Encrypting and Decrypting Messages

Due 11:59pm, two weeks from the date of your handout

Objectives:

- Gain more experience with String methods
- Manipulate arrays
- Use arrays as parameters and return types in methods

You and a friend would like to send each other encrypted messages. Write a program that will encrypt and decrypt messages. The program should have 5 different options for encrypting a message.

When the program first starts it should ask first the user to enter

1. to decrypt a message
2. to encrypt a message.

The program then asks the user to enter:

1. for an entire message reversal encryption
2. for a Pig-Latin encryption
3. for a simple Caesar style encryption

If option 3 is picked the program should then ask the user to input a key for encryption. The program should then ask the user to enter the message to encrypt or decrypt.

The program should display the result to the user and ask them to enter:

1. to continue with the program (the program will start over from the beginning)
2. to quit the program

Rules:

Your program must contain the following methods.

a) reverseEncryptandDecrypt

This method performs the entire message reversal. It takes a char [] as an input and returns a char [] that is the reverse of the inputted char array. It can be used to both encrypt and decrypt messages for the entire message reversal encryption option.

For example if the input char array was {'o' , 'n' , 'e'} the method would return a char array that was {'e' , 'n' , 'o' }

b) pigLatinEncrypt

This method encrypts a message using Pig Latin. It takes a String as an input and returns a String that is the inputted String encrypted using Pig Latin. You may assume that the messages do not contain any punctuation. However, you must not convert digits. For example, if 345 appears in a message, it must unchanged in the encrypted message.

The rule for Pig Latin are as follows. A word that does not start with a vowel **('a',' e' ,'i', 'o', 'u' and for** simplicity **'y')** has its first letter removed and then added to the end of the word along with the letters “ay”.

pig → igpay
trash → rashtay
duck → uckday

for words that begin with the vowel just add the letters “yay” to the end of the word.

eat → eatyay
are → areyay
omelet → omeletyay

NOTE:

You will need to split the String into words using the String method **split**, with “ ” as the argument (a String consisting of just a space) .

Some other useful methods in the String class will be the **chartAt** method and

the various **substring** methods.

Another helpful method would be **Character.isDigit()**

c) isVowel()

This method takes a character **('a',' e' ,' i', 'o', 'u' and for simplicity 'y')** as an input and returns a boolean, which is true if the character is a vowel and false if it is not.

d) pigLatinDecrypt

This method takes a String that is encrypted using Pig Latin as an input and returns the decrypted String. See Part b for the rules of Pig Latin.

e) simpleEncryp

This method takes a char [] and an int as an inputs. The integer is the key used to encrypt the message stored in the char []. To perform the encryption simply and the integer to each char in the array (recall that char data types are stored as numbers in memory). You will need to cast the result of the addition back to a char though. The method should return a char [] that is the encrypted version on the inputted char [].

For example: Given string is "HELLO WORLD"

Its ASCII equivalent is [72 69 76 76 79 32 87 79 82 76 68]

After encoding it by adding displacement in ASCII code of 10 it will become

[82 79 86 86 89 42 97 89 92 86 78] or [R O V V Y * a Y \ V N]

f) simpleDecrypt

This method takes a char [] and an integer as an inputs. The integer is the key used to decrypt the message stored in the char []. To perform the decryption, simply do the reverse of the encryption by subtracting the key from each char in the array. The method should return a char [] that is the decrypted version of the inputted char []

For example: Given encoded string is ROVVY*aYVN

Its ASCII equivalent is [82 79 86 86 89 42 97 89 92 86 78]

After decoding it by subtracting displacement in ASCII code of 10 it will become
[72 69 76 76 79 32 87 79 82 76 68] or [H E L L O   W O R L D]