ISCG 8045 – Lab Sessions 1 & 2

Due Date: 7 April 2015

Weighting: 15%

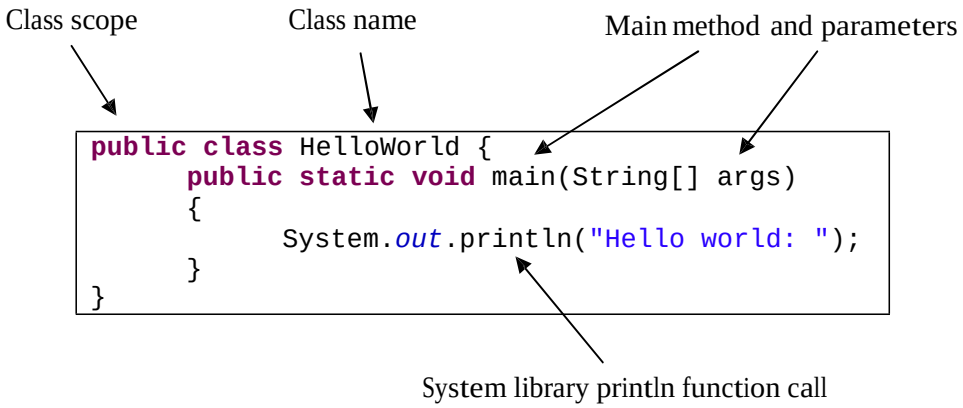
Marks: 15

The best way to learn Java is to actually grab a compiler and start coding. There is a wealth of information about Java and Java Programming available on the Internet, as specified in the lecture notes. There are nine questions in this lab. Please upload your answers on Moodle by 7 April 2015, 8:30am.

1) HelloWorld

   Read through the getting started trail on the Oracle website:
   http://docs.oracle.com/javase/tutorial/java/

   Make sure you pay attention to the section on the Java platform, so you understand how Java programs are executed using a JVM and how Java programs are compiled to bytecode. After finishing reading through these sections develop and run your own version of the HelloWorld program.



   Compile and run your program. Make sure you are able to identify the class definition, class name, class scope, main method and method scope.

2) Modify the hello world exercise so that your program is able to accept command line parameters which specify the user's name (**1 Mark**).
   For example, if the HelloWorld program were run with these parameters:

```
java HelloWorld Jack
```

   The program would output the following message.

```
Hello World: Jack
```

3) What is the output of this code if x= 6 and y = 9? What if x=5 and y=2 (**1 Mark**)?

```
If (X > Y)
{
  int t = x;
  y = x;
  x= t
  System.out.println("x = "+x+" and y= "+y);
}
else {
  System.out.println("Have a nice day!");
}
```

4) What is the output of this program if 6 is given as a parameter: `java PowersOfTwo 6`? Replace the "`while`" loop with a "`for`" loop and run the program again. Make sure it produces the same output (**2 Marks**).

```
public class PowersOfTwo {

     public static void main(String[] args) {

        // last power of two to print

        int N = Integer.parseInt(args[0]);

        int i = 0; // loop control counter

        int v = 1; // current power of two

        while (i <= N) {

             System.out.println(i + " " + v);

             i = i + 1;

             v = 2 * v;

        }

     }

}
```

5) Compile and run the following code using 3 as a parameter (`java Cubes 3`). What is the output (**1 Mark**)?

```
public class Cubes{

   public static int cube(int i) {

       i = i * i * i;

       return i;

    }

   public static void main(String[] args) {

       int N = Integer.parseInt(args[0]);

       for (int i = 1; i <= N; i++)

           System.out.println(i + " " + cube(i));
```

```
        }
    }
```

6) Define a new class: `TestCubes`. It should have a `main` method to create objects of `Cubes`. Remove the main method from `Cubes`. Compile the classes run `TestCubes` with 3 as a parameter, i.e. `java TestCubes 3`. It should produce the same output as (5) (**2 Marks**).

7) Given two vectors x[] and y[] of length N, their dot product is the sum of the products of their corresponding components. Study the following code and fill out the ? marks in the given table (**3 Marks**).

```
double[] x = { 0.3, 0.6, 0.1 };

double[] y = { 0.5, 0.1, 0.4 };

int N = x.length;

double sum = 0.0;

for (int i = 0; i < N; i++) {

     sum = sum + x[i]*y[i];

}
```

i	sum
0	?
1	?
2	?
3	?

7b) Given two N-by-N matrices a and b, define c, to be the N-by-N matrix where c[i][j] is the sum a[i][j] + b[i][j]. You may use nested `for` loops.

8) Explain what the following code does (**2 Marks**):

```
// Source code can be found at http://www.javaworld.com/javaworld/jw-07-1998/jw-07-
    exceptions.html
 class Example1 {
    public static void main(String[] args) {
        int temperature = 0;
        if (args.length > 0) {
            try {
                temperature = Integer.parseInt(args[0]);
            }
            catch(NumberFormatException e) {
                System.out.println(
                    "Must enter integer as first argument.");
                return;
           }
        }
        else {
           System.out.println("Must enter temperature.");
            return;
        }
       // Create a new coffee cup and set the temperature of
```

```
            // its coffee.
             CoffeeCup cup = new CoffeeCup();
             cup.setTemperature(temperature);
            // Create and serve a virtual customer.
              VirtualPerson cust = new VirtualPerson();
              VirtualCafe.serveCustomer(cust, cup);
        }
    }
```

Using the example above add a `try, catch` block to this code, so that it throws `ArrayIndexOutOfBoundsException` if `N > 3:`

```
public class exceptions{

   public static void main(String Args[]){

   int[] array = new int[3];

      for(int i=0;i<N;++i){

          array[i] = i;

      }

   System.out.println(array);

  }

}
```

9) Modify the following program so that the system prints out the factors of the composite numbers. Hint use an ArrayList to keep track of all the factors used to test primality in the inner for loop. Note the inner loop speeds things up using a break statement which will need to be removed to find all the required factors (**3 Marks**).

```
// define a public class named Prime
public class Prime {
        // generic main method
        public static void main(String[] args)
        {
                // Simple program to identify the first n prime numbers

                // A prime number is a number that is only divisable by 1 or itself
                // See http://en.wikipedia.org/wiki/Prime_number for more information
                // A factor is any number that equally divides another another number

                // initially we are only interested in the first 20 numbers
                int n = 20;

                // start the loop from 2
                for (int number = 2; number<=20;number++)
                {
                        // simple flag to toggle between prime and composite
                        boolean isPrime = true;

                        // what are the factors of the variable number?
                        for (int factor = 2;factor <number; factor++)
                        {
                                /*
                                 * test if a particular number / factor combination
results in                               * a remainder if the number is equally divided by
a factor
                                 * higher than 1 then it must be a composite number */

                                if(number % factor == 0)
                                {
                                        isPrime = false;
```

```
                                        break;
                                }
                        } // end inner factor loop

                        if(isPrime)
                        {
                                System.out.println(number+" is prime.");
                        }
                        else
                        {
                                System.out.println(number+" is composite.");
                        }
                } // end outer number loop
        } // end main method
} // end class
```

The output for the first 10 numbers should look like this:

```
2 is prime.
3 is prime.
4 is composite.
The factors include [2]
5 is prime.
6 is composite.
The factors include [2,3]
7 is prime.
8 is composite.
The factors include [2,4]
9 is composite.
The factors include [3]
10 is composite.
The factors include [2,5]
```