Writeup

Overall Goal: Create 3 programs and supporting files

Keygen: Generate a pair of RSA keys with optional user specifications

Encrypt: Encrypt a file using a given key.

Decrypt: Decrypt a file using a given key.

Supporting Files rsa.c, numtheory.c, randstate.c, MAKEFILE

With multiple different files having to be made, I decided to make my own test.c file that I would use to test my code.

One of the main reasons I did this, is because I wanted to be able to test every function/method that I made without affecting the files themselves. If I had made a “main” function in all of these files to test, I would have had to remove them when I turned in the assignment and I would not be able to test my code after that. By having a separate test file, I am able to keep the assignment in a state where I can turn it in and still be able to test my code.

By having a separate file, I would also be able to test how different functions interacted with each other as I could include all of my other files to call them in any way I needed in order to properly test them. If I had tried to do this in the main method, I would need to go back and remove header files and generally do cleanup throughout all of my files which would be extremely time-consuming.

How I actually tested the program

Since this project was quite large, I decided to test my code in increments. After finishing a function, I would immediately test it with a bunch of different values to make sure that it worked.

I found this to be extremely important because I would have other functions that would call each other. If I decided to test after I had finished the entire project in one go, it would have been extremely hard to track down where the bug was actually occurring. In addition, working one function at a time gave me a starting point as it was a bit overwhelming when looking at the entire project.

By focusing on a single method one at a time, it allowed me to narrow my focus and stay on track. This combined with having my own testing file meant that I could also go back and re-test without much issue at all which I found to be extremely handy when testing the functionality of the entire function.

Final Testing

Once I had my project mostly working, I decided to download the *-dist files from Piazza. This would allow me to directly compare functionality from my code to what would be expected. This allowed me to find interactions that I had implemented and discover how I should handle certain errors such as out-of-bounds inputs. Having access to the -dist files made it much easier to give the final polish to my program as I knew exactly what was going to be expected from me.

This also allowed me to match usage outputs and other error messages in order to make sure that I had the exact messages.

I was also able to generate keys from my program and use them to encrypt and decrypt from the distributed program and vice versa. This really allowed me to test my code in depth and find quirks that I do not think I would have been able to find otherwise.

Overall, I tested my program incrementally in a separate test file, then used the distributed binary files to polish the programs.