CMP SCI 5732 Project 2 Report

Jamie Harris

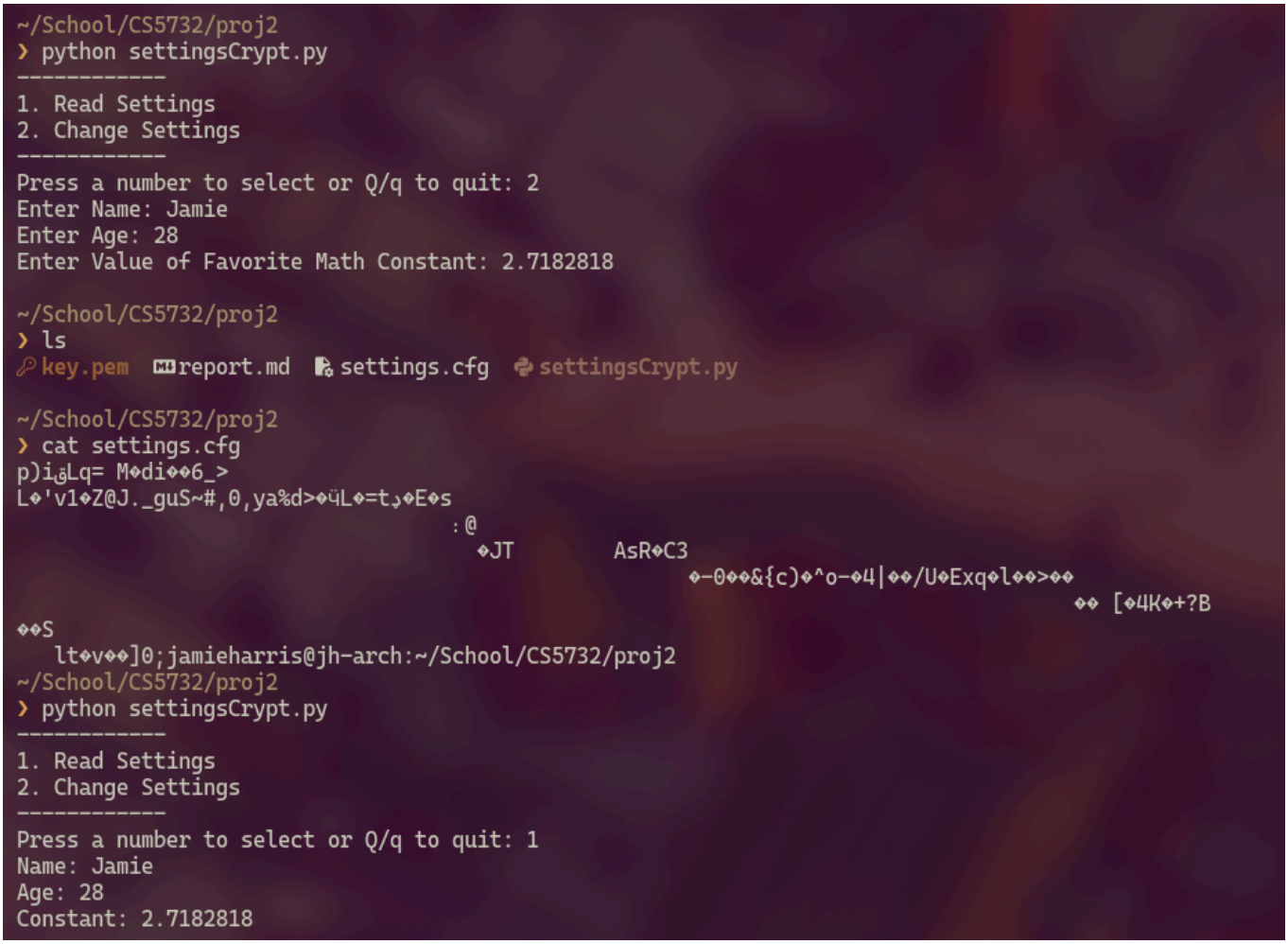
For this project, I used python 3.13 with the cryptography library. The program uses RSA encryption, with a 2048-bit key and a public exponent of 65537. Most function calls are taken straight from the cryptography library documentation. Below is the source code

```
# settingsCrypt.py
import os
from cryptography.hazmat.primitives.asymmetric import rsa,
padding
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives import serialization
from cryptography.hazmat.primitives.asymmetric.types import
PrivateKeyTypes

def generate_key():
    key = rsa.generate_private_key(65537, 2048)
    private_key_bytes =
key.private_bytes(serialization.Encoding.PEM,
serialization.PrivateFormat.TraditionalOpenSSL,
serialization.NoEncryption())

    with open(file="key.pem", mode="wb") as key_file:
        key_file.write(private_key_bytes)
    key_file.close()

def load_key():
    with open(file="key.pem", mode="rb") as key_file:
        private_key =
serialization.load_pem_private_key(key_file.read(),
password=None)
```

```
        public_key = private_key.public_key()
    key_file.close()
    return (private_key, public_key)



def print_menu(private_key, public_key):
    print("------------")
    print("1. Read Settings")
    print("2. Change Settings")
    print("------------")
    choice = input("Press a number to select or Q/q to quit:
")
    while choice not in ["1", "2", "Q", "q"]:
        print("Error: Bad Input")
        print("------------")
        print("1. Read Settings")
        print("2. Change Settings")
        print("------------")
        choice = input("Press a number to select or Q/q to
quit: ")

    if choice == '1':
        print_settings(private_key, public_key)
    elif choice == '2':
        change_settings(private_key, public_key)

def change_settings(private_key, public_key):
    name = input("Enter Name: ")
    age = int(input("Enter Age: "))
    fav_constant = float(input("Enter Value of Favorite Math
Constant: "))

    settings=str.encode(f"Name: {name}\nAge: {age}\nConstant:
{fav_constant}")

    with open(file="settings.cfg", mode="wb") as
```

```
settings_file:
        encrypted = public_key.encrypt(settings,
padding.OAEP(
            mgf=padding.MGF1(algorithm=hashes.SHA256()),
            algorithm=hashes.SHA256(),
            label=None
        ))
        settings_file.write(encrypted)

    settings_file.close()

def print_settings(private_key: rsa.RSAPrivateKey,
public_key):
    settings_encrypted = open(file="settings.cfg", mode="rb")
    settings_decrypted =
private_key.decrypt(settings_encrypted.read(), padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA256()),
        algorithm=hashes.SHA256(),
        label=None
    ))

    settings_decrypted = settings_decrypted.decode()

    print(settings_decrypted)

    settings_encrypted.close()

if __name__ == "__main__":
    if os.path.isfile(os.getcwd() + "/key.pem"):
        (private_key, public_key) = load_key()
        print_menu(private_key, public_key)
    else:
        generate_key()
```

Outputs


~/School/CS5732/proj2
❯ python settingsCrypt.py
1. Read Settings
2. Change Settings
Press a number to select or Q/q to quit: 2
Enter Name: Jamie
Enter Age: 28
Enter Value of Favorite Math Constant: 2.7182818
~/School/CS5732/proj2
❯ ls
key.pem report.md settings.cfg settingsCrypt.py
~/School/CS5732/proj2
❯ cat settings.cfg
~/School/CS5732/proj2
❯ python settingsCrypt.py
1. Read Settings
2. Change Settings
Press a number to select or Q/q to quit: 1
Name: Jamie
Age: 28
Constant: 2.7182818