Faculty of Science and Engineering

COMP125 Fundamentals of Computer Science
Workshop Week 3

Learning outcomes

By the end of this session, you will know some of Java basics. In particular, you will be able to design and write simple Java classes.

Questions

1. **Import-Export**

   It is important to know how to import Java projects from archive files (.jar/ .zip) and how to export your project(s) into archive files. First. we'll learn how to import a project.

   a. Click "File" –> "Import" –> "Existing Projects into Workspace"
   b. Select option "Select Archive file" and click on "Browse"
   c. Choose the archive files (".zip") that contains project(s) you want to open. Please note an archive file may contain multiple projects and click "ok"
   d. Check all projects you want to import and click "Finish"

   Practice the above procedure using file `classesAndObjectsProgram.zip` file uploaded on iLearn under Section 3. You should see a project `classesObjectsProject` if correctly imported.

   Next, we'll learn how to export a project.

   a. Click "File" –> "Export" –> "General" –> "Archive file"
   b. Select all projects you want to export in the archive file in the left panel
   c. In the "To archive file" section, choose file path and name and click "Finish"

   Export the project `classesObjectsProject` to an archive file `<studentID>.zip` where <studentID> is your Student ID.

2. (Problem solving and loops) Write a function (or method) that when passed an integer, returns the number of times it has to be divided by two, to reach zero. For example, 19 (or -19) needs to be divided by two, five times, to reach zero.

   a. 19/2 = 9,
   b. 9/2 = 4,
   c. 4/2 = 2,
   d. 2/2 = 1,
   e. 1/2 = 0

3. Design classes (no implementation) that encapsulate the following real life entities. Add up to three data members for each class. Select the three most important attributes if you think a class has more than three attributes. Describe your design in terms of a UML class diagram as shown in week 2 lecture.

   a. Person

   b. Cylinder

   c. Book

4. (a) Consider the following class definition,

```
public class Car {
        public String model;
        public int price;
}
```

   Declare and instantiate an object `myCar` of class `Car`. Assign the value `Corolla` to the data member `model` and the value 21999 to the data member `price` of object `myCar`.

   (b) Consider the following class definition,

```
public class Date {
        public int day, month, year;
}
```

   Declare and instantiate an object `graduation` of class `Date`. Assign values to data members of object `graduation` such that it represents the date 13th April, 2011.

5. (a) Consider the following class definition,

```
public class Time {
         public int hour, minute, second;
}
```

   Explain why it's a bad idea for the data members to be public, by writing a client that is malicious and assigns invalid values to the daa members of `Time` object.

   (b) Solve the problem of `public` data members in the previous part by first changing visibility of the data members of class `Time` to `private` and then adding getters and setters. The setter for `hour` should constrain the passed value in the range [0, 23]. That is, if the passed value is less than 0, `hour` should be 0, otherwise if the passed value is more than 23, `hour` should be 23, otherwise `hour` should become the passed value. Similarly, the setters for `minute` and `second` should constrain the passed value in the range [0, 59].

   (c) Declare, instantiate an object `t1` of class `Time` written in the previous part. Assign values to the data members such that it represents the time 19:30:45 (half past seven in the evening and another 45 seconds).

   (d) Declare and instantiate an object `t2` of class `Time` written in the previous part. Assign 95 to `hour`, -78 to minute, and 55 to second. Display all data members on the console. What time would `t2` represent?

   (e) List the mistakes (syntactical and logical) in the following constructor for class `Time` -

```
public void time(int h) {
         hour = h;
         minute = 0;
         second = 0;
}
```

   (f) Add two constructors to class `Time` with the following requirements:

   - A constructor that is passed three parameters, one for each data member.
   - A constructor that is passed two parameters, for hour and minute, and sets seconds to 0.

   (g) Assuming the two constructors have been added to class `Time` according to previous part. Will the following program run successfully, or result in a compilation error? Explain your answer. Also, if there is a compilation error, what should be done to fix it?

```
Time ourTime = new Time();
```

6. **Assessed exercise**

a. Write a class definition for a `Line` in the first quadrant, as represented by its end points (x_1, y_1) and (x_2, y_2) such that $x_1, y_1, x_2, y_2 \geq 0$. It should include,

   (a) Correct class header.

   (b) Data members with appropriate visibility and data types.

   (c) Getters

   (d) Setters

   (e) Constructors

      i. With no parameters. First and second points should be $(0,0)$.

      ii. With two parameters for (x_1, y_1). Second point should be $(0,0)$.

      iii. With four parameters for $(x_1, y_1), (x_2, y_2)$.

   (f) A method `getLength()` that returns the length of the line as computed by the pythagorean formula

   $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.
   You can compute square root of a numerical value *val* by using `Math.sqrt(val)`.

   (g) The method `toString()` that returns the `String` representation of object of class `Line`. For example, if a line goes from (1.5, 2.5) to (8, 6.5), the method should return "`(1.5, 2.5) to (8, 6.5)`".

b. Write a piece of client code that -

   (a) Declare and instantiate an object `myLine` of class `Line` that represents a line from (4.2, 3.5) to ((6.5, 1.2)

   (b) Display the details of the object `myLine`. Which method is used? Do you call that method explicitly?

   (c) Store the length of `myLine` in variable `myLength` and display it on the console.

   (d) Change one of the end-points of the line from the previous coordinate of (4.2, 3.5) to a new coordinate (1.5, 7.5), and display the details of the line once more.