CSE20 : Lab #8 - Dynamic Control Flow

Overview

We will first work on some misconceptions when using if-statements. There are three types of if-statements that should be used depending on the situation. Refer to lab6 for the description on the differences between the three structures. We will also introduce new operators that are shorthand for accumulating values into one variable.

Code Segments

We need to do three things in this sequence of code. First, read the number from the user. Add it to a running total that keeps track of the total of all the numbers entered. Then we have a running count that we decrease each time we need a number successfully from the input. The basic way of writing these three things is in three distinct java statements.

```
int num = input.nextInt();
total = total + num;
runningCount = runningCount - 1;
```

In some situations we do not need to keep track of the number read as the only thing we are interested in is the total. In this case, we do not need to store the number in a temporary variable. We can directly add it to the total using the following segment:

```
total = total + input.nextInt();
runningCount = runningCount - 1;
```

This type of operation is called accumulating. We can shorthand this type of logic using "+=". It will take same variable with the current value and store the new value after adding something to it. In other words, read and write operations are using the same variable.

```
total += input.nextInt();
runningCount = runningCount - 1;
```

Frequently in code segments, we will need to change the value of the variable by adding or subtracting 1. Shorthand for these operations are "++" and "--" to the counting variable. There are two ways of using these operators called prefix and postfix.

```
total += input.nextInt();
runningCount--;
```

Above usage is postfix operation and the below is the prefix operation. When they are used in isolation then there is no difference.

```
total += input.nextInt();
```

```
--runningCount;
```

When the operators are used in larger expressions or the "current" value has to be calculated then the differences come out. Prefix means it adds one before the value is checked. Postfix means the value is checked and then it adds one. The following code segment shows the effect of the unary operator applied prefix or postfix.

```
int i = 3;
i++;
System.out.println(i);          // "4"
++i;
System.out.println(i);          // "5"
System.out.println(++i);        // "6"
System.out.println(i++);        // "6"
System.out.println(i);          // "7"
```

Getting started

After starting Eclipse, create a new project called Lab 8. Import from the assignment page Error7.java and FiveAverage.java into the project and load them. FiveAverage makes use of the new operators we discussed and introduced in this lab. It dynamically figures out how many inputs to ask from the user and calculate the average

(Exercise) Fix – Error7.java

The program should ask the user to enter numbers 0th, 1st, 2nd, 3rd and 4th for a total of 5 numbers. Then it will calculate the average of all the five numbers regardless of the value. It will check each number and only total up the positive numbers. Finally it should print out the average of only the positive input numbers.

(Assessment) Logic Check

1. Enter the following values as input for each of the five numbers. Check the output to make sure your Error7 is calculating correctly. For each of the input, show the output of the program like the following:

```
*** This program will find the average of 5 numbers ***
Please enter 0 number:1
Please enter 1 number:2
Please enter 2 number:3
Please enter 3 number:4
Please enter 4 number:5
The average of 5 numbers is 3
Average of the positive numbers is 3
```

- 1, 2, 3, 4, 5
- 5, 10, 15, 20, -25
- 5, -5, 0, 0, 0
- 5, -5, 10, -10, 0
- -1, -2, -3, -4, -5

2. What is the output of the following code sequence? (code included at the end of FiveAverage.java)

```
int i = 10;
i--;
System.out.println(i);
--i;
System.out.println(i);
System.out.println(++i);
System.out.println(i++);
System.out.println(i);
System.out.println(--i);
System.out.println(i--);
System.out.println(i);
if (i++ == 8)
      System.out.println("Eight");
if (++i == 9)
      System.out.println("Nine");

System.out.println("Final value " + i);
```

3. Provide the code change so it also prints out Nine.

(Exercise) TenAverage.java

Program asks the user to input number 0-10.

- Ask the # to average from 0 to whatever number chosen
- Copy and extend the structure in FiveAverage
- Print out the average of the numbers entered

What to hand in

When you are done with this lab assignment, you are ready to submit your work. Make sure you have done the following ***before*** you press Submit:

- Include answers to Assessment questions
- Attach edited Error7.java and TenAverage.java
- List of Collaborators