Lab 1: Writing, Compiling and Executing a Java Program

Lab Objectives: After this lab, the students should be able to:

1. Write a simple Java application
2. Compile a Java application in the JDK
3. Run Java applications in the JDK
4. Program arguments
5. Write, compile and debug a simple Java application using an IDE (student presentation)
6. Programming refresher exercises

1. Installing and using the JDK

JDK, which stands for Java Development Kit, consists of the Java commands and libraries needed to develop and run Java applications. The JDK is free and can be downloaded from several websites. You can get the JDK from Oracle through this link:

www.oracle.com/technetwork/java/javase/downloads/index.html

The JDK offers a set of tools that can be run from a command line. We will be using two principle applications.

Once installed, check that your JDK is accessible. In the console type

```
java -version
```

If a message stating that the command is unknown, the path to the bin folder of JDK has to be added to the PATH global variable.

2. Writing and running your first Java program (using console)

Using a text editor, write the following code and save it into a file.

```
/* This is a simple Java program */
class MyFirstApp {

     // a java program begins with a call to main()
     public static void main(String args[]){
          String message="Hello world";
          System.out.println(message);
     }

}
```

Note:

- The file name has to be the same as the class name (Here, MyFirstApp)

- The file extension has to be **.java**

Compiling the program

To compile the Java program, use the **javac** command supplied by the JDK. In the command line type:

```
javac MyFirstApp.java
```

The compilation yields a **Bytecode** file with the same name as the compiled file and **.class** as extension. Check that your compilation is completed successfully.

Running the program

To run the Java program, the Java Virtual Machine is required. JVM is part of the JDK and it can be invoked with the command `java`, where the Bytecode file is provided as parameter. In the console type:

```
Java MyFirstApp
```

Note: do not append any extension to the file name.

Passing parameters to the program

In the main method : `public static void main(String args[])`

The args (arguments) is an array of strings. It serves to pass parameters to the program via the command line. You can provide as many arguments as you need when you run the program:

```
Java MyFirstApp arg1 arg2 arg3 …
```

In your program the arguments are accessed by `args[0], args[1], args[2], etc.`

Change your first program so that you provide your first name, last name and student ID as arguments and the program prints them.

3. Using an IDE with Java

An Integrated Development Environment (IDE) is a software which offers set of tools which makes the application development easier and faster. Many IDEs, like Eclipse, JavaBeans, VSCode, etc. supports Java.

Student presentation setting up and using an IDE

Pick an IDE of your choice (the one that you have already used, for example) and explore the following tasks: Create a first project, write the first program, compile the program, run the program and debug the program.

4. Programming refresher exercises

write a Java programs that solves the following tasks

1. Calculates and displays the circumference and the area of a circle of a given diameter. The diameter value is given by the user as an input. Define and use a constant which represents an approximation of π to the fifth decimal digit (3.14159).
2. Reads from the user three integer numbers then returns the greatest, the smallest and the average of these numbers.
3. Read an integer number and find whether it is odd or even.
4. Write a Java program to read an integer number and find the sum of the digits of this number. Example: input=123, output=6 (1+2+3).
5. Determine whether an input numbers is Armstrong. An Armstrong number is a three digit number which is equal to the sum of the cubes of its digits. Example: $153=1^3+5^3+3^3$.