CSCI 1100 – 2017
Computer Science I

Assignment 2
Due: June 5, 2017 at 11:59pm
Submit on Brightspace

Remember that assignments are intended to be your <u>own</u> work!
It is okay to ask for help, to talk with TAs about a problem, or talk with your classmates about a problem.
It is <u>not</u> okay to submit a solution that is substantially the same as somebody else's solution.

Remember the golden rule of programming plagiarism: talk it out, but take no notes. Walk away for 30 minutes. Then, whatever you can remember after that break is yours; use it.

If you are in doubt, ask before submitting!

Question 1.
Include all of the following work in a single class named `Question1`. Use appropriate comments for all custom methods. It is your responsibility to ensure your code compiles before you submit!

Write several methods that deal with Strings. Be aware of whether each method is asking you to take input from parameters or take input from the user (using the Scanner class).

a) Write a method called `shiftString` that shifts a certain number of characters from the end of a String to the front. Your method should take a String and an integer as parameters. The String parameter is the String to be used, and the integer is the number of characters to shift from the end to the beginning of the String. Your method should return a String that corresponds to the parameter String after the appropriate number of characters on the end have been shifted to the beginning, in the same order they originally appeared.

   For example, the return value after calling this method with parameters "Java" and 2 should be "vaJa". The return value after calling this method with parameters "Winston" and 4 should be "stonWin".

b) Write a method called `allShifts` that calculates all of the possible back-to-front character shifts for a given String. Your method should take a single String as a parameter and return a String that represents all of the possible shifts of characters (from the end to the front), starting with the original String, and with each shift being represented on a separate line. Your method should not print any output of its own.

   For instance, the return value after calling this method with parameter "Java" should be:

```
Java
aJav
vaJa
avaJ
```

and after calling with parameter “Nepal” should be:

```
Nepal
lNepa
alNepa
palNe
epalN
```

c) Write a method called `isPalindrome` that determines whether a String value is the same backwards as forwards. Your method should take one String parameter that is the String value to be checked. Your method should return true only if the String parameter consists of the same sequence of characters back to front as it does front to back (ignoring case, in both instances). Otherwise, it should return false.

For example, the return value after calling this method with parameter “Brightspace” should be `false`, and the return value after calling this method with parameter “avid DiVA” should be `true`.

d) Write a method called `checkForPalindrome` that determines whether a String given as input by the user is a palindrome. Your method should not take any parameters, but should use the `Scanner` class to take input from the user when the method is called. Give the user a meaningful prompt for their input. Your method should return true if the user-inputted String was a palindrome (ignoring case), and false otherwise.

For example, the return value after the user inputs “hello” should be `false`, and the return value after the user inputs “Anna” should be `true`.

e) Write a method called `flip` that reverses the order of the characters in a String. Your method should take a single String as a parameter and return a String that corresponds to the parameter String when written in reverse order.

For example, the return value after calling this method with parameter “Winston” should be “notsniW”.

f) Write a method called `mirror` that creates a palindrome from a String given as input by the user. Your method should not take any parameters, but should use the `Scanner` class to take input from the user when the method is called. Give the user a meaningful prompt for their input. Your method should process the String given as input, as return a String that is mirrored around the last letter of the input.

For example, the return value after the user inputs “hello” should be “hellolleh”, and the return value after the user inputs “Java” should be “JavavaJ”.

g) Write a `main` method that calls the methods from parts (b), (d), and (f) at least once each and prints out a meaningful message for each of their outputs. As an example of a “meaningful” message, your output could look like this after calling the method from part (d):

```
The input ‘Ana’ is a palindrome: true
```

Verify that your methods behave as you expect; however, you do not need to submit any output for your program.

Question 2.
Include all of the following work in a single class named `Question2`. Use appropriate comments for all custom methods. It is your responsibility to ensure your code compiles before you submit!

Write a simple calculator program that takes input from the user to perform addition, subtraction, multiplication, and division.

a) Write a method called `prompt` that will print to output a prompt for the user, asking whether they want to perform (a)ddition, (s)ubtraction, (m)ultiplication, or (d)ivision, or else (q)uit the calculator. Your method should not take any parameters, but should use the `Scanner` class to take input from the user after giving the appropriate prompt. Your method should return a character that corresponds to the first character of the String given by the user as input. If it is a letter, the returned character should be lower case.

b) Write a method called `intCalculator` that will perform integer arithmetic on a set of values given by the user. Your method should take a single character as a parameter that should indicate whether addition, subtraction, multiplication, or division will be performed ('a', 's', 'm', or 'd', respectively). Your method should also print out a prompt indicating that the user should enter two integer values. Your method should not return any value, but should print out the result of performing the indicated arithmetic operation on the two values given by the user.

c) Write a `main` method that uses the methods from parts (a) and (b) together to perform as many integer arithmetic operations as the user desires, until they enter 'q' to quit. If the user enters something that does not correspond to one of the five characters indicated in the prompt ('a', 's', 'm', 'd', or 'q'), then the program should print another prompt, and continue printing prompts until valid input is given. For example, a single run of the program should result in something similar to the following set of output prompts and input values from the user:

```
Do want to (a)dd, (s)ubtract, (m)ultiply, (d)ivide, or (q)uit?
add
Enter two integer values to add:
2
5
Your result is: 7
Do want to (a)dd, (s)ubtract, (m)ultiply, (d)ivide, or (q)uit?
d
Enter two integer values to divide:
13 2
Your result is: 6
Do want to (a)dd, (s)ubtract, (m)ultiply, (d)ivide, or (q)uit?
wuit
Do want to (a)dd, (s)ubtract, (m)ultiply, (d)ivide, or (q)uit?
q
```

Note that your own prompts do *not* have to be exactly the same, but *should* provide similar levels of information.