Lab Partner 1 Name

Lab Partner 2 Name

CS 115 Fall 2019 Lab #5

Due: **Monday, October 7th, midnight**

Points: **20**

Instructions:

1. Use this document template to report your answers. Enter all lab partner names at the top of first page.

2. You don't need to finish your lab work during the corresponding lab session.

3. ZIP your lab report and java files (if any) into a single ZIP file. Name the ZIP file as follows:

```
LastName_FirstName_CS115_Lab5_Report.zip
```

4. Submit the final document to Blackboard Assignments section before the due date. No late submissions will be accepted.

5. ALL lab partners need to submit a report, even if it is the same document.

Objectives:

1. (2 points) Implement selection condition statements.

2. (10 points) Write programs that use selection.

3. (8 points) Write and test user-defined class.

Problem 1 [2 points]:

For these problems you do not need to write entire Java program, just the condition statements requested. You can assume that all referenced variables have already been DECLARED and INITIALIZED.

A. **[0.5 points]** Write statements containing a logical expression that assigns `true` to the boolean variable `isCandidate` if `satScore` is greater than or equal to 1100, `gpa` is not less than 2.5, and `age` is greater than 15. Otherwise, `isCandidate` should be `false`.

Your answer:

```
if (satScore >= 1100 && gpa >= 2.5 && age > 15) {
   isCandidate=true;
} else {
   isCandidate=false;
}
```

B. **[0.5 points]** Write a selection statement that will print "hot" if the `temperature` >= 80, "pleasant" if the `temperature` >= 60, "cool" if the `temperature` is >= 45, "cold" otherwise.

Your answer:

```
if (temperature >= 80){
   System.out.println("hot");
} else if (temperature>=60) {
   System.out.println("pleasant");
```

```
} else if (temperature>=45) {
   System.out.println("cool");
} else {
   System.out.println("cold");
}
```

C. **[0.5 points]** If integer variable `currentNumber` is odd, change its value so that it is now 3 times `currentNumber`, then add 1, otherwise change the sign of `currentNumber`.

Your answer:

```
if ((currentNumber % 2) == 1) {
   currentNumber = 3 * currentNumber + 1;
} else {
   currentNumber = -1 * currentNumber;
}
```

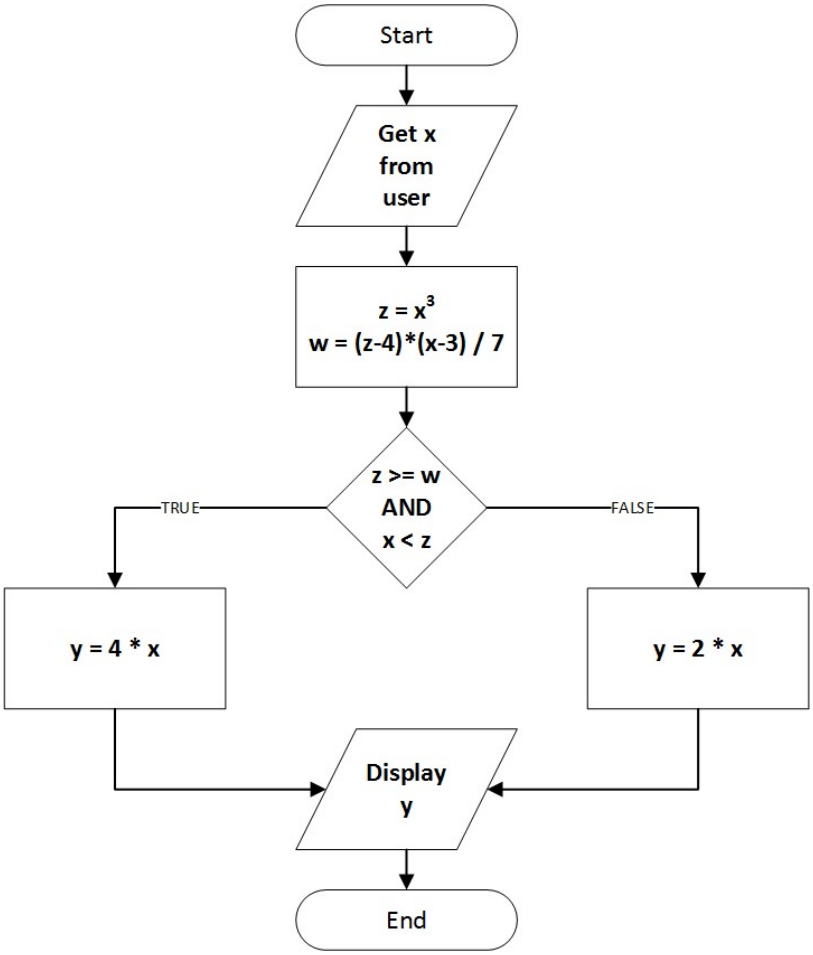
D. **[0.5 points]** Find the largest of three numbers `x`, `y`, and `z` and assign its value to the variable `largest`.

Your answer:

```
largest = x;
if (y > largest){
  largest = y;
}
if (z > largest) {
  largest = z;
}
```

Problem 2 [5 points]:

Implement the following program flowchart in Java.



You can use the skeleton code provided below.

Your Java program:

```
import java.util.Scanner;

public class Lab5Problem2 {

   public static void main(String args[]){
      // Declare variables
      double x, y, z, w;
```

```
      // Get the input from user
      Scanner scan = new Scanner( System.in );
      System.out.print( "Give me x: " );
      x = scan.nextDouble();

      // Statements that go before selection block
      z = Math.pow(x, 3.0);
      w = (z-4.0)*(x-3.0)/7.0;

      // Selection block:
      if (z >= w && x < z) {
         // "true" block
         y = 4.0 * x;
      } else {
         // "false" block
         y = 2.0 * x;
      }

      // Display the value of y
      System.out.println("The value of y is: " + y);
   }
}
```

Here are some sample program outcomes.

Sample behavior:

```
Give me x: 2.0
```

```
The value of y is: 8.0


Give me x: 0.2
The value of y is: 0.4


Give me x: -3.0
The value of y is: -6.0
```

Problem 3 [5 points]:

Recall the following problem from the beginning of the course:

Given the 3 dimensions of a box (length, width, and height), multiply them together to determine the volume.

Your task is to write a program (similar to that from Problem 4) that:

- Ask the user for input: length, width, and height,
- Check if all three inputs are valid and can be used:
  - If all three are valid, display a message on screen saying: "The volume is: 3.56" (assuming this is the value that was calculated),
  - If they're not, set volume to zero and display a message on screen: "ERROR: Cannot calculate volume."

You can use the skeleton code provided below.

Your Java program:

```
import java.util.Scanner;
```

```
public class Lab5Problem3 {

    public static void main(String args[]){
        // Declare variables
        double length, width, height, volume;

        // Get the input from user
        Scanner scan = new Scanner( System.in );
        System.out.print( "Length?: " );
        length = scan.nextDouble();
        System.out.print( "Width?: " );
        width = scan.nextDouble();
        System.out.print( "Width?: " );
        height = scan.nextDouble();

        // Selection block:
        if (length > 0.0 && width > 0.0 && height > 0.0 ) {
            // "true" block
            volume = length * width * height;
            System.out.println("The volume is: " + volume);
        } else {
            // "false" block
            volume = 0.0;
            System.out.println("Error: Cannot calculate volume");
        }
    }
}
```

Problem 4 [8 points]:

Write a user-defined Java class called `BoxVolumeCalculator` that could be used to calculate the volume of a box given length, height, and width (similar to your program from Problem 3).

Your class should:

- NOT have a `main` method,
- Should have four fields (use correct data types):
  - `length`,
  - `width`,
  - `height`, and
  - `volume`,
- A default constructor method that resets all four fields to zero,
- Three mutator / setter methods:
  - `setLength( )` that will set the `length` field to a new value passed as argument (regardless of whether it is valid or not),
  - `setWidth( )` that will set the `width` field to a new value passed as argument (regardless of whether it is valid or not),
  - `setHeight( )` that will set the `height` field to a new value passed as argument (regardless of whether it is valid or not),
  - Use public access modifiers for all four mutator / setter methods,
- Four accessor / getter methods:
  - `getLength()` that will return the value of the `length` field,
  - `getWidth()` that will return the value of the `width` field,
  - `getHeight()` that will return the value of the `height` field,
  - `getVolume()` that will return the value of the `volume` field,
  - Use public access modifiers for all four accessor/ getter methods,
- a `calculateVolume()` method which:
  - If all three dimension fields have valid values will calculate box volume based on current values of `length`, `height`, and `width` fields and update the `volume` field accordingly.
  - Otherwise it will set the `volume` field to zero

`BoxVolumeCalculator` class code:

```
class BoxVolumeCalculator {
    // Class fields (don't need to be private this time)
    private double length, width, height, volume;

    // Default constructor (no arguments)
    BoxVolumeCalculator() {
        // Reset fields to zero
        this.length = 0.0;
        this.width = 0.0;
        this.height = 0.0;
        this.volume = 0.0;
    }

    // Length field mutator / setter method
    public void setLength(double newLength){
        // Change the field value to a new one
        this.length = newLength;
    }

    // Width field mutator / setter method
    public void setWidth(double newWidth){
        // Change the field value to a new one
        this.width = newWidth;
    }

    // Height field mutator / setter method
    public void setHeight(double newHeight){
        // Change the field value to a new one
        this.height = newHeight;
    }

    // Length field accessor / getter method
    public double getLength(){
        // Return the field value
        return this.length;
    }
```

```
    // Width field accessor / getter method
    public double getWidth(){
        // Return the field value
        return this.width;
    }

    // Height field accessor / setter method
    public double getHeight(){
        // Return the field value
        return this.height;
    }

    // Volume field accessor / setter method
    public double getVolume(){
        // Return the field value
        return this.volume;
    }

    // calculateVolume method
    public void calculateVolume(){
        // Check if dimension fields have correct values...
        if (this.length > 0.0 && this.width > 0.0 && this.height > 0){
            // ...if yes, calculate volume...
            this.volume = this.length * this.width * this.height;
        } else {
            // ...if not, set it to zero
            this.volume = 0;
        }
    }
}
```

If you implemented your `BoxVolumeCalculator` class correctly and compile/run the following program:

Lab5Problem4Test **program:**

```
public class Lab5Problem4Test {


    public static void main(String args[]){

        // Instantiate the BoxVolumeCalculator class object (calling
constructor)

        BoxVolumeCalculator myCalculator = new BoxVolumeCalculator();


        // Show field values before using setter methods to change them

        System.out.println("Initial dimensions:");

        System.out.println("Length: " + myCalculator.getLength() + " |
Width:   "   +   myCalculator.getWidth()   +   "   |   Height:   "   +
myCalculator.getHeight());


        // Set box dimensions using mutator / setter methods

        myCalculator.setLength(2.0);

        myCalculator.setWidth(3.0);

        myCalculator.setHeight(4.0);


        // Show field values before using setter methods to change them

        System.out.println("New dimensions:");

        System.out.println("Length: " + myCalculator.getLength() + " |
Width:   "   +   myCalculator.getWidth()   +   "   |   Height:   "   +
myCalculator.getHeight());


        // Re-calculate box volume

        myCalculator.calculateVolume();


        // Show the current box volume

        System.out.println("Calculated       volume:       "       +
myCalculator.getVolume());


        // Let's try again with invalid dimensions

        System.out.println("Let's try again:");
```

```
        // Reset box dimensions using mutator / setter methods
        myCalculator.setLength(2.0);
        myCalculator.setWidth(3.0);
        myCalculator.setHeight(-4.0);

        // Show field values before using setter methods to change them
        System.out.println("New, reset, dimensions:");
        System.out.println("Length: " + myCalculator.getLength() + " | Width:  "  +  myCalculator.getWidth()  +  "  |  Height:  "  +  myCalculator.getHeight());

        // Re-calculate box volume
        myCalculator.calculateVolume();

        // Show the current box volume
        System.out.println("Calculated       volume:       "       +  myCalculator.getVolume());



    }
}
```

You should see the following output:

```
Initial dimensions:
Length: 0.0 | Width: 0.0 | Height: 0.0
New dimensions:
Length: 2.0 | Width: 3.0 | Height: 4.0
Calculated volume: 24.0
Let's try again:
New, reset, dimensions:
Length: 2.0 | Width: 3.0 | Height: -4.0
Calculated volume: 0.0
```

In the program above, note how the calculator object is instantiated and how its methods are being used to achieve the goal.

Our `BoxVolumeCalculator` class design is not the most practical. Can you identify any flaws? What could go wrong and what would you change (you don't need to modify code, just explain it):

Your answer:
No need to grade these, but provide feedback if possible: Mutator methods should test validity of inputs Mutator methods could recalculate volume anytime a new value is set etc.