CSE 2102: Introduction to Software Engineering

Conditionals, Branching and Loops

Assigned: September 21, 2022, Due: October 5, 2022

Problem A (25 pts.)

Upgrade the program `ChocolateBars.java` from Homework #2 with the following two features:

1. Ask the user to input string "M" if the user is a man, and "W" if the user is a woman. Use only the male formula to calculate the calories if "M" is entered and use only the female formula to calculate calories if "W" is entered. If the user enters any input other than "M" or "W", an error message should be printed and the user should be prompted again.
2. Ask the user if they are Sedentary ("S"), Somewhat active, exercise occasionally ("O"), Active, Exercise 3-4 times a week ("A"), and Highly Active ("H"). The user will provide four inputs "S", "O", "A", and "H". If the user enters "S", increase the calculated BMR by 20%. If the user enters somewhat active, increase the calculated BMR by 30%. If the user enters "A", increase the calculated BMR by 40%, and if the user enters "H", increase the calculated BMR by 50%. The user is expected to provide only one of the four inputs. If the user enters any other input, an error message should be printed and the user should be prompted again.

Combine the inputs from (1) and (2) to calculate the adjusted BMR, and using this BMR calculate the number of chocolate bars. The program prints the adjusted BMR and the number of chocolate bars. The following sample output shows interactions with the user.

Input cmd:

```
java ChocolateBars
```

Output:

```
Enter the weight in pounds: 115
Enter the height in inches: 59
Enter the age in years: 50
Enter M for man, W for woman: W
Enter S for Sendtary, O for Ocassionally, A for Active, H for Highly Active: A
BMR adjusted for gender and lifestyle: 1668.52
Number of chocolate bars: 7
```

Submit at least 2 test cases you have used to test the code.

Problem B (25 pts.)

Write a program `ExamAverager.java` that computes the average of a list of non-negative scores. The program asks the user to enter all the scores followed by a negative sentinel value to mark the end of the data. Once the scores of the first exam are averaged, the program prints the average and each score, and whether it is below average, average, or above average. It then asks

the user if there is another exam to be averaged. The user can enter “Y” or “y” if there is another exam or “N” or “n” to exit. The user can repeat the entire process for another exam, and another, until the user wishes to exit the program. Assume that the input scores as well as the indicators to continue/exit are always correct (that is no error checking is needed). The following represents an example interaction:

Input cmd:

```
java ExamAverager
```

Output:

```
This program computes the average of a list of (nonnegative) exam scores.
Enter all the scores to be averaged.
Enter a negative number after you have entered all the scores.
15
20
15
-1
The average is 16.666666666666668
Score #1: 15.000000 -- Below average
Score #2: 20.000000 -- Above average
Score #3: 15.000000 -- Below average

Want to average another exam?
Enter yes or no: y
Enter all the scores to be averaged.
Enter a negative number after you have entered all the scores.
10
10
20
20
-1
The average is 15.0
Score #1: 10.000000 -- Below average
Score #2: 10.000000 -- Below average
Score #3: 20.000000 -- Above average
Score #4: 20.000000 -- Above average

Want to average another exam?
Enter yes or no: n
```

Show at least two interactions similar to the above that you used to test code.

Problem C (25 pts.)

Upgrade the program `ManipulateDNA.java` from Homework #2 with the following features.

1. The program should prompt the user for the name of the input file which has the DNA sequences stored. The DNA sequences are stored one per line. The program should read the input DNA sequences from this file, until the end of the file is reached.
2. For each input sequence, the program should compute and print the reverse of the input sequence, complement of the input sequence, and reverse complement of the input

sequence. To compute the complement, the following rules apply: complement of “A” is “T”, complement of “T” is “A”, complement of “C” is “G” and complement of “G” is “C”.

Input cmd:

```
java ManipulateDNA
```

Output:

```
Please enter the filename: in.txt

Sequence Number: 1
Input Sequence: ACGT
Reverse sequence: TGCA
Complement: TGCA
Reverse Complement: ACGT

Sequence Number: 2
Input Sequence: ATGCAGCT
Reverse sequence: TCGACGTA
Complement: TACGTCGA
Reverse Complement: AGCTGCAT
```

Show at least two test cases similar to above that you used to test code.

Problem D (25 pts.)

Write a Java program `TriangleTester.java` that prompts the user to input lengths of three line segments. Using the triangle inequality theorem, first determine whether these three line segments form a triangle. Once it is determined that the lengths form a triangle, further decide whether it is an acute, right or obtuse triangle using the converse Pythagoras theorem.

Input cmd:

```
javac TriangleTester.java
```

Output:

Use Case #1:

```
Enter the first length: 3
Enter the second length: 4
Enter the third length: 5
These lengths can form a triangle
It is a right triangle
```

Use Case #2:

```
Enter the first length: 3
Enter the second length: 4
Enter the third length: 7
These lengths cannot form a triangle
```

Show at least 4 test cases, one for each type of a triangle, and another for the case where the line segments cannot form a triangle.

Submission

The following deliverables must be submitted on HuskyCT by midnight on October 5, 2022.

a) Well-documented code.
b) Instructions to design test cases are specific to each problem.
c) Please make sure that your code compiles, we will test your code offline with specific test cases (common to all).
d) Late submissions (without any legitimate excuse) will incur a penalty of 10% per day.
e) In this homework, you have an opportunity to earn extra credit. On September 28, you can drop into any lab, and give an update to the TA on how far you have made progress on the homework. The purpose of this extra credit opportunity is to encourage you to make continuous progress on the project rather than waiting until the last minute (i.e., discourage procrastination). The maximum extra credit you can earn will be 1%, and this credit will be used to offset only the points that you may lose on this homework. Each member of the group must provide an update in person (unless there is a medical issue, in which case updates over WebEx will be accepted) to receive extra credit. Both the members of the group cannot receive extra credit if only one member shows up to the lab.