Assignment 3

Milestone 1

Eli Fonseca

10/4/15

\*\*I am choosing the Cryptography program

List any assumptions you will be making

Assuming:

* The user will be using the English alphabet when inputting phrases to be encoded and decoded

List all methods you plan to use, including the inputs and outputs for each method

1. affineCipher – takes a and b and a String in and returns an encrypted string (using affine cipher)
2. affineResults – takes an encoded string and a decoded string and prints the two strings to the console. Also does a test to see if the original phrase matches the decrypted phrase and prints the result of this test to the console
3. aPrime – takes in a and calculates and returns aprime which will be used for the decrypting process for the affine cipher
4. decryptionFunctionAffineCipher – takes a b and a string and returns a the decrypted version of the entered string
5. explainer – prints an explanation of what the program will do
6. promptForA – takes scanner object and prompts the user for a valid value of a. Then returns value of a.
7. promptForB – takes scanner object and prompts the user for a valid value of b. Then returns value of b.
8. rot13 – takes string and encrypts it using rot13 then returns the string
9. rot13Results – takes the result of rot13 encryption, result of rot13 decryption, and original phrase. Prints the encoded and decoded phrases. Also performs a test to see if original phrase is the same as the decoded phrase and prints result of test to the console.
10. succesTest- takes original inputted phrase and the decrypted phrase (using either affine or rot13). Then tests to see if the two strings are the same. Prints the result of this test.
11. testA- takes in a. Then tests to see if user entered valid value of a. (needs to be coprime with 26 for a decryption function to exist) if user entered incorrect value then prompts the user for a new value for a. Then returns new value of a.
12. testB takes in b. Then tests to see if user entered a valid value for b (needs to be between 0-26) if user entered an invalid number then re-prompts the user for a new value of b.

\*\*Note the error testing and re-prompting for a and b will not be turned on when I turn in the assignment. Instead the program will simple stop executing if the user enters invalid inputs for a or b.

 List the order in which your methods will be called

explainer

promptForA

* + testA

promtpForB

* testB

rot13

rot13

affineCipher

decryptionFunctionAffineCipher

affineResults

* successTest

rot13Results

* successTest

Extra stuff for the affine cipher:

To encrypt using affine cipher I will run each character of the phrase through the following equation after subtracting the its corresponding (if it is uppercase or lowercase) ascii value for A.

\mbox{E}(x)=(ax+b)\mod{m}, m = number of letters in alphabet (so for our cause 26)

E(x) is the encoded string, a is the multiplicative factor, x is the index of the character (0-25), and b is the shift factor. After finishing running the character through the equation I will add back the value of A I subtracted in the being so the correct letter is displaced to the console when printed.

To decrypt a affine cipher encrypted phrase I will run each character of the phrase through the following equation after subtracting it’s corresponding ascii value for A.

\mbox{D}(x)=a^{-1}(x-b)\mod{m}, m = 26

D(x) will be the decoded phrase, a^-1 will be the modular multiplicative inverse of a, x will be the index of the character, and b will be the shift factor originally used for encoding. After finishing running the character through the equation I will add back the value of A I subtracted in the being so the correct letter is displaced to the console when printed.

To handle the possibility that the a^-1(x-b) part of the equation will go negative (which It obviously will) I will check the index to see if it is negative before I mod m it. If the index is negative instead of simple taking mod m of the index I will do m + index mod m. This takes care of the problem java has with negative number and mod.

I will calculate a^-1 base off the fact that it has to obey the following conditions.

a\,x \equiv 1 \pmod{m}. x being a^-1

And that the multiplicative invers will be included in the ring of integers modulo *m*, denoted\mathbb{Z}_m. That is to say that to say that a^-1’s first multiple will be between 1 and 25 since we are using a 26 letter alphabet.

So when the user input a specific value for a I will test it with every integer 1-25 by using the equation above. If **that** equation ever becomes equal to 1 then I will have found my a^-1.