CSC8501 Coursework 2 – 2018
The Safe Problem
Due 26th October 2017 at 10am

Set by Graham.Morgan@ncl.ac.uk

0 1 2 3 4

Specification (what you need to do): You will extend your computer program from coursework 1 to allow the determination of combinations to unlock multi-lock safes

Clarifying the problem (approach to take)

- Your original program must be extended create outputs of valid and locked multi-lock safes. This output must be in the form of a text file known as the *locked safe file*. The output must take the form of the ROOT value followed by the LN values for each combination lock
- A *locked safe file* must be represented by first indicating the number of multi-lock safes that are contained in the file followed by the settings of each safe in turn in the following format (an example, not a real *locked safe file*):

NL 2
ROOT: 4 5 6 2
LN0: 2 5 7 2
LN1: 4 7 3 2
LN2: 6 8 9 1
LN3: 5 0 2 1

ROOT: 3 5 8 1
LN0: 2 3 7 2
LN1: 1 7 3 9
LN2: 3 8 9 1
LN3: 3 9 2 1

Your program

- Create a program that will use a method of your choice (without cheating - i.e., looking at the solution held in the associated multi-safe file or key file) to determine appropriate four digit inputs for each hash function (**UHF**, **LHF**, **PHF**) to generate the **LN** values as displayed in the *locked safe file.* Once this is achieved the **CN** values can be easily derived (they are actually derived as a step in this process)

- Your output should be a *multi-safe file* (as in coursework 1) showing the solutions and the *key file* (as in coursework 1) showing how they were derived from the original *locked safe file*. These can then be checked again for validity using coursework 1
- Finally, extend your solutions in coursework 1 and coursework 2 to allow a user to specify the number of locks that are present in a multi-lock safe
- How quickly can your program run?

Deliverables (what we want to see submitted):

- C++ source code authored by the student
- Executable file containing solution
- A sample of no more than 10 key files and associated multi-safe files
- Demonstration (your chance to explain and show your solution):
- On Friday 26th October from 10am onwards students will demonstrate their
- solutions

Learning Outcomes (what we expect you to demonstrate in a general way)

- To be able to design and create programs
- To be able to identify appropriate techniques for analysing the efficiency of programs
- To be able to realise inappropriate usage of programming languages
- To be able to manage memory
- To be able to create and use data structures
- To be able to use condition statements, loops and functions
- To be able to utilise concurrency when appropriate
- To be able to create programs that handle run-time errors
- To be able to use appropriate techniques for debugging and analysing existing algorithms
- To be able to design programs using a well known methodology

Marking Scheme (what is worth what)

Marks are out of 25 and are awarded as follows:

- 10 Marks for achieving correct output (5 lock multi-safe)
- 5 Marks for appropriate file input and output (5 lock multi-safe)
- 5 Marks for extending solutions to allow multi-lock safes of any length
- 5 Marks for advanced features (e.g., templates, threading)
- 10 Marks for achieving correct output

Additional direction (making the task clearer)

- **Task 1**: Start with a single combination lock with only one hash input value missing
- **Task 2**: Increase the complexity gradually to tackle larger problems (more locks)
- **Task 3**: Use the locked safe files of others so you can validate your program
- **Task 4**: Try to integrate multi-threading into your solutions
- **Task 5**: Provide a user interface for use at the demonstration to make life easy