CSE20 : Lab #3 - Input/Output

By now everyone is sick of just doing prints and having the program behave exactly the same way each time you run it. There must be more to programming than just hard-coding everything for a static behavior. Luckily for us there is, this week we will be learning to use *Input* methods, ways of asking the user of the program to enter more or new information which we can then utilize in our program.

Overview

In order to make *Input* most useful, we also need to store them for later use. That means whatever stuff we want to remember has to be understood by the computer so when we ask for it again, we get exactly what we put in. This leads to *variables* where different data types are created based on a common convention understood by the computer and us.

(Reading) zyBooks : Chapter 1.4 & 1.5

Read chapters 1.4 and answer Participation Activities 1.4.1 and 1.4.2 about Scanners. Also answer 1.5.2 after reading chapter 1.5 about Comments.

Use of Scanners

We will use a newer utility provided by Java called Scanner to give us ability to read input on the keyboard by the user. In order to do this we need two pieces of code to enable us access to Scanner class and subsequently the inputs.

```
import java.util.Scanner;                  // at top of the file
Scanner input = new Scanner(System.in);
```

This allows using *input* to gather data from the keyboard with different calls for each data type we are going to need in this lab:

```
input.nextInt();          // get the next integer number
input.nextFloat();        // get the next floating point number
input.nextLine();         // get the next string line
```

Getting started

After starting Eclipse, create a new project called Lab 3. You should save the program Averages.java into this project and load it in Eclipse. You will see the following salient features in this new program.

- the *import* line at the top of the file
- *input* object created inside *main*.
- Declaration of *variables*: *n1* and *n2*
- pairs of println and nextInt calls (to inform user and gather data)
- save data into *n1* and *n2*
- use saved data to calculate new *variable* called *average*
- output the result to the user

(Assessment) Finding and Fixing Errors

We call the process of finding and fixing errors debugging. You will be introduced to it in this lab. In the assignment page you will find 3 files: Errors1.java, Errors2.java and Errors3.java. Obviously, they don't run as is or produce the correct output. Your job is to find the correct behavior based on what you learned from Averages.java. Your changes should follow the convention below:

```
/* I figured all this out and made it all work.

   These comments are optional */

// System.out.println("This is old erroneous statement");

System.out.println("This is corrected statement");
```

You'll turn the existing line of code inside the files into a comment (required). Then have the corrected line preceding it. You're free to add more comments as notes to yourself to explain the nature of the error(s) you fixed. This way if the TA asks you why you made the changes, it'll be easier to recall or subsequently when you review this lab. Your exams can have similar questions in the vein of these exercises so get comfortable with them. **Remember Java is case sensitive**.

(Reflection) Debugging Process

1. How many types of errors were there?
2. How do you know when you have found every error (bugs)?
3. Is naming convention important to avoid errors?

(Exercise) Interviewer Program

Create a new class called Interviewer. You will use the 5 questions or more from Lab 2 to create this. It will ask the user the questions one by one to gather information about them. Make sure at least one question requires an *integer* as an answer. You will also need to know how to create *String* objects. One example is given below:

```
String name;
System.out.print("What is your name? ");
name = input.next();

System.out.print("Their name is ");

System.out.println(name);
```

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 1.4.1, 1.4.2, and 1.5.2
- Answers to Reflection questions (1-3)
- Attach fixed up Errors1.java, Errors2.java and Errors3.java
- Attach your new Interviewer.java
- List of Collaborators