Assignment 3

Document

Kobie Marsh

Dr. Grinberg

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input text | Input Key | Encrypted (method1) | Encrypted (method2) | Decrypt (method1) | Decrypt (method2) |
| Kobie Marsh | 423 | 26)0,G4(9:/ |  | KOBIE MARSH |  |
| Gonna test Bellasso | Test this |  | [T!”!4\N&(%UY, I\&# |  | GONNA TEST BELLASSO |
| Test for out of bounds {} | 11111111 | [LZ['MVY'V\['VM'IV\UKZ'BD |  | TEST FOR OUT OF BOUNDS ;= |  |
| Test for long string key | Hello there please work I need this to work |  | \J\_ /&#Z%^T.W,XU%N.^/]P9 |  | TEST FOR LONG STRING KEY |

# Test table

# Pseudocode

Import utilities

Public class CryptoManager

Declare and Initialize constant character LOWER\_BOUND to ‘ ‘

Declare and Initialize constant character UPPER\_BOUND to ‘\_’

## Create method string in bounds stringInBounds that accepts a string plaintext

Declare and initialize boolean trueOrFalse to false

For int i is 0; i is less than the length of the string length; i increments

Declare and Initialize char ascii to the character of the element i

Declare and Initialize lowerB to LOWER\_BOUND

Declare and Initialize upperB to UPPER\_BOUND

If ascii is greater than or equal to lowerB and ascii is less than or equal to upperB

Initialize trueOrFalse to true

Else

Initialize trueOrFalse to false

break

Return trueOrFalse

## Create method string encryptCaesar that accepts a string plainText and an int key

Create the character arraylist encryptedChar

Declare and initialize encryptedString to null

If key is less than the int casted LOWER\_BOUND or key is greater that the in casted UPPER\_BOUND

Do

If key is greater than the int casted LOWER\_BOUND

Key is equal to key subtracted by the range

Else if key is less than the int casted UPPER\_BOUND

Key is equal to the key added by the range

While key is less than the int casted LOWER\_BOUND or key is greater that the in casted UPPER\_BOUND

For int i is 0; i is less than the length of the string length; i increments

Declare and Initialize int ascii to the character of the element i

Ascii is equal to ascii plus key

If ascii is less than the int casted LOWER\_BOUND or ascii is greater that the in casted UPPER\_BOUND

Do

If ascii is greater than the int casted LOWER\_BOUND

Ascii is equal to key subtracted by the range

Else if key is less than the int casted UPPER\_BOUND

ascii is equal to the key added by the range

While ascii is less than the int casted LOWER\_BOUND or ascii is greater that the in casted UPPER\_BOUND

Add the character casted ascii to encryptedChar

encryptedString equals string.valueOf the encryptedChar

format encryptedString

return encryptedString

## Create string method encryptedBellaso that accepts plaintext and string bellasoStr

Create arraylist encryptedChar

Declare and Initialize encryptedString

Declare and initialize k to 0

For int i is 0; i is less than plaintext length; i increments

If k is less than equal to bellasoStr length minus 1

i decrements

k is 0

continue the loop

else if k is less than or equal to bellasoStr length minus 1

declare and initialize to int casted bellasoStr character k then k increments plus int casted plaintextcharacter at i

if addChar is lower than int casted LOWER\_BOUND or addChar is greater than int casted UPPER\_BOUND

do

if(addChar is greater than int casted UPPER\_BOUND)

addChar is equal to addChar minus RANGE

else if(addChar is less than int casted LOWER\_BOUND)

addChar is equal to addChar plus RANGE

while(addChar is less than int casted LOWER\_BOUND or addChar is greater than int casted UPPER\_BOUND)

add char casted addChar to encryptedChar

initialize encryptedString to the string value of encryptedChar

format encryptedString

return encryptedString

## Create method String decryptCaesar that accepts String encryptedText and in key

Initialize and declare decryptedString to null

Create arraylist decryptedChar

If key is less than the int casted LOWER\_BOUND or key is greater that the in casted UPPER\_BOUND

Do

If key is greater than the int casted LOWER\_BOUND

Key is equal to key subtracted by the range

Else if key is less than the int casted UPPER\_BOUND

Key is equal to the key added by the range

While key is less than the int casted LOWER\_BOUND or key is greater that the in casted UPPER\_BOUND

For int i is 0; i is less than the length of the string length; i increments

Declare and Initialize int crypt to the character of the element i

crypt is equal to ascii minus key

If crypt is less than the int casted LOWER\_BOUND

Do

Crypt is equal to crypt plus RANGE

While crypt is less than the int casted LOWER\_BOUND

Add the character casted ascii to decryptedChar

decryptedString equals string.valueOf the decryptedChar

format decryptedString

return decryptedString

## Create method string decryptBellaso that accepts string encryptedText string bellasoStr

Create arraylist decryptedChar

Declare and Initialize decryptedString

Declare and initialize k to 0

For int i is 0; i is less than encryptedText length; i increments

If k is less than equal to bellasoStr length minus 1

i decrements

k is 0

continue the loop

else if k is less than or equal to bellasoStr length minus 1

declare and initialize to int casted bellasoStr character k then k increments subtracted from int casted encryptedTextcharacter at i

if addChar is lower than int casted LOWER\_BOUND or addChar is greater than int casted UPPER\_BOUND

do

if(addChar is greater than int casted UPPER\_BOUND)

addChar is equal to addChar minus RANGE

else if(addChar is less than int casted LOWER\_BOUND)

addChar is equal to addChar plus RANGE

while(addChar is less than int casted LOWER\_BOUND or addChar is greater than int casted UPPER\_BOUND)

add char casted addChar to decryptedChar

initialize decryptedString to the string value of decryptedChar

format decryptedString

return decryptedString

# Learning Experience

This program was daunting at first, actually making a program that both encrypts AND decrypts seemed like a tough challenge, albeit an interesting one. However, as I was making the program it felt much easier than I originally thought. That is, until I used the provided JUnit from the assignment documents. For some reason my StringInBounds method kept failing the tests.

For some reason the problem kept eluding me however, no matter what I changed it kept showing me that vexing red bar in the JUnit test. I tried changing the variable types, tried rearranging the if…else statements, and even went as far as to force the boolean to be false just to make sure it wasn’t some JUnit glitch. Of course none of those were the reasons for my failure. Instead the solution was a mind-numbingly simple one, I just needed to break when the boolean equals false. Apparently I had it set up to when a character was out of bounds it did actually set the boolean to false, however if the next character was in bounds then it would change the already existing boolean variable to true again, thus ignoring that the string was out of bounds. Now I might not be a professional programmer in any sense of the word, but I still think this mistake was juvenile even for a novice programmer such as myself.

In the end, if I learned anything from this project is that there is always a chance to miss a simple solution to a simple problem, and that a simple mistake can derail a whole program. It also showed me the usefulness of JUnit, since testing with JUnit is much easier than just running the program over and over again.