CentraleSupélec

IS1220 - Object Oriented Software Design

Tutorial 01

Paolo Ballarini

General instructions:

- If you have not done so already, create a workspace for your tutorial `IS1220/TDs`.
- Within the workspace, create a new package for this TD called `fr.ecp.IS1220.TD01/`.
- Carefully document your code (JavaDoc), i.e., explain the general idea of your algorithm, explain the relevant steps in the code implementing the algorithm, document assumptions (if any), corner cases, and error conditions.

Learning outcomes:

- getting familiar with notion of Java class
- learning how to use basic input/output functionalities in Java
- arbitrary precision operations on real-valued variables (`BigDecimal` class)

Exercise 1. A simple HelloWorld program with input/output

Using the ECLIPSE Integrated Development Environment (IDE) create a new JAVA project called "Tutorial1" (File->New->Java project)

- In the newly created "Tutorial1" project, add a new package named `fr.ecp.is1220.helloworld`. For this in the Package Explorer frame of Eclipse right-click on the `src` folder within the "Tutorial1" project then New->Package.
- Add a new class named "HelloWorld" to the `fr.ecp.is1220.helloworld` package. For this select the package `fr.ecp.is1220.helloworld` then right click and New->Class, then enter the name of the class (remember that by convention class' names must begin with a capital letter).
- In the "HelloWorld" class add a `main` method that displays on the screen the message "HelloWorld" (**hint**: use Eclipse auto-completion for adding the `main` method: simply type `main` then control+space).
- Extend the program so that it asks the user to input a (string) message and then displays it on the screen (**hint**: use Eclipse auto-completion for adding the `System.out.println` method: simply type `sysout` then control+space).
- Extend the program so that it also displays on screen the value of the π constant
- Extend the program so that it asks the user to input an integer value n and then displays on the screen a random number between 0 and n.

Exercise 2. Palindrome

Problem statement. Write a program that determine whether a word or sentence is a *palindorme* (a palindrome reads backwards as it does forwards, e.g. “radar” is a palindrome word and “Able was I ere I saw Elba” is a palindrome sentence). Blank spaces should be considered as significant.

Design. Start by sketching the algorithm (written in natural language) that provides us with a solution to the stated problem.

Exercise 3. A Circle class

- in the "Tutorial1" project you created before, add a new package named `fr.ecp.is1220.circle`.
- in package `fr.ecp.is1220.circle` add a class named "Circle" whose elements are described by the (UML) Class Diagram in Figure 1.

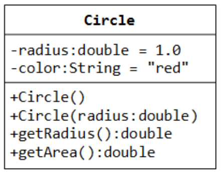

Figure 1: Class diagram for the "Circle" class

The characteristics of the elements of class Circles are as follow:

- fields `radius` and `colour` are private
- define two overloaded *constructors*:
    * first constructor sets both *radius* and *colour*
    * second constructor asks for *radius* but set *colour* do default value "red".
- a public method `getRadius()` for retrieving the *radius*
- a public method `getArea()` for calculating the *area* of the circle

- To test the `Circle` class create a `TestCircle` class containing a `main` method in which:
    - declare the following circle objects,
        * c_1 with radius 1 and colour "red"
        * c_2 with radius 2 and colour "red"
        * c_2 with radius 8.3 and colour "yellow"

      display on screen the radius and the area of each circle.

Exercise 4. Square inscribed into Circle (and viceversa).
Using the previously defined `Circle` class add a new class called `Square` that can be used to model square polygons

- define appropriate fields to represent the "geometric characteristics" of a square (beware that different modeling solutions are available)
- define a constructor that allows to set all relevant geometric characteristic of a square object
- add to the `Square` class a public method `inscribed(Circle c)` that takes as input a `Circle` object `circle` and returns a `boolean` value if the circle can be inscribed into the square
- add to the `Circle` class a public method `inscribed(Square s)` that takes as input a `Square` object `square` and returns a `boolean` value if the square can be inscribed into the circle

To test the `Square` class extend the `main` method by:

- declaring the following square objects:
  - s_1 of side equal to `sqrt(2)`
  - s_2 of side equal to 4.
- then testing if:
  - square s_1 can be inscribed in circle c_1? print a message on screen according to the answer
  - square s_1 can be inscribed in circle c_2? print a message on screen according to the answer
  - circle c_2 can be inscribed in square s_2? print a message on screen according to the answer
  - circle c_1 can be inscribed in square s_2? print a message on screen according to the answer

Exercise 5. Exporting/Importing JAVA project with Eclipse.

- Once you have completed your work rename the "Tutorial1" project using "Tutorial1_YOURNAME" as the new name. Then export the "Tutorial1_YOURNAME" project into an archive file named "Tutorial1_YOURNAME.zip". (**hint**: to export a project to a .zip file: Right-click->Export->General->Archive File)
- now remove your the Tutorial1_YOURNAME project from the Eclipse workspace.
- import the project previously saved under "Tutorial1_YOURNAME.zip" in the Eclipse current workspace (**hint**: to import a project from a .zip file:File->Import->General->Existing Projects into workspace->Select archive file)