What is this program?

This program is meant to simulate a security system by using the concepts of state machines and finite automata. There is a locking and unlocking code, and only when you enter those codes, the program will change its status from locked to unlocked and vice versa. You can input one character at a time, or you can input a string of characters if there is a single space between each character.

This program was built and tested in Windows 11 using VSCode

Building an executable

Windows 11

If you are using Windows 11, first install a compatible version of the GNU C/C++ compiler (such as MinGW). Open the command prompt, and CD into the directory where the git repository was cloned into. Then, CD into CS330-Programming-Assignment. Lastly, run the command:

g++ project.cpp -o [FILE NAME].exe

Ubuntu 22.04

If you are on Linux, open the terminal. If you haven't already, install GCC and G++ compilers. Then, CD into the directory where the repository was cloned, then CD into CS330-Programming-Assignment. Lastly, run the g++ command shown under the windows section

This will generate an executable file. You can then run the program by either double-clicking the exe file or typing "[FILE NAME].exe" into the terminal.

Unlock Code: 236211
Lock Code: 236214

Testing Security:

Since the unlock code is 6 digits, and there are 10 possible numbers for each digit, that means there are 1 million possible 6 digit numbers (10^6). This means that on average, if you only typed in integers and if it took you 1 second to verify that the program unlocked, it would take you 1 million seconds. This is equivalent to about 11.6 straight days of inputting numbers

To back this up, I ran a program that input random digits into the security device, and counted how many digits it had to generate before it unlocked. These were the results:

With 100 trials:
On average, 844,838 digits were generated
The most digits generated was 3,972,522
The least digits generated was 1,411

With 1000 trials:
On average, 974,045 digits were generated
The most digits generated was 7,646,982
The least digits generated was 5,435

I will include the excel file for these tests in the repo