Lab 2 : Variables

Topics covered variables, assignment

Exercise 1: To help us develop better course content for you, please take a few time to fill a mini suvery about your programming knowledge before this course. Simply click the following link to start the survey.
https://docs.google.com/forms/d/1Oqnk0HRQQ1QpCNvfi9agTBFIFG5PZs3AIR3yHYIc53s/viewform

Exercise 2: One of the first things you need to do when you decide to store some data in a program is to decide what kind of data you are using. If we don't tell our program the correct data type, then you can have all kinds of issue when working with that data.

Write an appropriate Java data type to store each of the following kinds of information:

The number of sledgehammers in Jane's garage ________________

If a light switch is on or off ________________

The time it takes for an Olympian to run 100m ________________

Number of rocks on Mars ________________

Percentage of students who ask questions in tutorials ________________

Student high school report grade; i.e. A, B, C, D, E, F ________________

Probability of winning the lotto ________________

Number of human fingers on a hand ________________

Number of sand grains in Simpson desert ________________

Blood type ________________

Names of the Australian floral emblems ________________

Number of cups of flour to make a cake ________________

States of the lunar cycle ________________

Colour ________________

Exercise 3: In Java there are both rules and conventions for what you can name your variables. Breaking one of the rules will cause your code to have a syntax error, and hence not compile, while breaking a convention is just bad practice.

Identify which of the following variable names break a rule or convention, and write why in the space provided. If you need help, have a look at Oracle's tutorial on variables and see the official rules.

`5nums` ________________________________

`input-length` ________________________________

`theCount` ________________________________

`UserName` ________________________________

`_needsDeleting` ________________________________

`first3Letters` ________________________________

`mycoolvariable` ________________________________

Exercise 4: What are the values of the variables below, at the places indicated? As you go through the algorithm, draw a diagram to demonstrate how these variables would be represented in the computer memory. Note: “←” means “gets the value of”.

Algorithm 1: VARIABLES

```
let $x$ be an integer
let $y$ be a floating-point number
$x \leftarrow 2$                       // What does $x$ equal?
$x \leftarrow x + 3$                   // What does $x$ equal?
$y \leftarrow 1/x - 1$                 // What does $y$ equal?
$x \leftarrow 150y$                    // What does $x$ equal?
$y \leftarrow x + y$                   // What does $y$ equal?
```

Exercise 5: Now create a new class called `VariablesDemo`. Fill in code to do the calculations you worked out above and print out the answers. Be careful! You might get some unexpected results…

Exercise 6: Discuss the following code with the person next to you, and try to answer these questions without the use of the computer:

1. Will this code compile?

2. What are the values of the variables at each stage of the program?

3. What is the effect of the extra set of braces starting on line 8 and ending on line 14?

```
1  public class VariablesAndScope {
2     public static void main(String[] args) {
3        int x = 7;
4        float y = x;
5        x = x + 3;
6        System.out.println(y);
7        System.out.println(x);
8        {
9           x = 17;
10          int m;
11          m = 0;
12          m = x + y;
13          x = x + 4;
14       }
15       System.out.println(m);
16       System.out.println(x);
17    }
18 }
```

Exercise 7: Use the space below (or your log book) to draw out the truth tables for the Boolean operators `AND`, `OR` and `NOT`.

Exercise 8: Given the following definitions:

```java
boolean a = true;
boolean b = false;
```

evaluate the following expressions:

`a && !b` ____________________

`!a || b` ____________________

`!(a || b) || a` ____________________

`!(a && b) && (!a || b)` ____________________

Exercise 9: Write a program that asks the user first for their first name, then for their last name, and finally prints out what their whole name is. You may find the code from last week's Exercise 7 helpful for getting input from the user.

Exercise 10: Now modify your program to make the program print out a new nickname for you. The program will generate the nickname by taking the first half of your first name, and joining that to the second half of your last name. The nickname will always start with an uppercase letter, and all other letters will be lowercase.

For example:

```
> java Nickname
   Please enter your first name:
   steven
   Please enter your last name:
   CAMPBELL
   Your nickname is "Stebell"
```

Oracle provides an API (Application Programming Interface) to describe all of their internal Java classes. You can find the API at http://docs.oracle.com/javase/8/docs/api/.

You will find the `String` API useful for this task as it will allow you to see the list of all the operations you can perform on a String variable with examples.

For this task, you may be particularly interested in the entries for `length`, `substring`, `toLowercase` and `toUpperCase`.

Extension 1: Write a program to print a message to say whether a given input number is prime. A prime number is one that has just itself and the number 1 as positive integer factors. The first few primes are 2, 3, 5, 7, 11, 13, 17, 19, 23. The program should print out the first (smallest) factor that divides n.

Here's some pseudocode to help you:

Algorithm 2: IsPrime (n)

```
1  if ($n \leq 1$) then
2      print ”not prime”
3      return
4  for ($i = 2$ up to $n$) do
5      if ($i$ is a factor of $n$) then
6          print ”not prime: i is a factor”        // where i is the factor
7          return
8  print ”n is prime”                              // where n is the number
```

Shortcut: actually, we only need to check upto $\sqrt{n}$ instead of n at Line 4.

Extension 2: Write a program to print out a list of all the prime numbers in the range 1 to 1000.