**Passcode:**

- Code’s format = startTime (current time) + endTime

- Encryption method:

- startTime + 1000000999

- endTime + 10870888

- (This is because the time format is something like “1704869007”, and I’m worried that a simple combination of them would make the code very easy to be tracked down.)

- Then, combine them into a 20 digits number.

- (why 10 digits for start and end time: this is because in order for them to increase to 11digits, at least 263 years are needed or so on, so it’s not necessary to be concerned of.)

- To make the numbers more ‘encrypted’:

- using ascii table, for characters at index bigger than 53, their position minus 6, and for the rest of the characters, their index is added by 19.

**For activation and time updates:**

- At the client’s side, two txt files are needed: one to stored time information and the other one to stored the codes that have been used.

- The two files are stored in C:/’user’/programdata/roaming. This is to isolate them from the software. Hence, multiple installation/uninstallation would not affect the codes that were used before.

- The two files are encrypted using the same method as the one used for the code: using ascii table and change the original character. Using same key, which is the integer that used to move up or down the able.

- For the one that stores time information, there would be two lines: startTime and endTime.

- startTime would be keep updating as the software runs.

- For time updates:

- When a new code is passed into, the difference between the start and end time in the code, after decryption, is accumulated to the endTime in the txt file.

- For activation:

- same as above, just that the endTime in the txt should be changed to the current time.

- Whenever a code is used successfully, it is stored inside a file. This is to prevent multiple usage.

- Since the code is made of startTime and endTime, users can easily fake one.

- For example, if the code is 17846899, which represents 360seconds. If a user wants to try to add or minus from the code and result in 17846898, which represents 359seconds, and this is actually a valid code.

- Hence, a filter is used at this point, such that the code must be divisible by 10. This is because the code generated is always ended with 0.

**For setting the passcode:**

- Convert the time user inputted to seconds, and create a startTime and endTime.

- Encrypt (has discussed above) and output it.