1DV506, Problem Solving and Programming, Autumn 2017

Assignment 1: Getting Started

Problems?

Do not hesitate to ask your teaching assistant at the practical meetings (or Jonas at the lectures) if you have any problems. You can also post a question in the assignment forum in Moodle.

Exercises

Lecture 1 (Getting Started)

1. **Install Java**
   Download and install Java SE JDK: www.oracle.com/technetwork/java/javase/downloads. A new version (Java 9) was recently released. We recommend you to use the previous version (Java SE 8u151/ 8u152). Also, there are plenty of instruction videos available in YouTube. Just search for "Install Java X" where X is your operating system.

2. **Install Eclipse**
   Download and install Eclipse IDE for Java Developers: http://www.eclipse.org/downloads/. We recommend you to use version Oxygen. Once again, there are plenty of instruction videos available in YouTube. Just search for "Install Eclipse X" where X is your operating system.

3. **Setup Eclipse Workspace**
   Before you start programming, do the following.
   - Create an Eclipse *workspace* (a folder) with the name `java_courses` on some location in your home directory.
   - Create a *Java project* with the name `1DV506` inside the workspace.
   - Create a *package* with the name `YourLnuUserName_assign1` inside the project. For example, it might look something like `wo222ab_assign1`.
   - Save all program files from the exercises in this assignment inside the package `YourLnuUserName_assign1`.
   - In the future: create a new package (`YourLnuUserName_assignX`) for each assignment and a new project (with the course code as name) for each new course using Java.

4. **Edit, compile and execute.**
   Create, compile and execute the following program inside your assignment 1 package:

```
/* The classical "Hello World!" program. */
public class Hello {

    public static void main(String[] args) {
       System.out.println("Hello World!"); // Print
    }
}
```

Lecture 2 - (Input/Output, Operations on Primitive Types)

5. **Printing**
   Write a program `Print.java`, which will print the phrase *Knowledge is power!*

- on one line,
- on three lines, one word on each line,
- inside a rectangle made up by the characters = and |.

6. **Quote**
   Write a program `Quote.java` which reads a line of text (using class `Scanner`) and then prints the same line as a quote (that is inside " "). An example of an execution:

```
Write a line of text:  I wish I was a punk rocker with flowers in my hair.
Quote: "I wish I was a punk rocker with flowers in my hair."
```

7. **Number of seconds**
   Write a program `Seconds.java` which reads three integers (hours, minutes, seconds) and then computes the corresponding time measured in seconds. For example, 1 hour, 28 minutes and 42 seconds is equal to 5322 seconds. An example of an execution:

```
Hours: 1
Minutes: 28
Seconds: 42

Total time measured in seconds: 5322
```

8. **BMI**
   Write a program `BMI.java` which computes the BMI (Body Mass Index) for a person. The program will read length and weight from the keyboard and then present the result as output. The BMI is computed as weight/(length)^2, where the length is given in meters and the weight in kilograms. An example of an execution:

```
Give your length in meters: 1,83
Give your weight in kilograms: 83
Your BMI is: 25
```

   Note: the BMI is always an integer.

9. **Time**
   Write a program `Time.java`, which reads a number of seconds (an integer) and then prints the same amount of time given in hours, minutes and seconds. An example of an execution:

```
Give a number of seconds: 9999
This corresponds to: 2 hours, 46 minutes and 39 seconds.
```

   Hint: Use integer division and the modulus (remainder) operator.

10. **Sum of Three**
    Write a program `SumOfThree.java` which asks the user to provide a three digit number. The program should then compute the sum of the three digits. For example:

```
Provide a three digit number: 483
Sum of digits: 15
```

If Time Permits

Exercise 11 is marked as *VG task* ==> only mandatory for those of you that aspire for a higher mark (A or B).

11. **Change** (VG-task)
    Write a program `Change.java` that computes the change a customer should receive when he has paid a certain sum. The program should exactly describe the minimum number of Swedish bills and coins that should be returned rounded off to nearest krona (kr). Example:

```
Price: 372.38
Payment: 1000

Change: 628 kronor
1000kr bills: 0
 500kr bills: 1
 100kr bills: 1
  50kr bills: 0
  20kr bills: 1
  10kr coins: 0
   5kr coins: 1
   1kr coins: 3
```

Lecture 3 - Using Library Classes

12. **Fahrenheit to Celsius**
    Write a program `Convert.java`, which reads a temperature in Fahrenheit and then converts it to Celsius using the formula:

```
C = (F-32)*5/9
```

    The result should be presented with a single decimal correctly rounded off.

13. **Short Name**
    Write a program ShortName.java, reading a first name and a last name (given name and family name) as two Strings. The output should consist of the first letter of the first name followed by a dot and a space, followed by the first four letters of the last name. An example of an execution:

```
First name: Anakin
Last name: Skywalker
Short name: A. Skyw
```

    Hint: Use methods of the String class.
    What will happen if the last name consists of less than four letters?

14. **Random Number**
    Write a program `TelephoneNumber.java`, generating and printing a random telephone number of the form `0XXX-ZYYYYY`. The area code consists of a zero followed by three digits (X). The local number can not start with a zero (Z), all other digits (Y) are random.
    Hint: Use the class `java.util.Random`.

15. **Square Root**
    Write a program `Distance.java` which reads two coordinates in the form (x,y) and then computes the distance between the points, using the formula

```
distance = Sqrt( (x1-x2)^2 + (y1-y2)^2 )
```

    Sqrt() means "the square root of" and ^ means "raised to". The answer should be presented with three decimal digits.
    Hint: Use the class `java.lang.Math` for the computations.

If Time Permits

Exercise 16 is marked as *VG task* ==> only mandatory for those of you that aspire for a higher mark (A or B).

16. **Wind Chill** (VG-task)
    Write a program `WindChill.java` that asks the user for a temperature (°C) and the wind speed (measured in m/s) and then computes the so-called *wind chill temperature* Twc using the Wind Chill Index formeln:

    ```
    Twc = 13.12 + 0.6215*T - 11.37*V^{0.16} + 0.3965*T*V^{0.16}
    ```

    where T is the temperature (Celsius) and V is the wind speed (kilometers per hour). Notice you must convert the wind speed from meter per second to kilometers per hour before you can apply the formula. And we expect you to present the result with one decimal (correctly rounded off). For example:

    ```
    Temperature (C):  -7.8
    Wind speed (m/s):  8.4

    Wind Chill Temperature (C): -16.7
    ```

Submission

Only exercises 5-16 should be handed in. (Notice that the VG exercises 11 and 16 are not mandatory.) We are only interested in your .java files. Hence, zip the directory named `YourLnuUserName_assign1` (inside directory named `src`) and submit it using the Moodle submission system.