Montgomery College
CMSC 203
Assignment 3 Design

Class: CMSC203 CRN XXXX

Program: Assignment 3 Design

Instructor: Grinberg

Summary of Description: Caesar and Bellaso encryption and decryption

Due Date: 10/3/2022

Integrity Pledge: I pledge that I have completed the programming assignment independently.

I have not copied the code from a student or any source.

Student: Michael Bushman

Part 1: Pseudo Code:

Turn in pseudo-code for each of the methods specified in CryptoManager.java.   Refer to the **Pseudocode Guideline** on how to write Pseudocode.

Public boolean isStringInBounds

Declare boolean inbound equals true

Repeat n times

    Declare int a equals int plainText.charAt

    if int a is less than LOWER_RANGE or int a greater than UPPER_RANGE

    inbound equals false

Return inbound

Public String caesarEncryption

If isStringInBounds is not true

    return The selected string is not in bounds, Try again

Declare String caesarEncryption

Repeat n times

    int a equals int plainText charAt plus key

    while int a greater than UPPER_RANGE

int a equals int a minus RANGE

    caesarEncryption equals caesarEncryption plus (char) a

Return caesarEncryption

Public String bellasoEncryption

Declare String bellasoEncryption

Repeat n times

    Declare int a equals int plainText.charAt plus bellasoStr.charAt

    while int a is greater than UPPER_RANGE

        int a equals int a minus RANGE

    bellasoEncryption equals bellasoEncryption plus (char)a

Return bellasoEncryption

Public String caesarDecryption

Declare decryptionOfCaesar

Repeat n times

    int a equals encryptedText charAt minus key

    while a is less than LOWER_RANGE

        a equals a plus RANGE

    decryptionOfCaesar equals decryptionOfCaesar plus (char)a

Return decryptionOfCaesar

Public String bellasoDecryption

Declare String bellasoDecryption

 Repeat n times

    Declare int a equals int plainText.charAt minus bellasoStr.charAt

    while int a is less than LOWER_RANGE

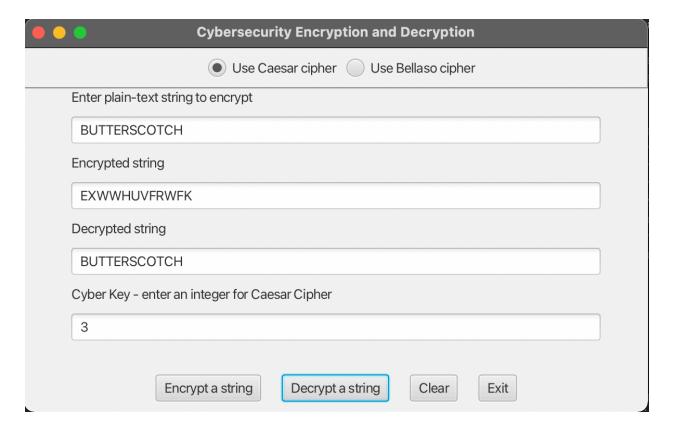   int a equals int a plus RANGE

    bellasoDecryption equals bellasoDecryption minus (char)a
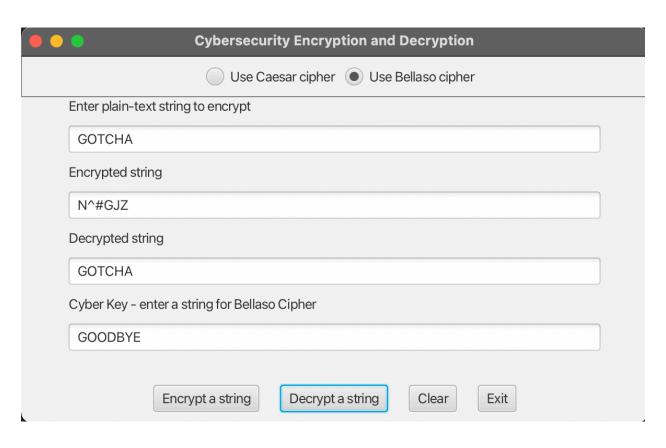
Part 2: Comprehensive Test Plan

Turn in a Test Plan table. Test Plan should include:

- at least two tests for the Caesar Cipher

- at least two for the Bellaso Cipher.

- at least one string that will fail because it has characters outside the acceptable ones.

Input text	Input Key	Encrypted (method1)	Encrypted (method2)	Decrypt (method1)	Decrypt (method2)
BUTTERS COTCH	3	EXWWHUVFR WFK	5HGG8EF6BG6;	BUTTERSCOT CH	BUTTERSCOTCH
HELP	89	!^%)		HELP	
GOTCHA	GOODBYE		N^#GJZ		
CMSC203	ROCKS		U\VNEBB		
THIS SHOULD FAIL SINCE { ISNT VALID	N/A	N/A	N/A	N/A	N/A

Cybersecurity Encryption and Decryption

(●) Use Caesar cipher   ( ) Use Bellaso cipher

Enter plain-text string to encrypt

```
BUTTERSCOTCH
```

Encrypted string

```
EXWWHUVFRWFK
```

Decrypted string

```
BUTTERSCOTCH
```

Cyber Key - enter an integer for Caesar Cipher

```
3
```

| Encrypt a string | Decrypt a string | Clear | Exit |

Cybersecurity Encryption and Decryption

(●) Use Caesar cipher   ( ) Use Bellaso cipher

Enter plain-text string to encrypt

```
HELP
```

Encrypted string

```
!^%)
```

Decrypted string

```
HELP
```

Cyber Key - enter an integer for Caesar Cipher

```
89
```

| Encrypt a string | Decrypt a string | Clear | Exit |

Cybersecurity Encryption and Decryption

○ Use Caesar cipher   ● Use Bellaso cipher

Enter plain-text string to encrypt

> GOTCHA

Encrypted string

> N^#GJZ

Decrypted string

> GOTCHA

Cyber Key – enter a string for Bellaso Cipher

> GOODBYE

[Encrypt a string] [Decrypt a string] [Clear] [Exit]

Cybersecurity Encryption and Decryption

○ Use Caesar cipher   ● Use Bellaso cipher

Enter plain-text string to encrypt

> CMSC203

Encrypted string

> U\VNEBB

Decrypted string

> CMSC203

Cyber Key – enter a string for Bellaso Cipher

> ROCKS

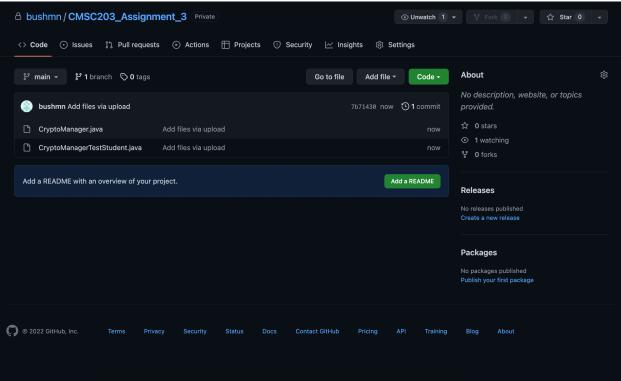[Encrypt a string] [Decrypt a string] [Clear] [Exit]

What have you learned?

I have learned to always watch your video breakdown and to always reread the document for key messages. The wrap around confused me the first time I was reading about it. But then once

I got further down into the document, you talked about wrapping the text by seeing if the key was greater than the upper limits, you would take the key minus the range.

What did you struggle with?

I struggled the most with the Bellaso encryption. The method was very similar to the Caesar encryption where you add the plainText value to the offset key (in this case it would be the bellasoString) and then that's your ASCII values. The issue I was having was using the same integer value (in the for loop) for plainText and the bellasoString. This is incorrect because the for loop runs through the plainText length and not the length of the bellasoString. So, we have to use the integer value modulo the length of the Bellaso string. If we don't do this, the code will continue adding ASCII values for bellasoString for the whole length of plainText even if the bellasoString is over with. We need the adding of the bellasoString to stop once we have reached the end length of the entered bellasoString.

Extra thoughts:

Your video breakdown really helped me on this one. You broke it down to where I understood that you had to add the plainText to the key or plainText to the bellasoString. Only based on the document, I had a vague idea of what to do. Once I watched your video on the assignment, everything made a lot more sense.