Lab 2 : Variables – Selected Solutions

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` Invalid. The only valid starting characters for a variable name are a letter, a dollar sign '$' or an underscore '_'.

`input-length` Invalid. You cannot use a hyphen `-` in a variable name as it would be interpreted as subracting `length` from `input`.

`theCount` Valid and follows correct camel case convention.

`UserName` Valid, but does not follow case convention. Variables should start with a lowercase letter.

`_needsDeleting` Valid, but Java convention states that we no longer start variables with an underscore '_'.

`first3Letters` Valid, though this would be better written as `firstThreeLetters`.

`mycoolvariable` Valid, but does not follow camel case convention.

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

a	b	a AND b
0	0	0
0	1	0
1	0	0
1	1	1

a	b	a OR b
0	0	0
0	1	1
1	0	1
1	1	1

a	NOT a
0	1
1	0

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.

```java
import java.util.Scanner;

public class Name {
   public static void main(String[] args) {
      Scanner keyboard = new Scanner(System.in);

      System.out.println("Please enter your first name:");
      String firstName = keyboard.next();
      System.out.println("Please enter your last name:");
      String lastName = keyboard.next();

      String fullName = firstName + " " + lastName;
      System.out.println("Your full name is: " + fullName);
   }
}
```

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`.

```java
import java.util.Scanner;

public class Nickname {
   public static void main(String[] args) {
      Scanner keyboard = new Scanner(System.in);

      System.out.println("Please enter your first name:");
      String firstName = keyboard.next();
      System.out.println("Please enter your last name:");
      String lastName = keyboard.next();

      // get the first half of the first name
      String firstHalf = firstName.substring(0, firstName.length() / 2);

      // convert it to lowercase
      firstHalf = firstHalf.toLowerCase();

      // capitalise the first letter
      char firstLetter = Character.toUpperCase(firstHalf.charAt(0));
      firstHalf = firstLetter + firstHalf.substring(1);

      // repeat for second half
      int length = lastName.length();
      String secondHalf = lastName.substring(length / 2, length);
      secondHalf = secondHalf.toLowerCase();

      String nickname = firstHalf + secondHalf;
      System.out.println("Your nickname is: " + nickname);
   }
}
```

Extension 1: Write a program to print a message to say whether an input number given 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 1: IsPrime (n)

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

```java
import java.util.Scanner;

public class CheckPrime {
   public static void main(String[] args) {
      Scanner keyboard = new Scanner(System.in);
      int n = keyboard.nextInt();
      if(n <= 1) {
         System.out.println(n + " is not prime.");
         return;
      }
      int i = 2;
      // only check up to sqrt(n) as anything larger
      //is checked implicitly by the nature of factors
      while(i <= Math.sqrt(n)) {
         if(n % i == 0) {
            System.out.println(n + " is not prime: " +
                  i + " is a factor.");
            return;
         }
         i++;
      }
      System.out.println(n + " is prime.");
   }
}
```

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

```
public class OneThousandPrimes {
   public static void main(String[] args) {
      int n = 1;
      while(n <= 1000) {
         if(checkPrime(n)) {
            System.out.println(n + " is prime.");
         }
         n++;
      }
   }

   // this method checks if the input integer is prime
   public static boolean checkPrime(int n) {
      if(n <= 1) {
         return false;
      }
      int i = 2;
      while(i <= Math.sqrt(n)) {
         if(n % i == 0) {
            return false;
         }
         i++;
      }
      return true;
   }
}
```