CSC207 Lab 1 — IntelliJ, variables, and loops

Overview

To earn lab marks, you must arrive on time, actively participate for the entire session, and make good progress.

This week, we are going to help you understand the difference between declaring and redeclaring variables, applications of types, and how to set up IntelliJ IDEA. From now on, we will only refer to it as "IntelliJ".

Icebreaker

Your TA will introduce themselves, and then help you introduce yourself to your neighbours. First, learn a few things about them. You will be using these later on in the tutorial:

1. Three (3) neighbours' names
2. Those neighbours' favourite number
3. Those neighbours' favourite ice cream

After that, choose up to three of your neighbours to do the remainder of this lab with. Make sure everyone has read through each step before you attempt it!

Setting up IntelliJ

Linux is case sensitive. Use exactly the capitalization we use throughout the course.

To start IntelliJ, go to the command line and type "`idea`".

IntelliJ has four levels of organization:

1. When you first run IntelliJ, it will ask you in which folder you want to store your projects. Use the default `/IdeaProjects/`.
2. The next level of organization are *projects*. You will (usually) create a new project for every lab and assignment. Do so now: go to *File* → *New* → *Project*. You may have to tell IntelliJ where Java is located. Java 1.8 is available in the *Project SDK* drop-down list, select it. If it is not available, click *new JDK* and go to `/local/packages/jdk1.8.0`. Once the project SDK has been set, click *next*. Then click *next* again. Set the project name to `Lab1` and press *Finish*. It may take a few seconds for the project to appear in the menu on the left side of your window.
3. After projects, you have *packages*. These are folders which contain logically connected Java *classes*. Expand *Lab1* and right-click on the *src* directory. Then click *New* → *Package*. Packages in Java begin with lower case letters. We will call ours, `lab1`, starting with a lower case "L".
4. Finally, create a new *class* in the same way you created the *package*, by right clicking your package and clicking *New* → *Java Class*.
   This lab has you find sets of neighbours with similar attributes, so enter `NeighbourAnalyzer` into the *Name* field and press *Finish*.

Method `main`

As we discussed in class, Java will start by running the main method first. Type the following so that your code now looks like:

```
public class lab1 {
   public static void main(String[] args) {
   }
}
```

Variables in Java

As you have seen in lecture, in Java, types are strictly checked. In Python, you could run this:

```
myVariable = 5
print(myVariable / 2)
myVariable = 'some string'
# The next call on function print causes an error when you run your program.
print(myVariable / 2)
```

Java, on the other hand, forces you to say what kind of variable you are declaring, and then forces you to stay consistent while that variable is set. For example, the equivalent in Java wouldn't work:

```
// You must declare a variable in Java before you use it.
// Variable declaration has the form [type] [name];
// Assigning to a variable has the form [name] = [value];
// You can combine declaration and assignment: [type] [name] = [value];
int myVariable = 5;
System.out.println(myVariable / 2.0);
// Now we want to reassign to myVariable, so we use the [name] = [value] form:
myVariable = "some string";
// Because Java is "type safe," the line above will be labeled as the
// troublemaker. Java gives the red squiggly lines of doom, because it notices,
// Hey! myVariable is declared as an "int" in the declaration above! You can't
// set it to be a String!
// If you try this, you'll notice that you can't run your program because Java
// can't compile it.
System.out.println(myVariable / 2.0);
```

Loops

In Java, the most common loop is the *for loop*. Here is an example:

```
String[] arr = new String[] {"a", "b", "c"};
for (int i = 0; i < arr.length; i++) {
    System.out.println(arr[i]);
}
```

That code is equivalent to this *while loop*:

```
String[] arr = new String[] {"a", "b", "c"};
int i = 0;
while (i < arr.length) {
    System.out.println(arr[i]);
    i++; // Notice that the increment happens at the end of the loop body.
}
```

Your task

Create a main method in your class: `public static void main(String[] args)`

Within your `main` method, declare arrays for your names, your favourite numbers, and your favourite ice creams. You must decide on the types for these arrays. What type have you seen used for names? What type is more suited for numbers? If you have questions, ask your TA to look over your work.

Next, loop over pairs of people and compare each pair's names, favourite numbers, and favourite ice creams. Print which neighbours like the same things. (Feel free to change people's favourite numbers if you and your neighbours have nothing in common.)

For example, if Sophia and Colin both had Rocky Road as their favourite ice cream, your program should print this:

```
Sophia and Colin like Rocky Road ice cream
```

and then continue comparing peoples' favourite numbers.

To run your code, right click your `NeighbourAnalyzer` class in the left pane, and select *Run NeighbourAnalyzer....main()*.

Let your TA know when you are done so that they can check over your work.