Java Refresher

Module 1 - Data, I/O and conditionals

After completing this module, you will have written basic programs that input, output and calculate data of various types, as well as make decisions based on the contents of variables.

Background Material

The Input class

To help you read input from the user, we provide you with a class called `Input`. This class is included in the Module 1 Skeleton Code and provides the following methods:

`String askString(String question)`	Asks the user to input a String and returns that String.
`int askInt(String question)`	Asks the user to input an integer and returns that integer.
`double askDouble(String question)`	Asks the user to input a double-sized floating point real number and returns that number.
`char askChar(String question)`	Asks the user to input a single character and returns that character.
`boolean askBoolean(String question)`	Asks the user to input a boolean (true or false) and returns that boolean.

Each method takes a question as a parameter, will print the question to the terminal, will wait for the user to type an answer, and will then return the user's answer. When using one of these methods, you should typically save the answer into a variable. Depending on the type of data that is returned, you will need to declare your variable of the appropriate type.

Reference example

Here is a simple example using the Input class:

```
int pickedNumber = Input.askInteger("Pick a number between 1-10:")
```

Because we asked for an integer, the variable into which you save it (in this case `pickedNumber`)must also be declared with the type of integers (i.e. "`int`").

Once you have the answer saved in a variable, you can refer to it by name. For example:

```
if (pickedNumber == 7)
{
  System.out.println("You guessed correctly!");
}
else
{
  System.out.println("Sorry, " + pickedNumber + " is incorrect.")
}
```

Testing for String equality

Note that the == equality operator works for integers and other primitive types, but does not work as expected for strings.

In Java, strings are objects, not primitive values. A peculiar consequence of this is that it is possible to have two distinct String objects which happen to contain the same sequence of characters, and yet which are still distinct objects. If this puzzles you, consider having two Person objects which happen to contain the same name and age. Just because two people have the exact same name and age, does that mean that those two objects are the same object? Definitely not.

In short, when applied to objects, the == equality operator will compare whether two the two sides refer to the same object. But if you are comparing two string objects, you probably don't care as much about that as you do about whether the contents inside the two string objects are the same. To compare the contents of two strings, write: `if (string1.equals(string2))`. For example:

```
String day = Input.askString("What day of the week is it?");

if (day.equals("Saturday") || day.equals("Sunday")) {
    System.out.println("It's the weekend!");
}
```

Submitting to PLATE

The following exercises are to be completed and submitted to the http://plate.it.uts.edu.au/ website. When you submit to PLATE, you will see a report indicating what is correct and incorrect about your solution. If your solution is incorrect, using this feedback to fix the problems with your solution and resubmit as many times as required until you PASS all tests.

Exercises

Each exercise below will involve writing a short program with the following structure:

```
public class ExerciseN
{
  public static void main(String[] args)
  {
    YOUR LINES OF CODE GO HERE
  }
}
```

In the skeleton code for this module, you will find 9 ready-made skeleton classes like this, named Exercise1, Exercise2, ... up to Exercise9. Your solution to each exercise should be written inside the corresponding ready-made class.

Exercise 1.

Learning objective: Familiarise yourself with different types and type errors.

Class to edit: `Exercise1`

Inside the main method of class `Exercise1`, you will find the following 5 variable declarations:

```
// Declare 5 variables (DO NOT CHANGE THESE DECLARATIONS)
int i;
boolean b;
char c;
String s;
double d;
```

These are different storage locations in the computer's RAM where you are able to store different types of data. Your task is to store the following 5 data values into these variables. Think carefully about which data value belongs in which variable:

1. true
2. 2.5
3. "hello"
4. 'y'
5. 3

Before you write your solution, locate the following code inside class `Exercise1`:

```
// You must edit the following 5 lines
i = true;
b = 2.5;
c = "hello";
s = 'y';
d = 3;
```

These 5 lines are a first (although incorrect!) attempt at a solution. Try to compile `Exercise1` now, and you will see the following error:

```
incompatible types: boolean cannot be converted to int
  i = true;
      ^
```

Since `i` is an integer variable and `true` is a boolean value, the value `true` cannot be stored into the variable `i`. Of the 5 values we asked you to store, only one of them is an integer: `3`. Therefore, the correct code for this line would be:

```
  i = 3;
```

After you store each of the values into the correct variables and the compiler accepts your code with no errors, export your project to a jar file being sure to "Include the source files", and then submit this to PLATE to confirm whether your solution is correct or not. If there are no compile errors, PLATE will show you the 5 lines of output produced by this program. These 5 lines are actually generated by the following 5 lines at the bottom of class `Exercise1`:

```
// Print the 5 variables
System.out.println("i = " + i);
System.out.println("b = " + b);
System.out.println("c = " + c);
System.out.println("s = " + s);
System.out.println("d = " + d);
```

If your program is incorrect, you may have unlimited attempts to edit your code, re-export to a jar file and resubmit.

Passed? Excellent! As you progress through each exercise below, be sure to continually submit to PLATE to confirm whether your solution is correct. If your solution is not correct, let PLATE's feedback report inform you on what you need to fix.

Exercise 2.

Learning objective: Understand how to store data into a variable and reference it later.

Class to edit: `Exercise2`

Using the supplied input class, write a program that asks the user `"How old are you?"`, and then prints `"Your age is xxx"` where `xxx` should be substituted by whatever age the user typed. A sample I/O trace of this program follows:

```
How old are you?     « user inputs 18 »
Your age is 18
```

Your program should be two lines of code:

1. Use the appropriate method in the Input class to ask a question and obtain an **integer** answer. Ask the question `"How old are you?"` and then store this age into a variable of an appropriate name and an **appropriate type**. The name of the variable should describe what is stored in it (e.g. "`xxx`" is NOT a good name for this variable). The type of the variable should match what type of data is being stored into it.
2. Print the message `"Your age is xxx"` message terminated by a newline (i.e. use the `System.out.println` method). The message `"Your age is xxx"` should be constructed by adding the string `"Your age is "` (including the space) together with the user's age which you can find in your variable.

Exercise 3.

Learning objective: Reinforce your understanding of variables by using a different data type.

Class to edit: `Exercise3`

Write a similar program asking for the user's name, according to the following sample I/O:

```
What is your name?     « user inputs Kelly »
Hello Kelly
```

The steps are similar to the previous program, except that your program needs to read a different type of data, and store it into a different type of variable.

Exercise 4.

Learning objective: Make a decision based on the contents of a variable.

Class to edit: `Exercise4`

Write a program that asks the user `"How old are you?"` and stores the answer into an appropriately named and typed variable. If the user is at least 17 years old, print `"You are old enough to drive."` otherwise print the message `"You are not old enough to drive."`.

The following sample I/O shows what should happen when the user is at least 17 years old:

```
How old are you?     « user inputs 17 »
You are old enough to drive.
```

The following sample I/O shows what should happen when the user is less than 17 years old:

```
How old are you?     « user inputs 16 »
You are not old enough to drive.
```

Before submitting to PLATE, be sure to get all punctuation and spacing correct in the message. In this case, note that the message has just one space between each word, and a single fullstop at the end.

Hint: It may be helpful to refer to the example in the Background Material section which includes an if/else statement that makes a decision based on the contents of a variable.

Exercise 5.

Learning objective: Compare the contents of a string to another string.

Class to edit: `Exercise5`

Write a program that asks the user `"Are you a student?"`. If the user answers `"yes"` (in all lowercase), print `"You are eligible for a discount."` otherwise print `"You are not eligible for a discount."`

Exercise 6.

Learning objective: Make a decision based on 2 conditions both being true using the `&&` operator.

Class to edit: `Exercise6`

Write a user login program that asks the user to input a username and then a password. If the username equals `"joe"` and the password equals `"guess"`, your program should accept this as the correct login and print the message `"Welcome, joe!"`. Otherwise your program should print the message `"Incorrect username or password."`.

The following sample I/O shows how your program should behave when both the username and password are correct

```
Username:     « user inputs joe »
Password:     « user inputs guess »
Welcome, joe!
```

Otherwise, your program should behave as follows:

```
Username:     « user inputs joe »
Password:     « user inputs secret »
Incorrect username or password.
```

Exercise 7.

Learning objective: Calculate a simple result.

Class to edit: `Exercise7`

Write a program that calculates and prints the area of a square according to the following sample I/O:

```
Enter square side length:     « user inputs 2.5 »
The area of the square is 6.25
```

Notice that all numbers and calculations are performed using the `double` data type.

When calculating something, it is usually convenient to temporarily store the result into a local variable such as:

```
double area = « formula for the area of a square »;
```

Your solution should be 3 lines of code:

1. Ask for the square side length and store it into a variable.
2. Calculate the area and store it in another variable.
3. Print the area.

Hint: The formula for the area of a square is length^2. There are two ways to calculate the square of a number. You could multiply `length * length` or you could use the `Math.pow()` method to calculate `length` to the power of `2`.

Exercise 8.

Learning objective: Perform a calculation involving a pre-defined constant.

Class to edit: `Exercise8`

Write a program that calculates and prints the area of a circle according to the following sample I/O:

```
Enter circle radius: « user inputs 2.5 »
The area of the circle is 19.634954084936208
```

Hint: The formula for the area is π*radius^2. For π, use the pre-defined Java constant `Math.PI`.

Exercise 9. *(Challenge Question)*

Learning objective: Calculate a complex result.

Class to edit: `Exercise9`

Write a program that asks the user to input the (x,y) coordinates of 2 points, and then calculates and shows the distance between those two points. The program should execute according to the following sample I/O:

```
Enter x coordinate for point 1: « user inputs 2.0 »
Enter y coordinate for point 1: « user inputs 1.0 »
Enter x coordinate for point 2: « user inputs 5.0 »
Enter y coordinate for point 2: « user inputs 5.0 »
```

```
The distance between the two points is 5.0
```

Hint: Use variable names x1,y1 for the coordinates of point 1, and use variable names x2,y2 for the coordinates of point 2. The formula for the distance between two points is: $\sqrt{(x2-x1)^2 + (y2-y1)^2}$. You have already seen how to calculate a number to the power of 2. To calculate the square root of a number, you can use the `Math.sqrt` method.