Daniel Aldridge
CS_485
Project1

Run Instructions

To compile this program:

```
dja0005@linux002:~/CS_485$ make
```

Type the command **make** to run the makefile and compile all the associated source files with the header.

To run this program:

```
dja0005@linux002:~/CS_485$ ./project1
```

Type the command **./project1** to run the program

This project will then ask for the user's alphabet choice as was described in the project1 guidelines

```
dja0005@linux002:~/CS_485$ ./project1
Are we using an all uppercase alphabet(modulo = 26, enter 'S' or both lowercase and uppercase(modulo = 52, 'L'))
```

The user types '**S**' for uppercase and '**L**' for lower and uppercase alphabet.

The program then reads the text from plaintext.txt and the key from vcipherkey.txt. It checks to make sure that both the key and plaintext are compatiable with the alphabet chosen. It also then pads the key to match the length of the plaintext. It then performs the encryption on the plaintext using the vigenere encrypter and writes this to vigenerecipheroutput.txt. It also then prepares the cipher text for the block affine encryption by splitting the text into blocks of two. These steps are shown below….

```
Reading plain text from plaintext.txt
Plain Text: LORdILLIDANknowsTHEWAY

Reading key for vigenere from vcipherkey.txt
Key: Daniel

Padding key to match length of plain text so encryption can be performed properly
Padded Key: DanielDanielDanielDani

Performing Vigenere Encryption on plain text now...
Writing Vigenere Encrypted text to vigenerecipheroutput.txt
Vigenere Encrypted Text: OoELmwOiqirVqOjaxsHwnG

Perfoming Affine Encryption now..
Dividing up the vigenere cipher text into blocks of two
Padded Cipher text Oo EL mw Oi qi rV qO ja xs Hw nG
Enter the desired multiplier
```

Next it asks the user's input for the block affine encryption, namely the user inputs a multiplier and offset. The program also checks to make sure the multiplier and modulo are co-prime as expected from the project guidelines. If they are not co-prime the program exits

```
Enter the desired multiplier
4
The Values are not coprime...exiting program...
dja0005@linux002:~/CS_485$
```

Otherwise the program proceeds on and asks the user for the offset.

```
Enter the offset
1
Writing Block affine encrypted text to blockaffinecipheroutput.txt

Perfoming Block Affine Decryption now...
Inverse Modulo found
Writing block affined decrypted text to blockaffinecipherplaintextouput.txt
Affine Decrypted cipher text: Oo EL mw Oi qi rV qO ja xs Hw nG

Perfoming Vigenere Decryption Now...
Writing final decrypted text to secondplaintext.txt
Final text: LORdILLIDANknowsTHEWAY
```

The program once it has a valid multiplier and offset will perform the block affine encryption and write this text out as numbers to blockaffinecipheroutput.txt

```
4317 1334 1141 4351 2351 2612 2343 0227 4429 2241 1419
```

Where each Letter is represented as two digits 'i.e 1 = 01' and it still groups the "text" or in this case numbers into blocks of two.

Next the program will perform the block affine decryption by finding the inverse modulo and writing the decrypted text from this step to blockaffinecipherplaintextouput.txt.

Lastly the program will execute the Vigenere Decryption and write the final text which should match the original plain text into secondplaintext.txt

As We can see they match from this trial run