Midterm Laboratory Activity 3

Score:

Name: _____
Subject Code & Schedule: _____
Course and Year: _____

TITLE: Loops

LEARNING OBJECTIVES:

At the end of this activity, the students should be able to:

1. Identify available loop control structures in Java.
2. Differentiate the different loop control structures in Java.
3. Implement these different loop control structures for a given problem.
4. Create a complete Java program that simulates these basic operations.

INSTRUCTIONS:

1. Make sure you have your own individual account.
2. Always keep your account secret to others to avoid unauthorized access to your files.
3. Always save your work and log-off when not using the computer.
4. By now you should have been familiarized using your text editor.
5. By now you should know how to create, save, compile, execute, and debug programs in Java.
6. Use the skills and learning obtained in Prelim Activity 1 to Midterm Activity 2 in order for you to successfully finish the learning objectives of this module.

DURATION: Two to Three Meetings

HANDS-ON:

1. Log-on using your own individual account. Use your own **username** and **password.**
2. Open your text editor.
3. Write your next Java program:
   3.1. Write your next program by copying the source code shown below to your text editor.

```
/* Programmed by: <write your name here>
   Program title: Sum1.java
   Program Date: <write the date today here> */

import java.io.*;
public class Sum1{
  public static void main(String[] args){
      int start = 0, end = 0, sum;
      String input = " ";

      BufferedReader in = new BufferedReader(new
                            InputStreamReader(System.in));

      try{
        System.out.print("Input starting number: ");
        input = in.readLine();
        start = Integer.parseInt(input);

        System.out.print("Input ending number: ");
        input = in.readLine();
        end = Integer.parseInt(input);
      }catch(IOException e){
        System.out.println("Error!");
      }

      if(start >= end){
        System.out.print("Starting number should be lesser ");
        System.out.println("than the ending number. ");
        System.out.println("Try again.");
        System.exit(0);
      }

      if(start%2 == 0)
      {
            start = start + 1;
      }

      sum = 0;
      while(start <= end)
      {
            sum = sum + start;
            start = start + 2;
      }
      System.out.println("Sum = " + sum);
  }
}
```

3.2. Save the program as **Sum1.java** then compile your program until no errors and warnings are reported.
3.3. Run your program.
3.4. Simulate and write what will be displayed on the screen.

4. Now let's try another implementation. Copy the source code below and save it as **Sum2.java**

```
/* Programmed by: <write your name here>
   Program title: Sum2.java
   Program Date: <write the date today here> */

import java.io.*;
public class Sum2{
  public static void main(String[] args){
      int start = 0, end = 0, sum;
      String input = " ";

      BufferedReader in = new BufferedReader(new
                           InputStreamReader(System.in));
      try{
        System.out.print("Input starting number: ");
        input = in.readLine();
        start = Integer.parseInt(input);

        System.out.print("Input ending number: ");
        input = in.readLine();
        end = Integer.parseInt(input);
      }catch(IOException e){
        System.out.println("Error!");
      }

      if(start >= end){
        System.out.print("Starting number should be lesser ");
        System.out.println("than the ending number. ");
        System.out.println("Try again.");
        System.exit(0);
      }
      if(start%2 == 0){
            start = start + 1;
      }
      sum = 0;
      for(start = start; start <= end; start = start + 2){
            sum = sum + start;
      }
      System.out.println("Sum = " + sum);
  }
}
```

4.1. Simulate and write what will be displayed on the screen. Try using the same input values in your **Sum1.java**

5. Let us try another implementation. Copy the source code and save it as **Sum3.C**

```
/* Programmed by: <write your name here>
   Program title: Sum3.java
   Program Date: <write the date today here> */

import java.io.*;
public class Sum3{
  public static void main(String[] args){
      int start = 0, end = 0, sum;
      String input = " ";

      BufferedReader in = new BufferedReader(new
                            InputStreamReader(System.in));
      try{
        System.out.print("Input starting number: ");
        input = in.readLine();
        start = Integer.parseInt(input);
        System.out.print("Input ending number: ");
        input = in.readLine();
        end = Integer.parseInt(input);
      }catch(IOException e){
        System.out.println("Error!");
      }
      if(start >= end){
        System.out.print("Starting number should be lesser ");
        System.out.println("than the ending number. ");
        System.out.println("Try again.");
        System.exit(0);
      }
      if(start%2 == 0){
            start = start + 1;
      }
      sum = 0;
      do{
            sum = sum + start;
            start = start + 2;
      }while(start <= end);

      System.out.println("Sum = " + sum);
  }
}
```

5.1. Simulate and write what will be displayed on the screen. Try using the same input values in your **Sum1.java** and **Sum2.java**

6. After your simulation for the three programs, what do you think is the main objective of these programs?

7. Identify what are the different loop control structures in Java.

8. What are the usual similar components of these loop-constructs?

9. What do you think is different among these identified loop constructs? Justify your answer.

10. Create a complete Java program that shall allow the user to accept three integer values representing the START, END, and STEP respectively. The START should always be lesser than the END value, and the STEP is always greater than zero. The program shall print vertically the values starting from the START to the last value which can be equal to or lesser than the END incremented by the STEP value.

    Example Output 1: if START = 1, END = 10, STEP = 2

```
Input START value   = 1
Input END value     = 10
Input STEP value    = 2

1
3
5
7
9

Do you want to try again (Y/N)? Y
```

    Example Output 2: if START = -10, END = 20, STEP = 5

```
Input START value   = -10
Input END value     = 20
Input STEP value    = 5

-10
-5
0
5
10
15
20

Do you want to try again (Y/N)? N
```

    10.1. Your program should automatically detect any errors in the initial input.
    10.2. Your program should have an additional feature that asks the user whether the user wants TO TRY AGAIN. The Program will not terminate until the user inputs any character other than ‘Y’ or ‘y’.
    10.3. Implement the program using all looping constructs identified earlier.

10.4. Write your complete programs in the space provided for.

```
// Using while Loop
```

```
// Using for Loop
```

```
// Using do-while Loop
```