "Bldg: "L11211_BLD_NUMBER.
PRINT "Room Num: "L11211_RM_NUMBER.
PRINT DEP FORMAT "Deposit Amount: "$3.2 END.
PRINT "Is this correct? (Y/N) "END.
ACCEPT L11211_RESPONSE.
IF L11211_RESPONSE EQ ('Y','y') THEN
  PRINT "The record has been deleted."
  DELETE.
  IF G1121_DELETE_FLAG EQ 'T' THEN
    DBSET NAME.
    FIND ALL.
    FIND SSN EQ G11_SS_NUMBER.
    DELETE.
ENDIF.
ENDIF.
RETURN.
Module Name: 11212 Increase Key Available

Parameters:
  In Only: gl12_key_number

Out Only:

In/Out:

Coded By: Robin FaunLeRoy

Date Last Modified: 7/15/85

Reason Modified:

Module Description:
  This module update the inventory file to reflect the return of a key.

DBSET INVEN.
FIND ALL.
FIND KIN EQ gl12_KEY_NUMBER.
CHANGE QON QON+1.
RETURN.

Module Name: 113 Distribute Info
Parameters:
  In Only: none
  Out Only: none
  In/Out: none
Coded By: Robin FauntLeRoy
Date Last Modified: July 12, 1985
Reason Modified:

Module Description:
The module is the main controlling module for the distribute info section and displays a menu to allow the user to choose what information is wanted. It passes no parameters and calls modules depending upon the user choice.

DEFINE TEXT 1 L113_INFO_MENU_CHOICE.

REPEAT
  REPEAT
    PUSH USING BL
    POP END.
    PRINT "
    **********
    PRINT "*
    **********
    PRINT "*
    **********
    PRINT "*
    **********
    PRINT "*
    **********
    PRINT "*
    **********
    PRINT "*
    PRINT "Enter information topic option:
    A Keyholder Information
    B Key Number Information
    C Building and Room Number Information
    D Information on Keys Held
    E Key Inventory Information
    F Daily Cash Balance
    G Exit
    PRINT "
    **********
    PRINT "
    **********
    PRINT "
    **********
    PRINT "
    **********
    PRINT "
    **********
    PRINT "

PRINT FMT "$ Enter Choice: " END.
ACCEPT L113_INFO_MENU_CHOICE.

IF L113_INFO_MENU_CHOICE NEQ ( 'A', 'a', 'B', 'b', 'C', 'c', 'D', 'd', 'E', 'e', 'F'
  , 'f', 'G', 'g' ) THEN
PRINT "Invalid choice, please try again.".

ENDIF.
UNTIL L113_INFO_MENU_CHOICE EQ ( 'A', 'a', 'B', 'b', 'C', 'c', 'D', 'd', 'E', 'e', 'F', 'f', 'G', 'g').

IF L113_INFO_MENU_CHOICE EQ ( 'A', 'a') THEN
  CALL DISPLAY_OWNER.
ELSEIF L113_INFO_MENU_CHOICE EQ ( 'B', 'b') THEN
  CALL DISPLAY_KEY_NUMBER.
ELSEIF L113_INFO_MENU_CHOICE EQ ( 'C', 'c') THEN
  CALL DISPLAY_LOCATION.
ELSEIF L113_INFO_MENU_CHOICE EQ ( 'D', 'd') THEN
  CALL DISPLAY_HOLDER_KEYS.
ELSEIF L113_INFO_MENU_CHOICE EQ ( 'E', 'e') THEN
  CALL DISPLAY_INVENTORY.
ELSEIF L113_INFO_MENU_CHOICE EQ ( 'F', 'f') THEN
  CALL DISPLAY_DEPOSIT.
ELSE
  PRINT " ".
ENDIF.
UNTIL L113_INFO_MENU_CHOICE EQ ( 'G', 'g').
RETURN.

DISPLAY_OWNER: @R1131_DISPLAY_OWNER.DMC
DISPLAY_KEY_NUMBER: @R1132_DISPLAY_KEY_NUMBER.DMC
DISPLAY_LOCATION: @R1133_DISPLAY_LOCATION.DMC
DISPLAY_INVENTORY: @R1135_DISPLAY_INVENTORY.DMC
DISPLAY_HOLDER: @R1136_DISPLAY_HOLDER.DMC
@
Module Name: 1131 Display Owner

Parameters:

In Only: G11_NAME
G11_SS_NUMBER

Out Only: none

In/Out: none

Coded By: Robin FauntLeRoy

Date Last Modified: 7/12/85

Reason Modified:

Module Description:

This module will display all individual who have a certain key. The user is prompted for either the building and room number for which he wishes information, or the key number, and will then see displayed a list of all those individuals who have been issued that key.

DEFINE TEXT 1 L1131_MENU_CHOICE
TEXT 3 L1131_BLD_NUMBER
TEXT 4 L1131_ROOM_NUMBER
TEXT 1 L1131_RESPONSE
TEXT 10 L1131_KEY
TEXT 1 L1131_WAIT
TEXT 10 L1131_KEY_NUMBER.

DEFINE INTEGER L1131_COUNT
INTEGER L1131_SOC
INTEGER L1131_SS_TEMP
INTEGER L1131_TEMP_CTR
INTEGER L1131_TOTAL.

LET L1131_RESPONSE EQ "N".

REPEAT
  REPEAT
    PUSH USING BL
    POP END.
    **************";
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*",
    PRINT "*
  on: A Building and room number
  B Key numbers
  C Exit

  REPEAT
    PUSH USING BL
    POP END.
    **************";

PRINT "Please enter menu choice. " END.
ACCEPT L1131_MENU_CHOICE.
IF L1131_MENU_CHOICE NEQ ('A', 'a', 'B', 'b', 'C', 'c') THEN
  PRINT "Invalid choice. Please try again."
ENDIF.
UNTIL L1131_MENU_CHOICE EQ ('A', 'a', 'B', 'b', 'C', 'c')
IF L1131_MENU_CHOICE EQ ('A', 'a') THEN
  REPEAT
    REPEAT
      PRINT "Please enter the building abbreviation".
      PRINT "(maximum 3 characters):"
      ACCEPT L1131_BLD_NUMBER.
      PRINT "Please enter the room number"
      PRINT "(maximum 4 characters):"
      PRINT "Room No:" L1131_ROOM_NUMBER.
      PRINT "Bldg:" L1131_BLD_NUMBER.
      PRINT "Room No:" L1131_ROOM_NUMBER.
      PRINT "Is this correct? (Y/N) " END.
      ACCEPT L1131_WAIT.
    UNTIL L1131_WAIT EQ ('Y', 'y')
  DBSET LOCATE.
  FIND ALL.
  SEARCH BLD EQ L1131_BLD_NUMBER.
  SEARCH RN EQ L1131_ROOM_NUMBER.
  IF SYSNREC EQ 0 THEN
    PRINT "You have an invalid building and room number".
  ELSEIF SYSNREC EQ 1 THEN
    MAP IND_RECORD VIA KIN.
    IF SYSNREC EQ 0 THEN
      PRINT "No one has been issued that key number.".
    ELSE
      REPEAT
        GETREC LEAVE.
        LET G11_SS_NUMBER EQ SSN.
        CALL FIND NAME.
      UNTIL SYSNREC EQ 0.
    ENDIF.
  ELSE
    REPEAT
      GETREC LEAVE.
      MAP IND_RECORD VIA KIN.
      REPEAT
        GETREC LEAVE.
        LET G11_SS_NUMBER EQ SSN.
        CALL FIND NAME.
      UNTIL SYSNREC EQ 0.
    UNTIL SYSNREC EQ 0.
  ENDIF.
  PRINT "Do you have another request? (Y/N) " END.
  ACCEPT L1131_RESPONSE.
UNTIL L1131_RESPONSE EQ ('N', 'n')
ELSEIF L1131_MENU_CHOICE EQ ('B', 'b') THEN
  REPEAT
    LET L1131_COUNT EQ 0.
    REPEAT
      LET L1131_COUNT EQ 0.
      PRINT "Please enter the building abbreviation".
      PRINT "(maximum 3 characters):"
      ACCEPT L1131_BLD_NUMBER.
      PRINT "Please enter the room number"
      PRINT "(maximum 4 characters):"
      PRINT "Room No:" L1131_ROOM_NUMBER.
      PRINT "Bldg:" L1131_BLD_NUMBER.
      PRINT "Room No:" L1131_ROOM_NUMBER.
      PRINT "Is this correct? (Y/N) " END.
      ACCEPT L1131_WAIT.
    UNTIL L1131_WAIT EQ ('Y', 'y')
  DBSET LOCATE.
  FIND ALL.
  SEARCH BLD EQ L1131_BLD_NUMBER.
  SEARCH RN EQ L1131_ROOM_NUMBER.
  IF SYSNREC EQ 0 THEN
    PRINT "You have an invalid building and room number".
  ELSEIF SYSNREC EQ 1 THEN
    MAP IND_RECORD VIA KIN.
    IF SYSNREC EQ 0 THEN
      PRINT "No one has been issued that key number.".
    ELSE
      REPEAT
        GETREC LEAVE.
        LET G11_SS_NUMBER EQ SSN.
        CALL FIND NAME.
      UNTIL SYSNREC EQ 0.
    ENDIF.
  ELSE
    REPEAT
      GETREC LEAVE.
      MAP IND_RECORD VIA KIN.
      REPEAT
        GETREC LEAVE.
        LET G11_SS_NUMBER EQ SSN.
        CALL FIND NAME.
      UNTIL SYSNREC EQ 0.
    UNTIL SYSNREC EQ 0.
  ENDIF.
  PRINT "Do you have another request? (Y/N) " END.
  ACCEPT L1131_RESPONSE.
UNTIL L1131_RESPONSE EQ ('N', 'n')
ELSEIF L1131_MENU_CHOICE EQ ('C', 'c') THEN
  REPEAT
    LET L1131_COUNT EQ 0.
    REPEAT
      LET L1131_COUNT EQ 0.
      PRINT "Please enter the building abbreviation".
      PRINT "(maximum 3 characters):"
      ACCEPT L1131_BLD_NUMBER.
      PRINT "Please enter the room number"
      PRINT "(maximum 4 characters):"
      PRINT "Room No:" L1131_ROOM_NUMBER.
      PRINT "Bldg:" L1131_BLD_NUMBER.
      PRINT "Room No:" L1131_ROOM_NUMBER.
      PRINT "Is this correct? (Y/N) " END.
      ACCEPT L1131_WAIT.
    UNTIL L1131_WAIT EQ ('Y', 'y')
  DBSET LOCATE.
  FIND ALL.
  SEARCH BLD EQ L1131_BLD_NUMBER.
  SEARCH RN EQ L1131_ROOM_NUMBER.
  IF SYSNREC EQ 0 THEN
    PRINT "You have an invalid building and room number".
  ELSEIF SYSNREC EQ 1 THEN
    MAP IND_RECORD VIA KIN.
    IF SYSNREC EQ 0 THEN
      PRINT "No one has been issued that key number.".
    ELSE
      REPEAT
        GETREC LEAVE.
        LET G11_SS_NUMBER EQ SSN.
        CALL FIND NAME.
      UNTIL SYSNREC EQ 0.
    ENDIF.
  ELSE
    REPEAT
      GETREC LEAVE.
      MAP IND_RECORD VIA KIN.
      REPEAT
        GETREC LEAVE.
        LET G11_SS_NUMBER EQ SSN.
        CALL FIND NAME.
      UNTIL SYSNREC EQ 0.
    UNTIL SYSNREC EQ 0.
  ENDIF.
PRINT "$ " "Please enter a key number or a 'q' to ".
PRINT "$ " "indicate no more keys. If no names appear ".
PRINT "$ " "then no one has been issued that group of keys.".
PRINT FMT $ "Key Number: " END.
ACCEPT L1131_KEY_NUMBER.
IF L1131_KEY_NUMBER NEQ ('Q', 'q') THEN
    LET L1131_COUNT EQ L1131_COUNT + 1.
    DBSET IND_RECORD.
    FIND KIN EQ L1131_KEY_NUMBER.
    IF SYSNREC GT 0 THEN
        IF L1131_COUNT EQ 1 THEN
            LET L1131_TOTAL EQ SYSNREC.
        ENDIF.
        REPEAT
            GETREC LEAVE.
            LET L1131_SOC EQ SSN.
            LET L1131_KEY EQ KIN.
            DBSET HOLDER.
            FIND ALL.
            ADD SNS L1131_SOC KW L1131_KEY.
        UNTIL SYSNREC EQ 0.
    ENDIF.
ENDIF.
UNTIL L1131_KEY_NUMBER EQ ('Q', 'q').
DBSET HOLDER.
FIND ALL.
IF SYSNREC EQ 0 THEN
    PRINT "$ " "There is no one who has all those keys issued to them".
ELSE
    LET L1131_TEMP_CTR EQ 0.
    REPEAT
        FIND ALL.
        LET L1131_TEMP_CTR EQ L1131_TEMP_CTR + 1.
        GETREC LEAVE L1131_TEMP_CTR.
        LET L1131_SS_TEMP EQ SSN.
        FIND ALL.
        SEARCH SNS EQ L1131_SS_TEMP.
        IF SYSNREC EQ L1131_COUNT THEN
            LET G11_SS_NUMBER EQ L1131_SS_TEMP.
            CALL FIND_NAME.
        ENDIF.
    UNTIL L1131_TEMP_CTR EQ L1131_TOTAL.
ENDIF.
DBSET HOLDER.
FIND ALL.
DELETE.
PRINT FMT $ "Do you have another request? (Y/N) " END.
ACCEPT L1131_RESPONSE.
UNTIL L1131_RESPONSE EQ ('N', 'n').
ELSE
    PRINT "$ " ".
ENDIF.
UNTIL L1131_MENU_CHOICE EQ ('C', 'c').
RETURN.
Module Name: 1132 Display Key Number

Parameters:
In Only: none
Out Only: none
In/Out: none

Coded By: Robin FauntLeRoy
Date Last Modified: 7/12/85

Module Description:
This module will take a building number and room number and displays the keys that will open that door.

DEFINE TEXT 3 L1132_BLD_NUMBER
TEXT 1 L1132_WAIT
TEXT 4 L1132_ROOM_NUMBER.

LET L1132_WAIT EQ "N".
REPEAT
PRINT " ".
PRINT "Please enter the building abbreviation".
PRINT "(maximum 3 characters) ".
ACCEPT L1132_BLD_NUMBER.
PRINT " ".
PRINT "Please enter the room number".
PRINT "(maximum 4 characters) ".
ACCEPT L1132_ROOM_NUMBER.
DBSET LOCATE.
FIND ALL.
SEARCH BLD EQ L1132_BLD_NUMBER.
SEARCH RN EQ L1132_ROOM_NUMBER.
IF SYSNREC NEQ 0 THEN
PRINT " ".
PRINT BLD RN KIN.
ELSE
PRINT " ".
PRINT "That is an invalid building and room number.".
ENDIF.
PRINT " ".
PRINT "Do you have another request? (Y/N) " END.
ACCEPT L1132_WAIT.
UNTIL L1132_WAIT EQ ("N", "n").
RETURN.
"
Module Name: 1133 Display Location

Parameters:

- In Only: none
- Out Only: none
- In/Out: none

Coded By: Robin FauntLeRoy

Date Last Modified: 7/12/85

Reason Modified:

Module Description:

This module, given a key number will display the location of the building and room that that key opens.

```
DEFINE TEXT 10 L1133_KEY_NUMBER
TEXT 1 L1133_WAIT.

LET L1133_WAIT EQ "N".

REPEAT
  PRINT ".
  PRINT "Please enter the key number.".
  PRINT FMT $ "Key Number: "END.
  ACCEPT L1133_KEY_NUMBER.
  DBSET LOCATE.
  FIND ALL.
  FIND KIN EQ L1133_KEY_NUMBER.
  IF SYSNREC NEQ 0 THEN
    PRINT ".
    PRINT KIN BLD RN.
  ELSE
    PRINT ".
    PRINT "That is an invalid key number.".
  ENDIF.
  PRINT ".
  PRINT FMT $ "Do you have another request? (Y/N) "END.
  ACCEPT L1133_WAIT.
UNTIL L1133_WAIT EQ ['N','n'].
RETURN.
```
Module Name: 1134 Display Deposit

Parameters:
  In Only: none
  Out Only: none
  In/Out: none

Coded By: Robin FauntLeRoy

Date Last Modified: 7/12/85

Reason Modified:

Module Description:

This module will print out the total of the amount in the
checkbook file. This file maintains a running total of the
deposit amount. It also provides the user with a method of zero'ing
out the total to start another days total.

DEFINE TEXT 1 L1134_CLEAR
  TEXT 1 L1134_WAIT.

DBSET CHECKBOOK.

FIND ALL.
PRINT " ".
PRINT "Deposit Total".
PRINT BAL*100 FORMAT 3X 155..2 END.
PRINT " ".
PRINT FMT $ "Do you wish to have the total cleared? (Y/N) " END.
ACCEPT L1134_CLEAR.
IF L1134_CLEAR EQ ('Y','y') THEN
  CHANGE BAL 0.
  PRINT "Deposit total is now $0.00.".
ENDIF.
PRINT " ".
PRINT FMT $ "Type a carriage return <cr> to continue " END.
ACCEPT L1134_WAIT.
RETURN.

&
Module Name: 1135 Display Inventory

Parameters:

In Only: none
Out Only: none
In/Out: none

Coded By: Robin FauntLeRoy
Date Last Modified: 7/12/85
Reason Modified:

Module Description:
This module will prompt the user for the key number for which the inventory information is requested. It will then print the information to the screen.

DEFINE TEXT 10 L1135_KEY_NUMBER
TEXT 1 L1135_WAIT.

DEFINE INTEGER L1135_TOTAL_OUT.

LET L1135_WAIT EQ "N".
REPEAT

PRINT "Please enter the Key Number.",
PRINT FMT $ "Key Number: "END.
ACCEPT L1135_KEY_NUMBER.
DBSET INVEN.
FIND ALL.
FIND KIN EQ L1135_KEY_NUMBER.
IF SYSNREC EQ 1 THEN
   LET L1135_TOTAL_OUT EQ TN-QON.
   PRINT "
   PRINT "KEY NUMBER QUANTITY ON HAND TOTAL OUT TOTAL N
   PRINT "___________ _________ -------
   PRINT "
   PRINT KIN QON L1135_TOTAL_OUT TN FORMAT I10 10X I3 16X I3 12X I3 END.
   PRINT "
ELSE
   PRINT "Invalid key number given.",
ENDIF.
PRINT "", ACCEPT L1135_WAIT.
UNTIL L1135_WAIT EQ ['n', 'N'].
RETURN.
Module Name: 1136 Display Holder

Parameters:

In Only:

Out Only:

In/Out: G11_SS_Number

G11_Last_Name

G11_First_Name

Coded By: Robin FauntLeRoy

Date Last Modified: 7/16/85

Reason Modified:

Module Description:

This module will determine which individual the user is requesting the information for, and then display all keys that that user has issued to him.

DEFINE TEXT 9 L1136_RESPONSE
TEXT 1 L1136_AGAIN
TEXT 1 L1136_ANSWER.

LET L1136_AGAIN EQ "N".
REPEAT
LET G11_SS_NUMBER EQ 0.
PRINT "Please enter the individual's SSN. (9 digits)."
PRINT FMT $ " or a carriage return <cr> if none " END.
ACCEPT L1136_RESPONSE.
IF $INT(L1136_RESPONSE) NEQ 0 THEN
LET G11_SS_NUMBER EQ $INT(L1136_RESPONSE).
CALL FIND_NAME.
ELSE
REPEAT
PRINT "Please enter the individual's last name."
PRINT FMT $ "Last Name: " END.
ACCEPT G11_LAST_NAME.
PRINT "Please enter the individual's first name."
PRINT FMT $ "First Name: " END.
ACCEPT G11_FIRST_NAME.
PRINT "Last Name: " G11_LAST_NAME.
PRINT "First Name: " G11_FIRST_NAME.
PRINT FMT $ "Is this correct? (Y/N) " END.
ACCEPT L1136_ANSWER.
UNTIL L1136_ANSWER EQ ('Y', 'y').
CALL FIND_SSN.
ENDIF.
IF G11_SS_NUMBER NEQ 0 THEN
DBSET IND_RECORD.
FIND ALL.
FIND SSN EQ G11_SS_NUMBER.
IF SYSNREC NEQ 0 THEN
PRINT "Key Number Date Out Deposit Amount".
PRINT FMT A 6X D9 5X F$3.2 END.
ELSE

PRINT " ".
PRINT "That individual does not have any keys issued to
him.".
ENDIF.
ELSE
PRINT " ".
PRINT "That individual does not have any keys issued to him.".
ENDIF.
PRINT ".
PRINT FMT $ "Do you have another request? (Y/N) " END.
ACCEPT L1136_AGAIN.
UNTIL L1136_AGAIN EQ ('n', 'N').
RETURN.

$
Module Name: 114 Gen Corres
Parameters:
  In Only: none
  Out Only: G1l SS Number
  In/Out: G1l_Name
          G1l_Last_Name
          G1l_First_Name
Coded By: Robin FauntLeRoy
Date Last Modified: 7/12/85
Reason Modified:
Module Description:
This module is the controlling module for the correspondence section of the system.

DEFINE TEXT 1 L114_CHOICE.
REPEAT
  PUSH USING BL
  POP END.
  PRINT "\n  PRINT "The following types of correspondence are available:"
  PRINT ","
  PRINT "L: Letter"
  PRINT "R: Report"
  PRINT "Q: Quit"
  PRINT 
  PRINT FMT $ "Please enter choice: "END.
  ACCEPT L114_CHOICE.
  IF L114_CHOICE EQ ( 'L', 'l' ) THEN
    CALL GENERATE_LETTERS.
  ELSEIF L114_CHOICE EQ ( 'R', 'r' ) THEN
    CALL GENERATE_REPORTS.
  ELSEIF L114_CHOICE EQ ( 'Q', 'q' ) THEN
    PRINT "\n    ELSE
    PRINT "Invalid choice, please try again."
    ENDIF.
UNTIL L114_CHOICE EQ ( 'Q', 'q' ).
RETURN.

GENERATE_LETTERS: @R1141_GEN_LETTERS.DMC
GENERATE_REPORTS: @R1142_GEN_REPORTS.DMC
Module Name: 1141 Gen Letters

Parameters:
- In Only: none
- Out Only: none
- In/Out: Gll_SS_number
  Gll_Name
  Gll_First_Name
  Gll_Last_Name

Coded By: Robin FauntLeRoy
Date Last Modified: 7/12/85
Reason Modified:

Module Description:
This module determines whether the user wants to generate student letters or personnel letters and then calls the appropriate modules.

DEFINE TEXT 1 L1141_MENU_CHOICE.

REPEAT
  PRINT " ".
  PRINT "The following types of letters are available: ".
  PRINT " S: Student".
  PRINT " P: Personnel".
  PRINT " Q: Quit".
  PRINT ":
  PRINT FMT "$ "Please enter choice: ".END.
  ACCEPT L1141_MENU_CHOICE.
  IF L1141_MENU_CHOICE EQ ( 'S', 's' ) THEN
    CALL GEN_STUDENT_LETTERS.
    CALL PRINT_LETTERS.
  ELSEIF L1141_MENU_CHOICE EQ ( 'P', 'p' ) THEN
    CALL GEN_PERSONNEL_LETTERS.
    CALL PRINT_LETTERS.
  ELSEIF L1141_MENU_CHOICE EQ ( 'Q', 'q' ) THEN
    PRINT "Leaving letter generation ".
  ELSE
    PRINT "Invalid choice, please try again. ".
  ENDIF.
  UNTIL L1141_MENU_CHOICE EQ ( 'Q', 'q' ).
RETURN.

GEN_STUDENT_LETTERS: @R11411_GEN_STU_LETTERS.DMC
GEN_PERSONNEL_LETTERS: @R11412_GEN_PER_LETTERS.DMC
PRINT_LETTERS: @R11413_PRINT_LETTERS.DMC
Module Name: 11411_gen_per_letters
Parameters:
  In Only:
  Out Only:
  In/Out:
Coded By: Robin FauntLeRoy
Date Last Modified: 8/04/85
Reason Modified:

Module Description:
This module will determine which students are intending to graduate during the quarter indicated and have keys issued to them. It will generate a letter to each one, requesting the return of the keys which have been issued to each.

DEFINE TEXT 20 L11411_ADD
  TEXT 1 L11411_RESPONSE
  TEXT 1 L11411_FOUND
  TEXT 10 L11411_CITY
  TEXT 3 L11411_SEARCH
  TEXT 1 L11411_WAIT
  TEXT 30 L11411_NAME
  TEXT 2 L11411_STATE
  TEXT 6 L11411_ZIP.

DEFINE INTEGER L11411_QUARTER
  INTEGER L11411_SSN.

PRINT "This process may take some time."
PRINT FMT $ "Do you wish to continue? (Y/N) " END.
ACCEPT L11411_RESPONSE.
IF L11411_RESPONSE EQ ('Y', 'y') THEN REPEAT
  PUSH USING 8L
  POP END.
  PRINT "Please choose the quarter you would like your letters of graduation based upon."
  PRINT FMT $ "Quarter: " END.
  ACCEPT L11412_QUARTER.
  UNTIL L11412_QUARTER LE 4 AND L11412_QUARTER GT 0.
  LET L11411_FOUND EQ "F".
  OBSET GRAD.
  FIND ALL.
  LET L11411_SEARCH EQ ($RIGHT($TEXTR($YEAR(SYSDATE)),2) + $RIGHT($TEXTR(L11412_QUARTER),1)).
  FIND EQG_DATE EQ L11411_SEARCH.
  IF SYSREC EQ 0 THEN
    PRINT "".
  PRINT "This process may take some time."
  PRINT FMT $ "Do you wish to continue? (Y/N) " END.
  ACCEPT L11411_RESPONSE.
  IF L11411_RESPONSE EQ ('Y', 'y') THEN REPEAT
  PUSH USING 8L
  POP END.
  PRINT "Please choose the quarter you would like your letters of graduation based upon."
  PRINT FMT $ "Quarter: " END.
  ACCEPT L11412_QUARTER.
  UNTIL L11412_QUARTER LE 4 AND L11412_QUARTER GT 0.
  LET L11411_FOUND EQ "F".
  OBSET GRAD.
  FIND ALL.
  LET L11411_SEARCH EQ ($RIGHT($TEXTR($YEAR(SYSDATE)),2) + $RIGHT($TEXTR(L11412_QUARTER),1)).
  FIND EQG_DATE EQ L11411_SEARCH.
  IF SYSREC EQ 0 THEN
    PRINT "".
  PRINT "This process may take some time."
  PRINT FMT $ "Do you wish to continue? (Y/N) " END.
  ACCEPT L11411_RESPONSE.
  IF L11411_RESPONSE EQ ('Y', 'y') THEN REPEAT
  PUSH USING 8L
  POP END.
  PRINT "Please choose the quarter you would like your letters of graduation based upon."
  PRINT FMT $ "Quarter: " END.
  ACCEPT L11412_QUARTER.
  UNTIL L11412_QUARTER LE 4 AND L11412_QUARTER GT 0.
  LET L11411_FOUND EQ "F".
  OBSET GRAD.
  FIND ALL.
  LET L11411_SEARCH EQ ($RIGHT($TEXTR($YEAR(SYSDATE)),2) + $RIGHT($TEXTR(L11412_QUARTER),1)).
  FIND EQG_DATE EQ L11411_SEARCH.
  IF SYSREC EQ 0 THEN
    PRINT "".
  PRINT "This process may take some time."
  PRINT FMT $ "Do you wish to continue? (Y/N) " END.
  ACCEPT L11411_RESPONSE.
  IF L11411_RESPONSE EQ ('Y', 'y') THEN REPEAT
  PUSH USING 8L
  POP END.
  PRINT "Please choose the quarter you would like your letters of graduation based upon."
  PRINT FMT $ "Quarter: " END.
  ACCEPT L11412_QUARTER.
  UNTIL L11412_QUARTER LE 4 AND L11412_QUARTER GT 0.
  LET L11411_FOUND EQ "F".
  OBSET GRAD.
  FIND ALL.
  LET L11411_SEARCH EQ ($RIGHT($TEXTR($YEAR(SYSDATE)),2) + $RIGHT($TEXTR(L11412_QUARTER),1)).
  FIND EQG_DATE EQ L11411_SEARCH.
  IF SYSREC EQ 0 THEN
    PRINT "".
  PRINT "This process may take some time."
  PRINT FMT $ "Do you wish to continue? (Y/N) " END.
  ACCEPT L11411_RESPONSE.
  IF L11411_RESPONSE EQ ('Y', 'y') THEN REPEAT
  PUSH USING 8L
  POP END.
  PRINT "Please choose the quarter you would like your letters of graduation based upon."
  PRINT FMT $ "Quarter: " END.
  ACCEPT L11412_QUARTER.
  UNTIL L11412_QUARTER LE 4 AND L11412_QUARTER GT 0.
  LET L11411_FOUND EQ "F".
  OBSET GRAD.
  FIND ALL.
  LET L11411_SEARCH EQ ($RIGHT($TEXTR($YEAR(SYSDATE)),2) + $RIGHT($TEXTR(L11412_QUARTER),1)).
  FIND EQG_DATE EQ L11411_SEARCH.
  IF SYSREC EQ 0 THEN
    PRINT "".
  PRINT "This process may take some time."
  PRINT FMT $ "Do you wish to continue? (Y/N) " END.
  ACCEPT L11411_RESPONSE.
  IF L11411_RESPONSE EQ ('Y', 'y') THEN REPEAT
  PUSH USING 8L
  POP END.
  PRINT "Please choose the quarter you would like your letters of graduation based upon."
  PRINT FMT $ "Quarter: " END.
  ACCEPT L11412_QUARTER.
  UNTIL L11412_QUARTER LE 4 AND L11412_QUARTER GT 0.
  LET L11411_FOUND EQ "F".
  OBSET GRAD.
  FIND ALL.
  LET L11411_SEARCH EQ ($RIGHT($TEXTR($YEAR(SYSDATE)),2) + $RIGHT($TEXTR(L11412_QUARTER),1)).
  FIND EQG_DATE EQ L11411_SEARCH.
  IF SYSREC EQ 0 THEN
    PRINT "".
  PRINT "This process may take some time."
  PRINT FMT $ "Do you wish to continue? (Y/N) " END.
  ACCEPT L11411_RESPONSE.
  IF L11411_RESPONSE EQ ('Y', 'y') THEN REPEAT
  PUSH USING 8L
  POP END.
  PRINT "Please choose the quarter you would like your letters of graduation based upon."
  PRINT FMT $ "Quarter: " END.
  ACCEPT L11412_QUARTER.
  UNTIL L11412_QUARTER LE 4 AND L11412_QUARTER GT 0.
  LET L11411_FOUND EQ "F".
  OBSET GRAD.
  FIND ALL.
  LET L11411_SEARCH EQ ($RIGHT($TEXTR($YEAR(SYSDATE)),2) + $RIGHT($TEXTR(L11412_QUARTER),1)).
  FIND EQG_DATE EQ L11411_SEARCH.
  IF SYSREC EQ 0 THEN
    PRINT "".
  PRINT "This process may take some time."
  PRINT FMT $ "Do you wish to continue? (Y/N) " END.
  ACCEPT L11411_RESPONSE.
  IF L11411_RESPONSE EQ ('Y', 'y') THEN REPEAT
  PUSH USING 8L
  POP END.
  PRINT "Please choose the quarter you would like your letters of graduation based upon."
  PRINT FMT $ "Quarter: " END.
  ACCEPT L11412_QUARTER.
  UNTIL L11412_QUARTER LE 4 AND L11412_QUARTER GT 0.
  LET L11411_FOUND EQ "F".
  OBSET GRAD.
  FIND ALL.
  LET L11411_SEARCH EQ ($RIGHT($TEXTR($YEAR(SYSDATE)),2) + $RIGHT($TEXTR(L11412_QUARTER),1)).
  FIND EQG_DATE EQ L11411_SEARCH.
  IF SYSREC EQ 0 THEN
    PRINT "".
  PRINT "This process may take some time."
  PRINT FMT $ "Do you wish to continue? (Y/N) " END.
  ACCEPT L11411_RESPONSE.
  IF L11411_RESPONSE EQ ('Y', 'y') THEN REPEAT
  PUSH USING 8L
  POP END.
ELSE

INIT 7 STUD.LST.
REPEAT.
GETREC LEAVE.
LET L11411_NAME EQ NAME.
LET L11411_ADD EQ L_STR.
LET L11411_CITY EQ L_CITY.
LET L11411_STATE EQ L_STATE.
LET L11411_ZIP EQ L_ZIP.
LET L11411_SSN EQ $INT(SSNO).
DBSET IND_RECORD.
FIND ALL.
PRINT "IN IND_REC".
PRINT ALL.

IF SYSNREC GT 0 THEN
  LET L11411_FOUND EQ "T".
  PRINT ON 7 FMT /// END.
  PRINT ON 7 L11411_NAME.
  PRINT ON 7 L11411_ADD.
  PRINT ON 7 L11411_CITY L11411_STATE L11411_ZIP.
  PRINT ON 7 SYSDATE FMT /// D2 /// END.
  PRINT ON 7 FMT / "Dear Student/University Employee," / E
  PRINT ON 7 E
  PRINT ON 7 "It has come to our attention "END.
  PRINT ON 7 "that you will soon be leaving ".
  PRINT ON 7 FMT $ "the University system, and that " END.
  PRINT ON 7 "you still possess one or ".
  PRINT ON 7 FMT $ "more keys from the Physical Plant. " E
  PRINT ON 7 "Please return these to ".
  PRINT ON 7 FMT "$ the Physical Plant prior to "END.
  PRINT ON 7 "leaving and your deposit will ".
  PRINT ON 7 FMT $ "be refunded. The keys we " END.
  PRINT ON 7 FMT "show that you have are:" / / END.
  PRINT ON 7 FMT 10X A10 END.
  PRINT ON 7 FMT // "University of Montana Physical Plant"
/ 1 END.
  ENDIF.
  UNTIL SYSNREC EQ 0.
  IF L11411_FOUND EQ "F" THEN
    PRINT "", E
    PRINT "None of the individuals who were terminating had keys".
  ELSE
    PRINT "", E
    PRINT "The letters have been generated and can be picked up ".
    PRINT "at the Computer Center tomorrow.".
  ENDDIF.
ENDIF.
PRINT ".
PRINT FMT "$ Type a carriage return <cr> to continue: "END.
ACCEPT L11411_WAIT.
ENDIF.
RETURN.
This module will determine those individuals employed for the university, who are terminated and have keys issued to them. This information will be placed in a list to be printed.

```
DEFINE TEXT 20 L11412_LOAD
TEXT 1 L11412_FOUND
TEXT 20 L11412_WAIT
TEXT 30 L11412_NAME
TEXT 5 L11412_STATE
TEXT 6 L11412_ZIP.

LET L11412_FOUND EQ "F".
FIND ALL.
SEARCH THAT GT $TEXTR(000000) AND THAT LE $TEXTR(SYSDATE).
IF SYSREC EQ 0 THEN
  PRINT "There is no one terminated lately."
ELSE
  INIT 7 LET RST.
  REPEAT.
  GETREC LEAVE.
  LET L11412_NAME EQ $TRIM(FMNAM) + " " + $TRIM(LNAM).
  LET L11412_ADDR EQ ADDR1.
  LET L11412_ADDR EQ ADDR2.
  LET L11412_MEMBERSHIP EQ MMB.
  LET L11412_ZIP EQ ZIP.
  ID TO IND.lastname via SSN.
  IF SYSREC EQ 0 THEN
    LET L11412_FOUND EQ "T".
    PRINT ON 7 FMT "Dear Student/University Employee,"
    PRINT ON 7 "It has come to our attention that you will soon be leaving."
    PRINT ON 7 "You still possess one or more keys from the Physical Plant."
    PRINT ON 7 "Please return these to:"
    PRINT ON 7 "the Physical Plant prior to END."
    PRINT ON 7 "Ending your enrollment will end your obligation to return these keys."
    PRINT ON 7 "The keys we have are:" END.
    PRINT ON 7 "at the Computer Center tomorrow.".
  END.
  PRINT ON 7 FMT "Thank you for your cooperation."
  PRINT ON 7 "University of Montana Physical Plant"
  PRINT ON 7 "Cl END.
ENDIF.
UNTIL SYSREC EQ 0.
IF L11412_FOUND EQ "F" THEN
  PRINT "None of the individuals who were terminating had keys".
ELSE
  PRINT "The letters have been generated and can be picked up".
END.
ENDIF.
RETURN.
```
Module Name: l142 Gen Reports

Parameters:
In Only: none
Out Only: GIR report
In/Out: GIR report

Coded By: Robin FauntLeRoy
Date Last Modified: 7/26/85
Reason Modified:

Module Description:
This module sets the gl132 report flag, which determines whether or not the name found in Find_Name is printed out, and then determines which report the user wants.

DEFINE TEXT 1 L1142_RESPONSE.

REPEAT
  LET G1_REPORT EQ "T".
  PUSH USING BL
  POP END.

************
**
** REPORT GENERATION MENU
**
**
** Enter report option:
**
** A Keyholder Report
**
** B Inventory Report
**
** C Location Report
**
** D Key Retirement Report
**
** E Exit
**
**

******
PRINT ".
PRINT FMT $ "Please enter option choice: "END.
ACCEPT L1142 RESPONSE.
THEN
  IF L1142_RESPONSE NEQ ('A', 'a', 'B', 'b', 'C', 'c', 'D', 'd', 'E', 'e')
  PRINT "Invalid choice, please try again."
ENDIF.
UNTIL L1142_RESPONSE EQ ('A', 'a', 'B', 'b', 'C', 'c', 'D', 'd', 'E', 'e').
IF L1142_RESPONSE EQ ('A', 'a') THEN
  CALL KEYHOLDER_REPORT.
END
!PUSH USING PRINT INVEN.TXT.
!POP END.
ELSEIF L1142_RESPONSE EQ ('B', 'b') THEN
CALL INVENTORY_REPORT.
ELSEIF L1142_RESPONSE EQ ('C', 'c') THEN
CALL LOCATION_REPORT.
ELSEIF L1142_RESPONSE EQ ('D', 'd') THEN
CALL RETIREMENT_REPORT.
ELSE
PRINT " ".
ENDIF.
UNTIL L1142_RESPONSE EQ ('E', 'e').
RETURN.
KEYHOLDER_REPORT: @R11421_KEY_REPT.DMC
INVENTORY_REPORT: @R11422_INVEN_REPT.DMC
LOCATION_REPORT: @R11423_LOCAT_REPT.DMC
RETIREMENT_REPORT: @R11424 RETIRE_REPT.DMC
Module Name: 11421 Key Rept
Parameters:
In Only: none
Out Only: none
In/Out: Gl_Report
Coded By: Robin Fauntleroy
Date Last Modified: 7/26/85
Reason Modified:

Module Description:
This module determines the extent of the keyholder report requested by the user, and sets up the information for the report. It also sets the report flag, so that names are not printed out when Find_Name is called.

DEFINE TEXT 1 L11421 RESPONSE
TEXT 10 L11421 NUM
TEXT 1 L11421_WAIT
TEXT 4 L11421_KEY.
REPEAT
PRINT " ".
PRINT "Please choose one of the following: ".
PRINT "  T: Report on Total Keyholders".
PRINT "  I: Individual Report".
PRINT "  (report by individual keyways)".
PRINT "  Q: Quit ".
PRINT " ".
PRINT FMT $ "Please enter your choice: ":END.
ACCEPT L11421 RESPONSE.
UNTIL L11421 RESPONSE EQ ( 'T', 't', 'I', 'i', 'Q', 'q' ) .
IF L11421 RESPONSE EQ ( 'T', 't' ) THEN
FIND ALL.
SORT BY SSN.
ELSE
REPEAT
PRINT " ".
PRINT "Please enter keyway number for report ".
PRINT FMT $ "Keyway Number: ":END.
ACCEPT L11421_KEY.
FIND ALL.
FIND KIN CT STRIM ( L11421_KEY ).
IF SYSNREC EQ 0 THEN
PRINT "Invalid Keyway Number.".
ENDIF.
SORT BY SSN.
ENDIF.
UNTIL SYSNREC GT 0.
SORT SSN.
ENDIF.
IF SYSNREC GT 0 THEN
REPORT START.
SECTION INITIAL.
INIT 5 KEYHOLDER.TXT.
HEADING ON 5 PRINT SYSDATE FMT // // 30T "KEYHOLDER REPORT" 60T
D3 ///END.
FOOTING ON 5 2 PRINT SYSPAGE FORMAT 40T 12..0 END.
PAGE 60.
BODY BU.
SECTION GETREC.
  LET GL1_SS_NUMBER EQ SSN.
  IF GL1_SS_NUMBER LT 0 THEN
    LET L11421_NUM EQ "DUMMY SSN:"
  ELSE
    LET L11421_NUM EQ STEXTL(GL1_SS_NUMBER).
  ENDIF.
  LET GL1_REPORT EQ "T".
  CALL FIND_NAME.
  DBSET IND_RECORD.
SECTION HEADING.
ON CHANGE SSN PRINT ON 5 L11421_NUM GL1_NAME FORMAT / A10 15T A3
0 END.
SECTION PRINT.
  PRINT ON 5 KIN DO DEF FORMAT 10T A10 25T D3 40T FS3.2 END.
SECTION TOTALS.
SECTION FINAL.
REPORT END.
PRINT " ".
PRINT "Your report has been sent to the line printer".
PRINT "and can be picked up at the computer center".
PRINT FMT 5 "Type a carriage return <cr> to continue: " END.
ACCEPT L11421_WAIT.
RELEASE 5.
ENDIF.
ENDIP.
RETURN.
@
Module Name: 11422 Inven Rept
Parameters:
- In Only: none
- Out Only: none
- In/Out: none
Coded By: Robin Fauntleroy
Date Last Modified: 7/26/85
Reason Modified:
Module Description:
This module determines the extent of the inventory report requested by the user, and sets up the information for the report.

DEFINE TEXT L11422_RESPONSE
TEXT L11422_WAIT
TEXT 4 L11422_KEY.
REPEAT
PRINT ".
PRINT "Please choose one of the following: ".
PRINT " T: Report on Total Inventory"
PRINT " I: Individual Report (report by individual keyways)"
PRINT " Q: Quit ".
PRINT FMT $ "Please enter your choice: "END.
ACCEPT L11422_RESPONSE.
UNTIL L11422_RESPONSE EQ ( ' T ' , ' t ' , ' I ' , ' i ' , ' Q ' , ' q ' )
IF L11422_RESPONSE NEQ ( ' Q ' , ' q ' ) THEN
DBSET INVEN.
IF L11422_RESPONSE EQ ( ' T ' , ' t ' ) THEN
FIND ALL.
SORT BY KIN.
ELSE
REPEAT
PRINT " ".
PRINT "Please enter keyway number for report."
PRINT FMT $ "Keyway Number: "END.
ACCEPT L11422_KEY.
FIND ALL.
FIND KIN CT STRIM(L11422_KEY).
IF SYSNREC EQ 0 THEN
PRINT "Invalid Keyway Number.".
ENDIF.
UNTIL SYSNREC GT 0.
SORT KIN.
ENDIF.
IF SYSNREC GT 0 THEN
REPORT START.
INIT 2 INVEN.TXT.
HEADING ON 2 PRINT SYSDATE FMT ///// 30T "INVENTORY REPORT" 60T D3 //END.
FOOTING ON 2 PRINT SYSPAGE FORMAT 40T 12..0 END.
PAGE 60.
BODY 60.
ON START PRINT ON 2 FMT 5T "KEY NUMBER:" 20T "QUANTITY ON HAND:"
40T "QUANTITY OUT:" 55T "TOTAL NUMBER:" /END.
ON START PRINT ON 2 FMT 5T "QUANTITY ON HAND:" 20T "QUANTITY ON HAND:" 40T "QUANTITY OUT:" 55T "TOTAL NUMBER:" /END.
PRINT ON 2 KIN QON TN QON TN PNT 7T A10 27T I3 45T 15 BUT I3 END

REPORT END.
PRINT "\nPRINT "Your report has been sent to the line printer".
PRINT "and can be picked up at the computer center".
PRINT PNT S "Type a carriage return <cr> to continue: "END.
ACCEPT L11422_WAIT.
RELEASE 2.

ENDIF.
RETURN.
"
Module Name: 11423 Locat Rept
Parameters:
  In Only: none
  Out Only: none
  In/Out: none
Coded By: Robin Fauntleroy
Date Last Modified: 7/26/85
Reason Modified:

Module Description:
This module determines the extent of the location report requested by the user, and sets up the information for the report.

DEFINE TEXT 1 L11423_RESPONSE
TEXT 1 L11423_WAIT
TEXT 1 L11423_SORT
TEXT 3 L11423_BLDG
TEXT 4 L11423_KEY.
REPEAT
  PRINT " ".
  PRINT "Please choose one of the following: ".
  PRINT " T: Total Location Report ".
  PRINT "  (sorted by either keyway number or building number)".
  PRINT " I: Individual Report ".
  PRINT "  (report by individual keyways or building numbers) ".
  PRINT " Q: Quit ".
  PRINT " ".
  PRINT FMT $ "Please enter your choice: "END.
  ACCEPT L11423_RESPONSE.
UNTIL L11423_RESPONSE EQ ('T', 't', 'I', 'i', 'Q', 'q').
IF L11423_RESPONSE NEQ ('Q', 'q') THEN
  DBSET LOCATE.
  IF L11423_RESPONSE EQ ('T', 't') THEN
    REPEAT
      PRINT " ".
      PRINT "Would you like the report sorted by Keyway Number".
      PRINT FMT S "or Building Number? (K/B): "END.
      ACCEPT L11423_SORT.
      UNTIL L11423_SORT EQ (K, 'K', B, 'B').
      IF L11423_SORT EQ ('K', 'k') THEN
        FIND ALL.
        SORT BY KIN.
      ELSE
        FIND ALL.
        SORT BY BLD RN.
      ENDIF.
    ENDIF.
  ELSEIF L11423_RESPONSE EQ ('I', 'i') THEN
    REPEAT
      PRINT " ".
      PRINT "Would you like a report based upon a Keyway Number".
      PRINT FMT S "or Building Number? (K/B): "END.
      ACCEPT L11423_SORT.
      UNTIL L11423_SORT EQ (K, 'K', B, 'B').
      IF L11423_SORT EQ ('K', 'k') THEN
        REPEAT
          PRINT " ".
          PRINT "Please enter keyway number for report. ".
          PRINT FMT $ "Keyway Number: "END.
        ENDIF.
      ELSE
        REPEAT
          PRINT " ".
          PRINT "Please enter building number for report. ".
          PRINT FMT $ "Building Number: "END.
        ENDIF.
      ELSE
        PRINT " ".
        PRINT "Invalid choice. Please try again. ".
        PRINT FMT $ "Please enter your choice: "END.
        ACCEPT L11423_RESPONSE.
      ENDIF.
    ENDIF.
  ELSE
    PRINT " ".
    PRINT "You have chosen to quit. Program will end. ".
    PRINT FMT $ "Would you like to continue? (Y/N): "END.
    ACCEPT L11423_CONTINUE.
    IF L11423_CONTINUE EQ ('N', 'n') THEN
      END.
    ELSEIF L11423_CONTINUE EQ ('Y', 'y') THEN
      REPEAT
        PRINT " ".
        PRINT "Please enter your choice: ".
        PRINT FMT $ "Please enter your choice: "END.
        ACCEPT L11423_RESPONSE.
        UNTIL L11423_RESPONSE EQ ('Q', 'q').
      ENDIF.
    ELSE
      PRINT " ".
      PRINT "Invalid choice. Please try again. ".
      PRINT FMT $ "Please enter your choice: "END.
      ACCEPT L11423_RESPONSE.
    ENDIF.
  ENDIF.
END.
ELSE
  PRINT " ".
  PRINT "Invalid choice. Please try again. ".
  PRINT FMT $ "Please enter your choice: "END.
  ACCEPT L11423_RESPONSE.
ENDIF.
ELSEIF L11423_RESPONSE EQ ('Q', 'q') THEN
  REPEAT
    PRINT " ".
    PRINT "You have chosen to quit. Program will end. ".
    PRINT FMT $ "Would you like to continue? (Y/N): "END.
    ACCEPT L11423_CONTINUE.
    IF L11423_CONTINUE EQ ('N', 'n') THEN
      END.
    ELSEIF L11423_CONTINUE EQ ('Y', 'y') THEN
      REPEAT
        PRINT " ".
        PRINT "Please enter your choice: ".
        PRINT FMT $ "Please enter your choice: "END.
        ACCEPT L11423_RESPONSE.
        UNTIL L11423_RESPONSE EQ ('Q', 'q').
      ENDIF.
    ELSE
      PRINT " ".
      PRINT "Invalid choice. Please try again. ".
      PRINT FMT $ "Please enter your choice: "END.
      ACCEPT L11423_RESPONSE.
    ENDIF.
  ENDIF.
ELSE
  PRINT " ".
  PRINT "Invalid choice. Please try again. ".
  PRINT FMT $ "Please enter your choice: "END.
  ACCEPT L11423_RESPONSE.
ENDIF.
ELSE
  PRINT " ".
  PRINT "Invalid choice. Please try again. ".
  PRINT FMT $ "Please enter your choice: "END.
  ACCEPT L11423_RESPONSE.
ENDIF.
ACCEPT L11423_KEY.
FIND ALL.
FIND KIN CT STRIM(L11423_KEY).
IF SYSNREC EQ 0 THEN
  PRINT "Invalid Keyway Number."
ENDIF.
UNTIL SYSNREC GT 0.
SORT KIN.
ELSE
  REPEAT
  PRINT "Please enter building abbreviation for report."
  PRINT FMT $ "Building Number: "
  ACCEPT L11423_BLDG.
  FIND ALL.
  SEARCH BLD EQ L11423_BLDG.
  IF SYSNREC EQ 0 THEN
    PRINT "Invalid building abbreviation."
  ENDIF.
  UNTIL SYSNREC GT 0.
  ENDIF.
ELSE
  PRINT " ".
ENDIF.
ENDIF.
IF SYSNREC GT 0 THEN
  REPORT START.
  INIT 3 LOCAT.TXT.
  HEADING ON 3 PRINT SYSDATE FMT //:/// 33T "LOCATION REPORT" 60T 0
  FOOTING ON 3 2 PRINT SYSPAGE FORMAT 40T 12..0 END.
  BODY 60.
  ON START PRINT ON 3 FMT 5T "KEY NUMBER:" 20T "BUILDING NUMBER:" 40T "ROOM NUMBER:" END.
  ON START PRINT ON 3 FMT 5T "__________" 20T "________________" 40T "_____________" /END.
  PRINT ON 3 KIN BLD RN FMT 7T A10 26T A3 45T A4 END.
  REPORT END.
  PRINT " ".
  PRINT "Your report has been sent to the line printer."
  PRINT "and can be picked up at the computer center."
  PRINT FMT $ "Type a carriage return <cr> to continue: 
  ACCEPT L11423_WAIT.
  RELEASE 3.
ENDIF.
RETURN.
Module Name: 11424 Retire Rept
Parameters:
  In Only: none
  Out Only: none
  In/Out: none
Coded By: Robin Fauntleroy
Date Last Modified: 7/26/85
Reason Modified:

Module Description:
This module prints a report consisting of a list of all keys that have been retired.

DEFINE TEXT 1 L11424_WAIT.
DBSET INVEN.
FIND ALL.
SEARCH ST EQ "R".
IF SYSNREC EQ 0 THEN
  PRINT "There are no retired keys at this time.".
ELSE
  SORT KIN.
  REPORT START.
  INIT 4 RETIRE.TXT.
  HEADING ON 4 PRINT SYSDATE FMT ///// 30T "RETIRED KEY REPORT" 60T D3 ///
  FOOTING ON 4 PRINT SYSPAGE FORMAT 40T 12..0 END.
  ON START PRINT ON 4 FMT IT "KEY NUMBER:" 15T "QUANTITY ON HAND:" 34T "QUANTITY OUT:" 49T "TOTAL NUMBER:" 65T "DATE RETIRED:" /END.
  ON START PRINT ON 4 FMT 4T A10 24T 13 38T 13 53T 13 66T D2 END.
  REPORT END.
  PRINT "Your report has been sent to the line printer."
  PRINT "and can be picked up at the computer center."
  PRINT FMT S "Type a carriage return <cr> to continue: "END.
  ACCEPT L11424_WAIT.
  RELEASE 4.
ENDIF.
RETURN.
#
Module Name: 115 Bookkeeping
Parameters:
  In Only: GILL_Deposit
  Out Only:
  In/Out:
  Coded By: Robin FauntLeRoy
  Date Last Modified: 7/11/85
  Reason Modified:

Module Description:
  This module will maintain the amount in the checkbook.
  It will open the checkbook file and update the balance there.

DBSET CHECKBOOK.
FIND ALL.
CHANGE BAL BAL+GILL_DEPOSIT.
RETURN.
@
Module Name: 116 Find SSN

Parameters:
  In Only: Gll_First_Name
  Gll_Last_Name
  Out Only:
  In/Out: Gll_SS_Number
  Gll_Name

Coded By: Robin FauntLeRoy
Date Last Modified: 7/11/85

Module Description:
This module will search the Personnel and Registrar files when given a name, for a matching name and social security number. If found, it sets the name variables to the value of that name.

DEFINE TEXT 1 L116_ANSWER.

LET Gll_NAME = Gll_FIRST_NAME + Gll_LAST_NAME.
DBSET PSMAST.
FIND ALL.
SEARCH LNAM CT STRIM(Gll_LAST_NAME).
SEARCH FMNAM CT STRIM(Gll_FIRST_NAME).
IF SYSNREC EQ 1 THEN
  LET Gll_SS_NUMBER = SSN.
ELSEIF SYSNREC GT 1 THEN
  PRINT "With the information individual cannot be above records apply.
  ACCEPT L116_ANSWER.
IF L116_ANSWER EQ ('Y','y') THEN
  REPEAT
  PRINT "Enter the corresponding SSN. " END.
  ACCEPT Gll_SS_NUMBER.
  PRINT "Is this correct? (Y/N): " END.
  ACCEPT L116_ANSWER.
UNTIL L116_ANSWER EQ ('Y','y').
FIND ALL.
FIND SSN EQ Gll_SS_NUMBER.
ENDIF.
ELSE
  PRINT " 
ENDIF.

IF Gll_SS_NUMBER EQ 0 THEN
  DBSET SRMAST.
  FIND ALL.
  SEARCH LNAM CT STRIM(Gll_LAST_NAME).
  SEARCH FMNAM CT STRIM(Gll_FIRST_NAME).
  IF SYSNREC EQ 1 THEN
    LET Gll_SS_NUMBER = ID.
  ELSEIF SYSNREC GT 1 THEN
  PRINT " 
ENDIF.

ENDF.
ELSE
  PRINT " 
ENDIF.

ENDF.
PRINT NAME ID.
PRINT " ".
PRINT "With the information given the correct ".
PRINT "individual cannot be determined. All the ".
PRINT "above records apply. Do you have additional".
PRINT FMT $ "information to distinguish between entries (Y/N) "
END.

ACCEPT L116_ANSWER.
IF L116_ANSWER EQ ( 'Y', 'y' ) THEN
REPEAT
   PRINT " ".
   PRINT FMT $ "Enter the corresponding SSN. " END.
   ACCEPT G11_SS_NUMBER.
   PRINT " ".
   PRINT "SSN: " G11_SS_NUMBER.
   PRINT " ".
   PRINT FMT $ "Is this correct? (Y/N): " END.
   ACCEPT L116_ANSWER.
UNTIL L116_ANSWER EQ ( 'Y', 'y' ).
FIND ALL.
FIND ID EQ G11_SS_NUMBER.
LET G11_SS_NUMBER EQ ID.
LET G11_NAME EQ NAME.
ENDIF.
ELSE
   PRINT " ".
ENDIF.
ENDIF.
IF G11_SS_NUMBER EQ 0 THEN
   DBSET NAME.
   FIND ALL.
   SEARCH LNM CT $TRIM(G11_LAST_NAME).
   SEARCH FNM CT $TRIM(G11_FIRST_NAME).
   IF SYSNREC EQ 0 THEN
      PRINT "The matching SSN could not be found".
   ELSEIF SYSNREC EQ 1 THEN
      LET G11_SS_NUMBER EQ SSN.
   ELSE
      PRINT ".
   END.
   PRINT LNM FNM SSN.
   PRINT " ".
   PRINT "With the information given the correct ".
   PRINT "individual cannot be determined. All the ".
   PRINT "above records apply. Do you have additional".
   PRINT FMT $ "information to distinguish between entries (Y/N) "
END.

ACCEPT L116_ANSWER.
IF L116_ANSWER EQ ( 'Y', 'y' ) THEN
REPEAT
   PRINT " ".
   PRINT FMT $ "Enter the corresponding SSN. " END.
   ACCEPT G11_SS_NUMBER.
   PRINT " ".
   PRINT "SSN: " G11_SS_NUMBER.
   PRINT " ".
   PRINT FMT $ "Is this correct? (Y/N): " END.
   ACCEPT L116_ANSWER.
UNTIL L116_ANSWER EQ ( 'Y', 'y' ).
FIND ALL.
FIND SSN EQ G11_SS_NUMBER.
ENDIF.
ENDIF.
RETURN.
Module Name: 117 FIND Name

Parameters:

In Only: G11_SS_Number
Out Only:
In/Out: G111 Found
G11_Last_Name
G11_First_Name

Coded By: Robin Fauntleroy

Date Last Modified: 7/11/85
Reason Modified:

Module Description:
This module will search all available files for an occurrence of the social security number passed to it. If it finds one, then it sets the social security variable to that number.

DEFINE TEXT 1 L117_FILLER.

LET L117_FILLER EQ " ".
IF G11_SS_NUMBER LT 0 THEN

DBSET NAME.
FIND ALL.
FIND SSN EQ G11_SS_NUMBER.
IF SYSNREC NEQ 1 THEN
PRINT " ".
ELSE
LET G11_LAST_NAME EQ LNM.
LET G11_FIRST_NAME EQ FNM.
LET G11_NAME EQ STRIM(G11_FIRST_NAME) + L117_FILLER + STRIM(G11_LAST_NAME).
LET G111_FOUND EQ "T".
IF G1_REP0RT EQ "F" THEN
PRINT "NAME: "G11_NAME.
ENDIF.
ENDIF.
ELSE
DBSET PMAST.
FIND ALL.
FIND SSN EQ G11_SS_NUMBER.
IF SYSNREC NEQ 1 THEN
DBSET SRMAST.
FIND ALL.
FIND ID EQ G11_SS_NUMBER.
IF SYSNREC NEQ 1 THEN
PRINT " ".
ELSE
LET G11_NAME EQ NAME.
LET G111_FOUND EQ "T".
IF G1_REP0RT EQ "F" THEN
PRINT "NAME: "G11_NAME.
ENDIF.
ENDIF.
ELSE
LET G11_FIRST_NAME EQ FNMAM.
LET G11_LAST_NAME EQ LNAM.
LET G111_FOUND EQ "T".
LET G11_NAME EQ STRIM(G11_FIRST_NAME) + L117_FILLER + $TRIM(G11_}
LAST_NAME).
IF G1_REP0RT EQ "F" THEN
PRINT "NAME: "G1_NAME.
ENDIF.
ENDIF.
ENDIF.
LET G1_REPORT EQ "F".
RETURN.
@
Module Name: 1.2 Locksmith Main
Parameters:
  In Only: none
  Out Only: none
  In/Out: none
Coded By: Michele Miley
Date Last Modified: July 5, 1985
Reason Modified: Initial module creation.

Module Description:
The Locksmith Main module is a controlling module which displays a user menu, and asks the user to choose which locksmith subprogram he wishes to use. It has no parameters and accesses no files.

DEFINE TEXT 1 L12_MENU_CHOICE.
REPEAT
PUSH USING BL
PRINT " *
PRINT " * LOCKSMITH MENU *
PRINT " *
PRINT " Choose Option:
PRINT " A Complete Key Orders *
PRINT " B Replace Key *
PRINT " C Rekey Lock *
PRINT " D Inventory Control *
PRINT " E Exit *
PRINT " *
PRINT FMT $ "Enter choice: " END.
ACCEPT L12_MENU_CHOICE.
IF L12_MENU_CHOICE EQ 'A' OR L12_MENU_CHOICE EQ 'a' THEN
  CALL M121_COMPLETE_KEY_ORDERS.
ELSEIF L12_MENU_CHOICE EQ ('B', 'b') THEN
  CALL M122_REPLACE_KEY;
ELSEIF L12_MENU_CHOICE EQ ('C', 'c') THEN
  CALL M123_REKEY_LOCK;
ELSEIF L12_MENU_CHOICE EQ ('D', 'd') THEN
  CALL M124_INVENTORY_CONTROL;
ELSEIF L12_MENU_CHOICE NEQ ('E', 'e') THEN
  PRINT "You have entered an invalid choice."
ENDIF.
UNTIL L12_MENU_CHOICE EQ ('E', 'e').
RETURN.

M121_COMPLETE_KEY_ORDERS: @M121_COMPLETE_KEY_ORDERS.DMC
M122_REPLACE_KEY: @M122_REPLACE_KEY.DMC
M123_REKEY_LOCK: @M123_REKEY_LOCK.DMC
M124_INVENTORY_CONTROL: @M124_INVENTORY_CONTROL.DMC
Module Name: M_121 COMPLETE KEY ORDERS.DMC
Parameters:
  In Only: none
  Out Only: none
  In/Out: none
Coded By: Michele Miley
Date Last Modified: July 6, 1985
Reason Modified: Initial module creation.
Module Description:
This module presents the user with a menu pertaining to choices available to him in the locksmith subprogram of completing key orders.

DEFINE TEXT 1 L121 MENU_CHOICE.

REPEAT
  PUSH USING BL
  POP END.
  PRINT "***********************************************************************"
  PRINT "COMPLETE KEY ORDERS MENU"
  PRINT "Choose Option:"
  PRINT "A List Orders"
  PRINT "B Change Orders"
  PRINT "C Record Completion"
  PRINT "D Print Orders"
  PRINT "E Exit"
  PRINT "Enter choice: " END.
  ACCEPT L121 MENU_CHOICE.
  IF L121 MENU_CHOICE EQ ('A', 'a') THEN
    CALL LIST ORDERS.
  ELSEIF L121 MENU_CHOICE EQ ('B', 'b') THEN
    CALL CHANGE ORDERS.
  ELSEIF L121 MENU_CHOICE EQ ('C', 'c') THEN
    CALL RECORD COMPLETION.
  ELSEIF L121 MENU_CHOICE EQ ('D', 'd') THEN
    CALL PRINT ORDERS.
  ELSEIF L121 MENU_CHOICE EQ ('E', 'e') THEN
    PRINT "You have entered an invalid choice."
  ENDIF.
  UNTIL L121 MENU_CHOICE EQ ('E', 'e').
RETURN.

LIST ORDERS: #M1211 LIST ORDERS.DMC
CHANGE ORDERS: #M1212 CHANGE ORDERS.DMC
RECORD COMPLETION: #M1213 RECORD COMPLETION.DMC
PRINT ORDERS: #M1214 PRINT ORDERS.DMC
Module Name: M1211_LIST_ORDERS.DMC
Parameters:
In Only: none
Out Only: none
In/Out: none
Coded By: Michele Miley
Date Last Modified: July 7, 1985
Reason Modified: Initial module creation.
Module Description:
This module sorts the Order File by date and priority and
displays the contents of the file on the screen.

DEFINE TEXT 1 L1211_CONTINUE.
DBSET ORDER.
FIND ALL.
SORT BY PRIORITY ENTRY_DATE.
PUSH USING BL.
POP END.
PRINT " ".
PRINT " Priority Key Number Amount Needed Date Ordered".
PRINT " "
PRINT " -------- ---------- -------------- -------------- ".
PRINT " " PR KID AMT " " ED FMT A16 A9 A10 I9 A10 D3 END.
PRINT " ".
PRINT FMT $ "Type a carriage return <cr> to continue. " END.
ACCEPT L1211_CONTINUE.
RETURN.
@
Module Name: M1212_CHANGE_ORDERS.DMC
Parameters:
In Only: none
Out Only: none
In/Out: none
Coded By: Michele Miley
Date Last Modified: July 7, 1985
Reason Modified: Initial module creation.
Date Last Modified: July 21, 1985
Reason Modified: Format user interface.

Module Description:
This module allows the user to change fields in the Order File manually rather than use the system assigned values automatically.

DEFINE TEXT 1 L1212_DONE
L1212_NEW_PR
L1212_CORRECT.
DEFINE INTEGER L1212_AMT
L1212_ORDER_NUMBER.

CALL NUMBER_ORDER_FILE.
LET L1212_DONE EQ 'F'.
WHILE L1212_DONE NEQ 'T' DO
  DBSET ORDER.
  SORT BY PRIORITY ED.
  FIND ALL.
  PRINT "Please type the number of the record you wish to change."
  PRINT "(Enter a zero if you wish to exit this portion of the program.)".
  PRINT FMT S "Order Number: " END.
  ACCEPT L1212_ORDER_NUMBER.
  PRINT "What is the new priority you would like to assign?"
  PRINT FMT S "Priority: " END.
  ACCEPT L1212_NEW_PR.
  PRINT "What is the new amount to be assigned?"
  PRINT FMT S "Amount: " END.
  ACCEPT L1212_AMT.
  PRINT "New Priority: " L1212_NEW_PR.
  PRINT "New Amount: " L1212_AMT.
  PRINT FMT S "Are these the correct values? (Y/N) " END.
  ACCEPT L1212_CORRECT.
  UNTIL L1212_CORRECT EQ ('Y','y').
  GETREC LEAVE L1212_ORDER_NUMBER.
  CHANGE PR L1212_NEW_PR.
  CHANGE AMT L1212_AMT.
  PRINT "The order has been changed to the values you entered."
ELSE
  PRINT "There are no records corresponding to that order number."
ENDIF.
ENDWHILE.
RETURN.

NUMBER_ORDER_FILE: @M12121_NUMBER_ORDER_FILE.DMC
Module Name: M12121_NUMBER_ORDER_FILE.DMC
Parameters:
In Only: none
Out Only: none
In/Out: none
Coded By: Michele Miley
Date Last Modified: July 7, 1985
Reason Modified: Initial module creation.

Module Description:
This module accesses the order file and sorts it according to priority and date, then lists it along with a counter so that the user may select a record according to the counter value.

DEFINE INTEGER L12121_COUNT.
LET L12121_COUNT EQ 1.
DBSET ORDER.
FIND ALL.
SORT BY PRIORITY ED.
PUSH USING BL
POF END.
PRINT " ".
PRINT " Order Number Priority Key Number Amount Date Ordered".
PRINT "
WHILE SYSNREC GT 0 DO
GETREC LEAVE.
PRINT " " L12121_COUNT " " PR KID " " AMT " " ED FMT A15 I2 A11 I10
A10 A2 I5 A4 D3 END.
LET L12121_COUNT EQ L12121_COUNT+1.
ENDWHILE.
PRINT " ".
RETURN.
@
Module Name: M1213_RECORD_COMPLETION.DMC
Parameters:
In Only: none
Out Only: none
In/Out: none
Coded By: Michele Miley
Date Last Modified: July 7, 1985
Reason Modified: Initial module creation.
Module Description:
This module increases the amount of inventory listed in the
inventory file after keys have been ordered and cut.

DEFINE TEXT 10 L1213_KEY.
DEFINE TEXT 1 L1213_DONE.
DEFINE INTEGER L1213_ORDER_NUMBER.
DEFINE INTEGER L1213_AMT.
CALL M1213_NUMBER ORDER_FILE.
LET L1213_DONE EQ 'F'.
WHILE L1213_DONE NEQ 'T' DO
  DBSET ORDER.
  FIND ALL.
  SORT BY PRIORITY ED.
  REPEAT
    PRINT "Please type the number of the record which has been completed."
    PRINT "(Enter a zero if you wish to exit this portion of the program.)"
    PRINT FMT S "Order Number: "
    ACCEPT L1213_ORDER_NUMBER.
  END.
  IF L1213_ORDER_NUMBER LEQ SYSNREC THEN
    GETREC LEAVE L1213_ORDER_NUMBER.
    LET L1213_AMT EQ AMT.
    LET L1213_KEY EQ KID.
    DELETE.
    DBSET INVEN.
    FIND ALL.
    FIND KIN EQ L1213_KEY.
    CHANGE TN TN+L1213_AMT.
    CHANGE QON QON+L1213_AMT.
    DBSET KEYWAY.
    FIND KYN CT $LEFT(SCAPS($STRIM(L1213_KEY)), $LEN($STRIM(L1213_KEY)) - 6)
    CHANGE QON QON - L1213_AMT.
  END.
ELSE
  PRINT "There are no records corresponding to that order number."
ENDIF.
ENDWHILE.
RETURN.
M1213_NUMBER_ORDER_FILE: @M1213_NUMBER_ORDER_FILE.DMC
Module Name: Ml214_PRINT_ORDERS.DMC
Parameters:
   In Only: none
   Out Only: none
   In/Out: none
Coded By: Michele Miley
Date Last Modified: July 10, 1985
Reason Modified: Initial module creation.
Date Last Modified: July 21, 1985
Reason Modified: To stop screen display from scrolling.

Module Description:
This module prints the orders to be completed to the line printer.

DEFINE TEXT 1 L1214_CONT.

DBSET ORDER.
FIND ALL.
PUSH USING BL
POP END.
PRINT " ".
SORT BY PRIORITY ENTRY_DATE.
PRINT " Priority Key Number Amount Needed Date Ordered".
PRINT " ".
PRINT " PR KID AMT " ED FMT A16 A9 A10 I8 A10 D3 END.
PRINT " ".
PRINT FMT S "Type carriage return <cr> to continue: " END.
ACCEPT L1214_CONT.
RETURN.
@
Module Name: M122_REPLACE_KEY.DMC

Parameters:
- In Only: none
- Out Only: none
- In/Out: none

Coded By: Michele Miley
Date Last Modified: July 7, 1985
Reason Modified: Initial module creation

Module Description:
If a defective key is returned for replacement, this amount must be subtracted from the inventory file. This module accesses that file and automatically decrements the quantity on hand and the total amount.

DEFINE TEXT 10 L122_KEY_NUMBER.
DEFIN TEXT 1 L122_CRORECT
L122_CONT.

PRINT " ".
PRINT "Please enter the key number for the key to be replaced.".
PRINT "(If you wish to exit without entering the key number, type E for Exit.)".
PRINTFmt $"Key Number: " END.
ACCEPT L122_KEY_NUMBER.
PRINT " ".
IF L122_KEY_NUMBER NEQ ('E', 'e') THEN
PRINT FMT $ "Key Number: " END.
PRINT L122 KEY_NUMBER.
PRINT FMT $ "Is this the correct key number? (Y/N) " END.
ACCEPT L122_CORRECT.
IF L122_CORRECT EQ ('N', 'n') THEN
  REPEAT
    PRINT " ".
    PRINT FMT $ "Please re-enter the key number: " END.
    PRINT " ".
    PRINT FMT $ "Key Number: " END.
    PRINT L122 KEY_NUMBER.
    PRINT FMT $ "Is this the correct entry? (Y/N) " END.
  UNTIL L122_CORRECT EQ ('Y','y').
ENDIF.
DBSET INVEN.
FIND KIN EQ L122_KEY_NUMBER.
IF (QON LEQ 0) OR (TN LEQ 0) OR (SYSREC EQ 0) THEN
  PRINT " ".
  PRINT "The inventory file shows there are no keys listed".
  PRINT "by that number.".
  PRINT FMT $ "Type carriage return <cr> to continue: " END.
  ACCEPT L122_CONT.
ELSE
  PRINT " ".
  PRINT "The replacement of that key has been recorded.".
  PRINT FMT $ "Type carriage return <cr> to continue: " END.
  ACCEPT L122_CONT.
  CHANGE QON QON-1, TN TN-1.
ENDIF.
ENDIF.
RETURN.
Module Name: M123_REKEY_LOCK.DMC
Parameters:
  In Only: none
  Out Only: none
  In/Out: none
Coded By: Michele Miley
Date Last Modified: July 25, 1985
Reason Modified: Initial module creation.

Module Description:
This module prompts the user for the room being rekeyed and
calls the appropriate modules to generate the new key number
and update the files.

DEFINe TEXT 10 G123 KEY_NUM
  G123 NEW_KEY.
DEFINe TEXT 4 G123 ROOM.
DEFeNe TEXT 3 G123 BLDG.
DEFINe TEXT 1 G123 FOUND
  G123 ACCEPT.
DEFINe INTEGER G123 AMT
  G123 ORDER_AMT.
DEFINe TEXT 1 G123 BLDG.
DEFINe TEXT 1 G123 ROOM.
LET G123 FOUND EQ 'F'.
REPEAT
  PRINT "Please enter the building and room number that are being rekeyed,"
  PRINT "following their respective prompts.".
  PRINT FMT $ "Building: "  END.
  ACCEPT G123 BLDG.
  PRINT FMT $ "Room number: "  END.
  ACCEPT G123_ROOM.
  PRINT FMT $ "Are these values correct? (Y/N) "  END.
  ACCEPT G123 CORRECT.
UNTIL G123_CORRECT EQ ('Y','y').

CALL G123_GET_NUM.
IF G123 ACCEPT EQ ('Y','y') THEN
  CALL G123 UPDATE.
  PRINT "The files have been changed to reflect the new key number.".
  PRINT FMT $ "Type carriage return <cr> to continue: "  END.
  ACCEPT G123 CORRECT.
ENDIF.
RETURN.

G123_GET_NUM: @G123_GET_NEW_NUMBER.DMC
G123 UPDATE: @G123.UPDATE_FILES.DMC
@
Module Name: M1231_GET_NEW_NUMBER

Parameters:

In Only: G123_BLDG, G123_ROOM
Out Only: G123_ACCEPT, G123_KEY_NUM, G123_AMT, G123_ORDER_AMT,
G123_NEW_KEY
In/Out: none

Coded By: Michele Miley
Date Last Modified: July 25, 1985
Reason Modified: Initial module creation.

Module Description:
This module determines the new key number and accepts an existing number as the new number or calls the appropriate modules for determining a new number.

DEFINE TEXT 1 L1231_CORRECT
L1231_TYPE.

DBSET LOCATE.
FIND ALL.
SEARCH BLD CT SCAPS(STRIM(G123_BLDG)) AND RN CT SCAPS(STRIM(G123_ROOM)).
IF SYSNREC EQ 1 THEN
LET G123_KEY_NUM EQ KIN.
ELSEIF SYSNREC GT 1 THEN
PRINT "More than one key will fit that lock."
REPEAT
PRINT "Please enter the key number you are changing."
PRINT FMT $ "Key Number: " END.
ACCEPT G123_KEY_NUM.
PRINT "$".
PRINT "Key Number: " G123_KEY_NUM.
PRINT FMT $ "Is this the correct key number? (Y/N) " END.
ACCEPT L1231_CORRECT.
UNTIL L1231_CORRECT EQ ('Y', 'y').
ELSE
PRINT "There is no record of a key fitting that lock."
REPEAT
PRINT "Please enter the key number opening that lock."
PRINT FMT $ "Key Number: " END.
ACCEPT G123_KEY_NUM.
PRINT "$".
PRINT "Key Number: " G123_KEY_NUM.
PRINT FMT $ "Is this the correct key number? (Y/N) " END.
ACCEPT L1231_CORRECT.
UNTIL L1231_CORRECT EQ ('Y', 'y').
ENDIF.

REPEAT
PRINT ".
PRINT "Do you wish to use a new key or an existing one? ".
PRINT FMT $ "Type 'N' for new or 'E' for existing: " END.
ACCEPT L1231_TYPE.
IF L1231_TYPE EQ ('N', 'n') THEN
PRINT " ".
PRINT FMT $ "Do you have a number you would like to use? (Y/N) " END.
ACCEPT L1231_TYPE.
IF L1231_TYPE EQ ('Y', 'y') THEN

...
REPEAT
    PRINT " ".
    PRINT "Please enter the key number you would like to use.".
    PRINT FMT $ "Key Number: " END.
    ACCEPT G123_NEW_KEY.
    PRINT " ".
    PRINT "Key Number: " G123_NEW_KEY.
    PRINT FMT $ "Is this the correct key number? (Y/N) " END.
    ACCEPT L1231_CORRECT.
    UNTIL L1231_CORRECT EQ ('Y', 'y').
ELSE
    CALL L1231_GEN_NEW.
ENDIF.
ELSE
    REPEAT
        PRINT " ".
        PRINT "What is the existing number you would like to use? ".
        PRINT FMT $ "Key Number: " END.
        ACCEPT G123_NEW_KEY.
        PRINT " ".
        PRINT "Key Number: " G123_NEW_KEY.
        PRINT FMT $ "Is this the correct key number? (Y/N) " END.
        ACCEPT L1231_CORRECT.
        UNTIL L1231_CORRECT is: " G123_NEW_KEY.
    PRINT FMT $ "Do you wish to use this number? (Y/N) " END.
    ACCEPT G123_ACCEPT.
    IF G123_ACCEPT EQ ('Y', 'y') AND L1231_TYPE EQ ('E', 'e') THEN
        PRINT " ".
        PRINT "How many keys do you have on hand?".
        PRINT FMT $ "Amount on Hand: " END.
        ACCEPT G123_AMT.
    ENDIF.
    IF G123_ACCEPT EQ ('Y', 'y') THEN
        PRINT " ".
        PRINT "How many keys do you need to make?".
        PRINT FMT $ "Amount to Order: " END.
        ACCEPT G123_ORDER_AMT.
    ELSE
        PRINT " ".
        PRINT "The files have not been changed.".
        PRINT FMT $ "Type carriage return <cr> to continue: " END.
        ACCEPT L1231_CORRECT.
    ENDIF.
RETURN.
L1231_GEN_NEW: @M12311_GENERATE_NEW_NUMBER.DMC
L1231_CHECK: @M12312_CHECK_APPLICABILITY.DMC
Module Name: M12311_GENERATE_NEW_NUMBER.DMC
Parameters:
In Only: G123 KEY_NUM
Out Only: G123_NEW_KEY
In/Out: none
Coded By: Michele Miley
Date Last Modified: July 25, 1985
Reason Modified: Initial module creation.

Module Description:
This module generates a new key number when a lock is being rekeyed and the user does not wish to use an old number.

DEFINE TEXT 4 L12311_KWY.
DEFINE TEXT 6 L12311_KEYNO.
DEFINE INTEGER L12311_KID.

LET L12311_KWY EQ $LEFT(STRIM(G123_KEY_NUM),SLEN(STRIM(G123_KEY_NUM)) - 6).

DBSET INVEN.
FIND ALL.
FIND G123_FOUND NEQ 'T' DO
GETREC LEAVE.
WHILE G123_FOUND NEQ 'T' DO
LET L12311_KEYNO EQ $RIGHT(STRIM(KIN),6).
LET L12311_KID EQ $INT(L12311_KEYNO).
LET L12311_KEYNO EQ $EXITR(L12311_KID).
FIND KIN CT $CAPS(STRIM(L12311_KWY)) AND KIN CT $CAPS(STRIM(L12311_KEYNO)).
IF SYSNREC EQ 0 THEN
LET G123_FOUND EQ 'T'.
LET G123_NEW_KEY EQ $CAPS(STRIM(L12311_KWY) + STRIM(L12311_KEYNO)).
ENDIF.
ENDWHILE.
RETURN.
0
Module Name: M12341_CHECK_APPLICABILITY

Parameters:
In Only: G123_NEW_KEY, G123_AMT
Out Only: G123_FOUND, G123_NO_MORE
In/Out: none

Coded By: Michele Miley
Date Last Modified: July 25, 1985
Reason Modified: Initial module creation.

Module Description:
This module checks for the availability for use of a previously retired key.

DBSET INVEN.
FIND ALL.
FIND KIN EQ $CAPS($STRIM(G123_NEW_KEY)).
IF SYSNREC EQ 0 THEN
   LET G123_FOUND EQ 'T'.
ELSEIF STATUS EQ 'R' AND DR LT SYSDATE - 365 THEN
   LET G123_FOUND EQ 'T'.
ELSE
   PRINT " ".
   PRINT "That number is not available for use.".
   LET G123_FOUND EQ 'F'.
ENDIF.
RETURN.

@
Module Name: M1232_UPDATE_FILES
Parameters:
  In Only: G123_NEW_KEY, G123_AMT, G123_ORDER_AMT, G123_KEY_NUM
  Out Only: none
  In/Out: none
Coded By: Michele Miley
Date Last Modified: July 25, 1985
Reason Modified: Initial module creation.

Module Description:
If a key number is accepted, this module updates all the files associated with the rekeyed lock.

DBSET INVEN.
FIND ALL.
FIND KIN EQ SCAPS(G123_KEY_NUM).
CHANGE ST "R" DR SYSDATE.
FIND ALL.
FIND KIN EQ SCAPS(G123_NEW_KEY).
IF SYSNREC GT 0 THEN
  CHANGE ST " " DR " ".
  CHANGE QON G123_AMT TN G123_AMT.
ELSE
  ADD KIN $SCAPS(G123_NEW_KEY) KY "C" QON G123_AMT TN G123_AMT ST " " DR " ".
ENDIF.
DBSET LOCATE.
FIND ALL.
FIND KIN EQ SCAPS(G123_KEY_NUM).
SEARCH BLD CT SCAPS(STRIM(G123_BLDG)) AND RN CT SCAPS(STRIM(G123_ROOM)).
IF SYSNREC GT 0 THEN
  CHANGE KIN SCAPS(G123_NEW_KEY).
ELSE
  ADD KIN $SCAPS(G123_NEW_KEY) BLD $SCAPS(G123_BLDG) RN $SCAPS(G123_ROOM).
ENDIF.
DBSET ORDER.
ADD PR 1 KID $SCAPS(G123_NEW_KEY) AMT G123_ORDER_AMT.
RETURN.
Module Name: M124_INVENTORY_CONTROL

Parameters:

In Only: none
Out Only: none
In/Out: none

Coded By: Michele Miley

Date Last Modified: July 5, 1985

Reason Modified: Initial module creation.

Module Description:

This module is a controlling module for the inventory control subsystem of the locksmith portion of the Key Inventory Control system. It displays a menu allowing the user to choose the options available to him.

DEFINE TEXT 1 L124_MENU_CHOICE.

REPEAT

PUSH USING BL
POP END.
PRINT "INVENTORY CONTROL MENU"
PRINT "Choose Option:
PRINT "A Change Inventory"
PRINT "B Order Parts"
PRINT "C Add Item"
PRINT "D Delete Item"
PRINT "E Print Inventory"
PRINT "F Exit"
PRINT "Enter choice: " END.
ACCEPT LI24_MENU_CHOICE.
IF LI24_MENU_CHOICE EQ ('A', 'a') THEN
CALL CHANGE_INVENTORY.
ELSEIF LI24_MENU_CHOICE EQ ('B', 'b') THEN
CALL ORDER_PARTS.
ELSEIF LI24_MENU_CHOICE EQ ('C', 'c') THEN
CALL ADD_ITEM.
ELSEIF LI24_MENU_CHOICE EQ ('D', 'd') THEN
CALL DELETE_ITEM.
ELSEIF LI24_MENU_CHOICE EQ ('E', 'e') THEN
CALL PRINT_INVENTORY.
ELSEIF LI24_MENU_CHOICE EQ ('F', 'f') THEN
PRINT "You have entered an invalid choice."
ENDIF.
UNTIL LI24_MENU_CHOICE EQ ('F', 'f').
RETURN.

CHANGE_INVENTORY: @M1241_CHANGE_INVENTORY.DMC
ORDER_PARTS: @M1244 ORDER_PARTS.DMC
ADD_ITEM: @M1242_ADD_ITEM.DMC
DELETE_ITEM: @M1245_DELETE_ITEM.DMC
PRINT_INVENTORY: @M1243_PRINT_INVENTORY.DMC
Module Name: M1241_CHANGE_INVENTORY.DMC
Parameters:
   In Only: none
   Out Only: none
   In/Out: none
Coded By: Michele Miley
Date Last Modified: July 8, 1985
Reason Modified: Initial module creation.
Date Last Modified: July 21, 1985
Reason Modified: To allow for lower case text entries.

Module Description:
This module allows the user to change an inventory record in the keyway file. Only the amount field may be changed.

DEFINE TEXT 4 L1241_KWY.
DEFINE TEXT 1 L1241_CORRECT.
DEFINE TEXT 1 L1241_CONTINUE.
DEFINE INTEGER L1241_AMT.

PRINT " ".
PRINT FMT $ "Do you wish to continue with the inventory change process? (Y/N) " END.
ACCEPT L1241_CONTINUE.
IF L1241_CONTINUE EQ ('Y', 'y') THEN
   REPEAT
      PRINT " ".
      PRINT "Please enter the keyway number for which you wish to make a change."
      PRINT FMT $ "Keyway Number: " END.
      ACCEPT L1241_KWY.
      PRINT " ".
      PRINT FMT $ "Keyway Number: " END.
      PRINT L1241_KWY.
      PRINT FMT $ "Is this the correct entry? (Y/N) " END.
      ACCEPT L1241_CORRECT.
   UNTIL L1241_CORRECT EQ ('Y', 'y').

DBSET KEYWAY.
FIND KYN CP $CAPS($TRIM(L1241_KWY)).
PRINT " ".
PRINT "The current inventory record is: ".
PRINT "Keyway Number: " L1241_KWY.
PRINT "Quantity on Hand: " QON.
IF SYSNRG GT 0 THEN
   LET L1241_CORRECT EQ 'F'.
   REPEAT
      PRINT " ".
      PRINT "What is the new amount you would like to enter?"
      PRINT FMT $ "Amount: " END.
      ACCEPT L1241_AMT.
      PRINT " ".
      PRINT FMT $ "Amount: " END.
      PRINT L1241_AMT.
      PRINT FMT $ "Is this the correct amount? (Y/N) " END.
      ACCEPT L1241_CORRECT.
   UNTIL L1241_CORRECT EQ ('Y', 'y').
   CHANGE QON L1241_AMT.
   PRINT ".
   PRINT "The inventory record has been changed to: ".
"
PRINT "Keyway Number:" XIN "Quantity on hand:" QUN.
PRINT "."
PRINT PVT "$ "Type carriage return <cr> to continue: " END.
ACCEPT L1241_CONTINUE.
ELSE
PRINT "There are no entries in the file corresponding to that".
PRINT "keyway number."
PRINT PVT "$ "Type carriage return <cr> to continue: " END.
ACCEPT L1241_CONTINUE.
ENDIF.
ENDIF.
RETURN.
Module Name: M12421_UPDATE_FILE.DMC

Parameters:
In Only: G1242_AMT, G1242_KWY
Out Only: none
In/Out: none

Coded By: Michele Miley
Date Last Modified: July 8, 1985
Reason Modified: Initial module creation.
Date Last Modified: July 21, 1985
Reason Modified: To allow for lower case text entries.

Module Description:
This module accepts the keyway number and amount to be added to
the keyway quantity on hand from M1242_ADD_ITEM, and proceeds
to add the amount to the keyway file quantity on hand.

+-------------------------------------------------------------+
| DEFINE TEXT 1 L12421_CORRECT.                              |
| DEFINE TEXT 1 L12421_CONT.                                 |
| DBSET KEYWAY.                                              |
| FIND KYN CT SCAPS(STRM(G1242_KWY)).                        |
| IF SYSREC CT 0 THEN                                         |
|   CHANGE QON QON+G1242_AMT.                               |
|   PRINT " The updated inventory record is: ".             |
|   PRINT " Keyway Number: " KYN.                           |
|   PRINT " Quantity on Hand: " QON.                         |
|   PRINT " ".                                              |
|   PRINT FMT $ "Type carriage return <cr> to continue: " END.|
|   ACCEPT L12421_CONT.                                     |
| ELSE                                                      |
|   PRINT " The keyway file shows no entry for that keyway number.". |
|   PRINT " ".                                              |
|   PRINT FMT $ "Keyway Number: " END.                      |
|   PRINT G1242_KWY.                                         |
|   PRINT FMT $ "Is this the correct entry? (Y/N) " END.     |
|   ACCEPT L12421_CORRECT.                                  |
|   IF L12421_CORRECT EQ ('Y', 'y') THEN                      |
|     ADD KYN G1242_KWY QON G1242_AMT.                      |
| ENDIF.                                                    |
RETURN.
Module Name: M1243_PRINT_INVENTORY.DMC

Parameters:

In Only: none
Out Only: none
In/Out: none

Coded By: Michele Miley
Date Last Modified: July 10, 1985
Reason Modified: Initial module creation.

Date Last Modified: July 21, 1985
Reason Modified: To change user interface.

Module Description:
This module prints a listing of the inventory file on the line printer.

DEFINE TEXT 1 L1243_CONT.

DBSET KEYWAY.
FIND ALL.
PUSH USING BL
POP END.
PRINT " 
PRINT " Keyway Number Quantity on Hand".
PRINT " 
PRINT " KYN QON FMT A21 A20 I3 END.
PRINT 
PRINT FMT $ "Type carriage return <cr> to continue; " END.
ACCEPT L1243_CONT.
RETURN.
@
DEFINE TEXT 1 L1244__CONT.

CALL CHECK_FILE.
PUSH USING BL
POP END.
DBSET KEY_ORDER_FILE.
FIND ALL.
PRINT "  Keyway Number Amount".
PRINT "  -------------- -----
PRINT "  KYN QON FMT A19 A13 13 END.
PRINT "$ Type carriage return <cr> to continue: " END.
ACCEPT L1244__CONT.
RETURN.

CHECK_FILE: @M12441_CHECK_FILE.DMC

*
Module Name: M12441_CHECK_FILE.DMC

Parameters:
- In Only: none
- Out Only: none
- In/Out: none

Coded By: Michele Miley

Date Last Modified: July 10, 1985
Reason Modified: Initial module creation.
Date Last Modified: July 21, 1985
Reason Modified: To change file access method.

Module Description:
This module checks the keyway file for amounts less than the threshold number and places these in the key order file.

DBSET KEY_ORDER_FILE.
CLOSE.
FILE DELETE KEY_ORDER_FILE.DMS.
DBSET KEYWAY.
FIND ALL.
SELECT QON LT 100.
DUMP SET KEY_ORDER_FILE.
OPEN NOCLOSE KEY_ORDER_FILE.
KEY NOMSG ALL.

FIND ALL.
CHANGE QON 100 = QON.
RETURN.
@
Module Name: Ml245_DELETE_ITEM.DMC
Parameters:
  In Only: none
  Out Only: none
  In/Out: none
Coded By: Michele Miley
Date Last Modified: July 8, 1985
Reason Modified: Initial module creation.
Date Last Modified: July 21, 1985
Reason Modified: To allow for lower case text entries.

Module Description:
This module will delete an entry from the keyway file when
the user indicates this keyway number is no longer in use.

DEFINE TEXT 4 L1245_KWY.
DEFINE TEXT 1 L1245_CORRECT
L1245_CONTINUE.
REPEAT
  PRINT " ".
  PRINT "Please enter the keyway number for the record to be deleted."
  PRINT FMT $ "Keyway Number: " END.
  ACCEPT L1245_KWY.
  DBSET KEYWAY.
  FIND KYN CT SCAPS(STRIM(L1245_KWY)).
  IF SYSNRSC CT 0 THEN
    PRINT " ".
    PRINT FMT $ "Keyway Number: " END.
    PRINT L1245_KWY.
    PRINT FMT $ "Is this the correct entry? (Y/N) " END.
    ACCEPT L1245_CORRECT.
    IF L1245_CORRECT EQ ('Y', 'y') THEN
      PRINT " ".
      PRINT "Continuation of this procedure will result in the deletion"
      PRINT FMT $ "of that record. Do you wish to continue? (Y/N) " END
      ACCEPT L1245_CONT.
      IF L1245_CONT EQ ('Y', 'y') THEN
        DELETE.
        PRINT "The record for that keyway number has been deleted."
      ENDIF.
    ENDIF.
  ELSE
    PRINT " ".
    PRINT "There are no records corresponding to that number."
  ENDIF.
ENDIF.
ELSE
  PRINT " ".
  PRINT FMT $ "Do you wish to delete another entry? (Y/N) " END.
  ACCEPT L1245_CONTINUE.
UNTIL L1245_CONTINUE EQ ('N', 'n').
RETURN.
@
Module Name: 13 Maint Main

Parameters:
- In Only: none
- Out Only: none
- In/Out: none

Coded By: Robin FauntLeRoy
Date Last Modified: 7/12/85
Reason Modified:

Module Description:
This module is the controller for all that occurs in the upkeep of this system. It includes such things as backing up the information on tape, and ordering keys made.

CALL BACKUP_INFO.
CALL ORDER KEY.
CALL UPDATE_KEY_STATUS.
RETURN.

BACKUP INFO: @R131_BACKUP_INFO.DMC
ORDER KEY: @R132_ORDER_KEY.DMC
UPDATE_KEY: @R133_UPDATE_KEY.DMC
@
Module Name: 131 BACKUP INFO
Parameters:
In Only: G1_REQUESTED
Out Only: G131_FILE_CHOICE
In/Out: none
Coded By: Robin FauntLeRoy
Date Last Modified: 7/15/85
Reason Modified:
Module Description:
This module is passed requested, which will determine whether
the user wished to enter this section of the program. If so, the
menu will be displayed, otherwise, if it has been 7 days since the
last save, then an automatic save of all files for backup will be
requested.

DEFINE TEXT 1 G131_FILE_CHOICE.

IF G1_REQUESTED EQ "F" THEN
DBSET LAST_SAVE.
FIND ALL.
IF $DATEJUL(DLS) LT $DATEJUL(SYSDATE)-7 THEN
CALL SAVE.
ENDIF.
ELSE
REPEAT
PUSH USING BL
POP END.
REPEAT
PRINT "* • >
PRINT "• • *
PRINT "*  • •
PRINT "* FILE RESTORATION MENU
PRINT "* » »PRINT " *
PRINT "* Enter file to be restored:
PRINT "* A Individual Record File
PRINT "* B Name File
PRINT "* C Inventory File
PRINT "* D Location File
PRINT "* E Keyway File
PRINT "* F Order File
PRINT "* G All Files
PRINT "* H Exit
PRINT "Please enter the file to be restored: " END.
ACCEPT G131_FILE_CHOICE.
IF G131_FILE_CHOICE NEQ ('A', 'a', 'B', 'b', 'C', 'c', 'D', 'd', 'E', 'e', 'F', 'f', 'G', 'g', 'H', 'h') THEN
  PRINT "Invalid choice, please try again."
ENDIF.
UNTIL G131_FILE_CHOICE EQ ('A', 'a', 'B', 'b', 'C', 'c', 'D', 'd', 'E', 'e', 'F', 'f', 'G', 'g', 'H', 'h').
IF G131_FILE_CHOICE NEQ ('H', 'h') THEN
  CALL RESTORE.
ENDIF.
UNTIL G131_FILE_CHOICE EQ ('H', 'h').
ENDIF.
RETURN.

RESTORE: @R1312_RESTORE.DMC
SAVE: @R1311_SAVE.DMC
Module Name: 1311 SAVE
Parameters:
In Only: none
Out Only: none
In/Out: none
Coded By: Robin FauntLeRoy
Date Last Modified: 7/15/85
Reason Modified:

Module Description:
This module calls a batch job which will mount the tape used for saving files and save all pertinent files for the key inventory control system.

PUSH USING SAVE.CTL
POP END.

RETURN.
Module Name: 1312 Restore
Parameters:
  In Only: G131_File_Choice
  Out Only: none
  In/Out: none
Coded By: Robin FauntLeRoy
Date Last Modified: 7/15/85
Reason Modified:
Module Description:
This module is passed the menu choice and thereby decides which batch job to call to save the particular file(s) of choice.

IF G131_FILE_CHOICE EQ ('A','a') THEN
  PUSH USING INDIVIDUAL.CTL/OUTPUT:ERRORS
  POP END.
ELSEIF G131_FILE_CHOICE EQ ('B','b') THEN
  PUSH USING NAME.CTL/OUTPUT:ERRORS
  POP END.
ELSEIF G131_FILE_CHOICE EQ ('C','c') THEN
  PUSH USING INVENTORY.CTL/OUTPUT:ERRORS
  POP END.
ELSEIF G131_FILE_CHOICE EQ ('D','d') THEN
  PUSH USING LOCATION.CTL/OUTPUT:ERRORS
  POP END.
ELSEIF G131_FILE_CHOICE EQ ('E','e') THEN
  PUSH USING KEYWAY.CTL/OUTPUT:ERRORS
  POP END.
ELSEIF G131_FILE_CHOICE EQ ( 'F','f') THEN
  PUSH USING ORDER.CTL/OUTPUT:ERRORS
  POP END.
ELSE
  PUSH USING RESTORE.CTL/OUTPUT:ERRORS
  POP END.
ENDIF.
RETURN.
Module Name: 132 Order Key
Parameters:

In Only:

Out Only:

In/Out:

Coded By: Robin Fauntleroy
Date Last Modified: 7/17/85
Reason Modified:

Module Description:
This module will access the inventory file and check to be sure that the quantity on hand is adequate. If not, it will map the Order file to find out whether it has been ordered or not, and if not, insert the order for more keys into the order file.

DEFINE TEXT 1 L132_PRIOR.
DEFINE INTEGER L132_QON.
DEFINE TEXT 10 L132_KEY_NUMBER.

DBSET ORDER_DATE.
FIND ALL.
IF OD NEQ SYSDATE THEN
  CHANGE OD SYSDATE.
  DBSET INVEN.
  FIND ALL.
  SEARCH QON LT 3 ST NEQ "R".
  IF SYSNREC GT 0 THEN
    REPEAT.
    GETREC LEAVE.
    LET L132_QON EQ QON.
    LET L132_KEY_NUMBER EQ KIN.
    MAP TO ORDER VIA KIN TO KID.
    IF SYSNREC EQ 1 THEN
      IF (3-L132_QON) GT AMT THEN
        CHANGE AMT (3-L132_QON).
      ENDIF.
      ELSE
        IF L132_QON EQ 1 THEN
          LET L132_PRIOR EQ "1".
        ELSE
          LET L132_PRIOR EQ "2".
        ENDIF.
      ENDIF.
    ADD PR L132_PRIOR KID L132_KEY_NUMBER AMT (3-L132_QON).
  ENDIF.
UNTIL SYSNREC EQ 0.

ENDIF.

RETURN.

Module Name: R133_UPDATE_KEY.DMC
Parameters:
   In Only: none
   Out Only: none
   In/Out: none
Coded By: Michele Miley
Date Last Modified: July 21, 1985
Reason Modified: Initial module creation.

Module Description:
   This module updates the inventory file so that keys retired
   for more than one year are listed as available.

RETURN.
@
Module Name: 2 Data Entry Main
Parameters:
In Only:
Out Only:
In/Out:
Coded By: Robin FauntLeRoy
Date Last Modified: 7/12/85
Reason Modified:

Module Description:
This module simply calls other modules until the user wishes to quit.

CLEAR.
DEFINE TEXT 17 G11_LAST_NAME.
SET FMSG OFF.
OPEN IND_RECORD DUMMY_S5 PSMAST SRMAST NAME.

PL1022 START.
REPEAT
CALL ACCEPT_NAME.
CALL ENTER_INFO.
UNTIL G11_LAST_NAME EQ Q.

ACCEPT_NAME: @R21_ACCEPT_NAME.DMC
ENTER_INFO: @R22_ENTER_INFO.DMC

PL1022 STOP.
CLOSE.
CLOSE.
CLOSE.
SET FMSG ON.
PL1022 END.

Module Name: 21 Accept Name
Parameters:
In Only:
Out Only: gll_ss_number
In/Out:
Coded By: Robin Fauntleroy
Date Last Modified: 7/29/85
Reason Modified:

Module Description:
This module will determine if a name exists on campus for the
d individual who has a key issued to him

DEFINE TEXT 1 L21_ANSWER
TEXT 30 Gll_NAME
TEXT 13 Gll_FIRST_NAME.
DEFINE INTEGER Gll_SS_NUMBER.

LET Gll_SS_NUMBER EQ 0.
REPEAT
PRINT " ".
PRINT "Please enter issuee's last name.".
PRINT FMT $ "( or a Q to quit entering ). "END.
ACCEPT Gll_LAST_NAME.
PRINT " ".
PRINT "Name: "Gll_LAST_NAME.
PRINT FMT $ "Is this correct? (y or n): "END.
ACCEPT L21_ANSWER.
UNTIL L21_ANSWER EQ ( 'Y','y').
IF Gll_LAST_NAME NEQ "Q" THEN
REPEAT
PRINT " ".
PRINT FMT $ "Please enter issuee's first name. "END.
ACCEPT Gll_FIRST_NAME.
PRINT " ".
PRINT "Name: "Gll_FIRST_NAME.
PRINT FMT $ "Is this correct? (y or n): "END.
ACCEPT L21_ANSWER.
UNTIL L21_ANSWER EQ ( 'Y','y').
LET Gll_NAME EQ Gll_LAST_NAME + Gll_FIRST_NAME.
CALL FIND_SSN.
IF Gll_SS_NUMBER EQ 0 THEN
DBSET DUMMY_SS.
FIND ALL.
LET Gll_SS_NUMBER EQ SSN.
CHANGE SSN SSN-1.
ENDIF.
ENDIF.

FIND_SSN: @R16_FIND_SSN.DMC
Module Name: 22 Enter Info
Parameters:
  In Only: gll_ss_number
  Out Only:
  In/Out:
Coded By: Robin Fauntleroy
Date Last Modified: 7/12/85
Reason Modified:
Module Description:
This module enters into the appropriate files the incoming
information about a key which is issued.

DEFINE TEXT 10 L22_KEY_NUMBER
TEXT 5 L22_ANSWER
TEXT 1 L22_WAIT.
DEFINE INTEGER L22_AMOUNT
INTEGER L22_DEPOSIT.
IF gll_LAST_NAME NEQ "Q" THEN
  IF gll_SS_NUMBER LT 0 THEN
    DBSET NAME.
    ADD NM gll_NAME SSN gll_SS_NUMBER.
  ENDIF.
  DBSET IND_RECORD.
REPEAT
  PRINT " ".
  PRINT FMT $ "Please enter key-number: " END.
  ACCEPT L22_KEY_NUMBER.
  PRINT " ".
  PRINT FMT $ "Key Number: " L22_KEY_NUMBER.
  PRINT FMT S "Is this correct? (y or n): " END.
  ACCEPT L22_WAIT.
  UNTIL L22_WAIT EQ ("Y", "y").
  PRINT " ".
  PRINT "Enter deposit amount".
  PRINT IF amount is other than normal, type in ".
  PRINT IF amount is other than normal, type in new amount. Otherwise type a carriage return. " END.
  ACCEPT L22_ANSWER.
  LET L22_AMOUNT EQ $INT(L22_ANSWER).
  IF L22_AMOUNT GT 0 THEN
    LET L22_DEPOSIT EQ L22_AMOUNT.
  ELSE
    LET L22_DEPOSIT EQ 200.
  ENDIF.
  LET L22_DEPOSIT EQ 200.
ENDIF.
ADD SSN gll_SS_NUMBER KIN L22_KEY_NUMBER DEP L22_DEPOSIT NUL.
ENDIF.
RETURN.
MAIL.CTL.2

MAIL
*CS.GRAD.FAUNTLEROY
*
*UNAUTHORIZED ACCESS
*An attempt was made to enter the Key Inventory
*Control system by someone who did not know the
*proper password for entry.
*2

NAME.CTL.1

@MOUNT TAPE KIC:/WRI
@DUMPER
*TAPE KIC:
*FILES
*REWIND
*RESTORE NAME.DMS
*REWIND
*EXIT
@DISMOUNT KIC:
*FILE
*REWIND
*RESTORE NAME.DMS
*REWIND
*EXIT
@DISMOUNT KIC:

ORDER.CTL.1

@MOUNT TAPE KIC: /WRI
@DUMPER
*TAPE KIC:
*FILES
*REWIND
*RESTORE ORDER.DMS
*REWIND
*EXIT
@DISMOUNT KIC:

RESTORE.CTL.1

@MOUNT TAPE KIC: /WRI
@DUMPER
*TAPE KIC:
*FILES
*REWIND
*RESTORE
*REWIND
*EXIT
@DISMOUNT KIC:

SAVE.CTL.1

@MOUNT TAPE KIC: /WRI
@DUMPER
*TAPE KIC:
*FILES
*REWIND
*SSNAME KEY INVENTORY CONTROL DATA FILES
*SAVE INQ_RECORD.DMS TMPINQ_RECORD.DMS, NAME.DMS TMPNAME.DMS, LOCATION.DMS
*TMPLOCATION.DMS, INVENTORY.DMS TMPINVENTORY.DMS, KEYWAY.DMS TMPKEYWAY.DMS
*ORDER.DMS TMPORDER.DMS
*EXIT
@DISMOUNT KIC:
@