Trevor Ratchford
CMPT 220 Final Project
Arias

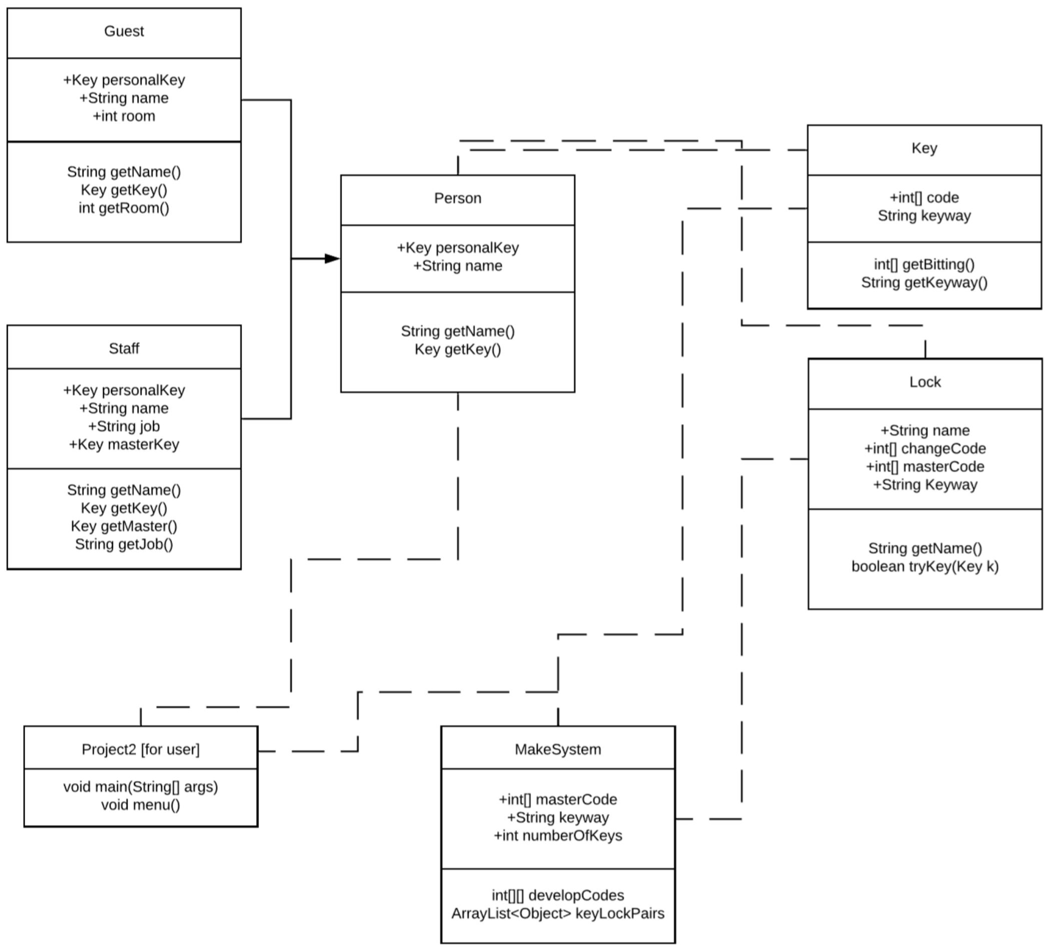
Abstract

This program was created with the intention of creating and simulating a mastered system of pin-tumbler locks as it would be implemented by a locksmith. This paper will explain the specific execution and use of said program.

Introduction

Prior to this project, I had taken up an interest in locksmithing, and the mechanics of pin-tumbler locks. When this was introduced, it seemed natural for me to base my project off of this interest. When doing some initial research, I was looking for used methods of developing bitting codes for mastered key systems, and found that there are no common ways of doing this. When implementing such a system, most locksmiths will simply rely on lock manufacturers to provide codes (and keys) from bitting arrays that are calculated and created by said manufacturers. My program allows for the creation of a simple mastered system without the use of existing bitting array, which can be used to simulate a system in greater detail rather than using placeholders as is commonly done.

Detailed System Description



This system contains realistic representations of physical pin-tumbler lock and key systems and simulates their creation and application in buildings. The user can create a set of non-duplicated change keys and matching mastered locks, as well as create users with varying levels of access and assign change keys to them to simulate potential use of the created system.

Requirements

The most unique part of this program is the development of a bitting array from a master key. This type of resource is generally not freely available, which is a problem I hope to solve with this software.

Literature Review

KeySoft has released a program previously that has similar functionality to mine, however: Their software is not free, and it does not allow for simulation of the use of the system. The main use of their software is to create bitting arrays.

Beyond this program, I couldn't find any software that attempted to address this problem. During my research process, I overwhelmingly found that lock manufacturers are the source of bitting arrays, including a direct response from a locksmith through an online forum.

The remainder of my research was on general practice for developing master keyed systems.

Mathematics of Master Keying
This pdf explains the basic mathematics behind the creation of change keys, and informed the development of my own algorithm.

How Master Key Systems Work by Ralph Goodman
This article is a more broad overview of the terminology behind master key systems explained in layman's terms. This was helpful for my overall understanding of my goal and helping me to structure my classes and methods properly.

Keying Systems and Nomenclature by Door and Hardware Institute
This is a more technical breakdown of convention and practices surrounding the development and implementation of mastered systems. Also helpful for developing the structure of my program.

The Concepts and Mechanics of Master Keying: Developing Master Keys Using Generic Types by Rod Oden
This document served as the main guidelines I used to construct my *DevelopCodes()* method.

These sources represent the bulk of useful information found from my research. I have left my personal comments in my bibliography to show a bit of my own process.

User Manual

The user can interact with the software through the Project2 class, which essentially only consists of the main method, and contains the only main in the package. The class itself if fairly self explanatory. The user is walked through the process of generating change codes from a master code, and can then explore a few options via the main menu. A this point the user has an empty list of keys, and empty list of locks, and an empty list of people. There are two hidden generated lists of corresponding keys and locks respectively. There are 7 options from the main menu:

1) Generate key: this takes the next change key from the generated list and adds it to the user's list.
2) Generate lock: this takes the next lock from the generated list and adds it to the user's list.
3) Create person: this allows the user to create either a staff member or guest and assign them a key from their list of keys.
4) View keys: displays the keys the user has created.
5) View locks: displays the locks the user has created.
6) View people: displays the people the user has created.
7) View master key: displays master key
8) Quit: exits the program.

Conclusion

This program certainly accomplished its goal of generating a mastered system. That said, there are a few things that can be improved with further work. The change code generation algorithm is limited in the number of codes it can produce. This is a significant limitation that can be fixed. Additionally I would, with more time, like to improve the quality of the simulation with the addition of rooms as objects and provide more ways for the user to create interaction between the objects the program offers. However as it stands, I am happy with the product in its current form.

References and Research

https://extranet.assaabloydss.com/library/keysystems/pdf/mathematics_of_mk_webinar.pdf
Basic Mathematics of Master Keying

https://unitedlocksmith.net/blog/how-master-key-systems-work
Introduction to mechanics of physical system

https://www.assaabloyacademy.com/Local/assaabloyacademyCOM/Americas/Lesson%20Resources/2018KeySystemAssessment/1%20General%20Documents%20and%20Interactive%20Video%20-%20Designing%20Master%20Key%20Systems/Keying%20Systems%20and%20Nomenclature_2008.pdf
Longer and more clear document explaining the planning and execution of a key system

So plan:

```
        GMK
         |
    MK1 — MK2
     |        |
    KD       KD
    KD       KD
    KD       KD
    KD       KD
    KD       KD
```

This simulates a building with 2 floors, and 5 rooms on each floor, each of which are keyed differently. Each floor has a master key, and there is a grandmaster key that operates all doors in the building.

https://www.locksmithledger.com/keys-tools/article/10290193/the-concepts-and-mechanics-of-master-keying-developing-master-keys-using-generic-types
To help develop bitting codes, used in main method?

https://www.locksmithledger.com/keys-tools/article/10323366/masterkeying-by-the-numbers
This is better, guide on developing a key bitting array

Bitting generally goes X-Y-Z-A-B for master; change keys will originate along the line of (X±2)-(Y±2)... and so on. Subsequent change keys generally follow AA1 = HIJKL; AA2 = HIJK(L+2); AA3 = HIJK(L+4); and so on.

http://lockwiki.com/index.php/Differs
This is an interesting piece of info for real world locksmithing, but for my purposes I have no reason to get this complex. I will approximate MACS requirements by the method outlined above.