**RSA Encryption**

For this exercise, your task is to implement parts of the RSA Encryption algorithm. RSA is one of the most important algorithms used to establish encrypted connected between a server and a computer. The algorithm is an application of simple number theory concepts, but nonetheless a secure means of encrypting data. It is used in enterprise applications to establish Virtual Private Connections to ensure anyone who intercepts a connection, at the very least, cannot decrypt the data. Typically, the algorithm utilizes very large prime numbers, but for this exercise, we will just use two small prime numbers. Particularly, 3 and 5. Consequently, the smaller prime numbers used in the algorithm, the less encryption there is to the data.

For simplicity, the full detail of the algorithm will not be discussed here. Instead, only the parts that you need to implement, and the final output expected will be detailed here.

**TASK**

In the zip file provided, there are 3 files in it:

* LASTNAME.c (TO BE SUBMITTED)– This is where all your solutions will go. Rename this file according to your last name. Do not put spaces in it. Please see comments found in LASTNAME.c for the specific details of your tasks.
* RSAEncryption.c (TO BE SUBMITTED) – Modify only the third line of this file, corresponding to the version of your LASTNAME.c file. Main() function is located in this file. Compile only this file and not the other two files when testing.
* Util.c – Do not modify any part of this code. Do not submit this code.

Ensure that all three \*.c files are on the same folder before compiling. Make sure your code compiles using the following command before submitting:

gcc -Wall -std=c99 RSAEncryption.c -o RSAEncryption.exe

Codes that will not compile using the command above will merit a **grade of 0** for this exercise.

**SAMPLE RUN 1**

Enter a message: 103050709 🡨 This is user input.

Encrypted: 112051309

A\*I\*O

Decrypted: 103050709

ANIMO

Message received!

**SAMPLE RUN 2**

Enter a message: 2040608 🡨 This is user input.

Encrypted: 8040602

ULSD

Decrypted: 2040608

DLSU

Message received!

**SAMPLE RUN 3**

Enter a message: 10302 🡨 This is user input.

Encrypted: 11208

A\*U

Decrypted: 10302

AND

Message received!

\*\* Not all messages will be encrypted correctly. There are some combinations that will cause the program to crash. For safe measure, limit your message to 3 letters at most (see table found in getLetter() description in LASTNAME.c for the list of letters you can use)