



COMP-455  
OOP using Java Language

EXERCISES/2<sup>ND</sup> SET PART (B)

<b>Lecturer:</b>	<i>Dr. Constandinos X. Mavromoustakis</i>
<b>Skills Examined:</b>	<i>Java Classes and Java methods and types/ Object-Oriented Programming: Inheritance, early objects</i>
<b>Date of release:</b>	20/11 □

## Java Exercises – simple objects and classes

### Ex1

(The [MyTriangle](#) class)

Create a class named [MyTriangle](#) that contains the following two methods:

```
/** Returns true if the sum of any two sides is
```

```
* greater than the third side. */
```

```
public static boolean isValid(  
    double side1, double side2, double side3)
```

```
/** Returns the area of the triangle. */
```

```
public static double area(  
    double side1, double side2, double side3)
```

The formula for computing the area is

$$s = (side1 + side2 + side3)/2;$$

$$area = \sqrt{s(s - side1)(s - side2)(s - side3)}$$

Write a test program that reads three sides for a triangle and computes the area if the input is valid. Otherwise, it displays that the input is invalid.

## Ex2

(The `Triangle` class) Design a class named `Triangle` that extends `GeometricObject`. The class contains:

- Three `double` data fields named `side1`, `side2`, and `side3` with default values `1.0` to denote three sides of the triangle.
- A no-arg constructor that creates a default triangle.
- A constructor that creates a rectangle with the specified `side1`, `side2`, and `side3`.
- The accessor methods for all three data fields.
- A method named `getArea()` that returns the area of this triangle.
- A method named `getPerimeter()` that returns the perimeter of this triangle.
- A method named `toString()` that returns a string description for the triangle.

For the formula to compute the area of a triangle, see [Exercise 1](#). The `toString()` method is implemented as follows:

```
return "Triangle: side1 = " + side1 + " side2 = " + side2 +  
" side3 = " + side3;
```

## Ex3

(The `Person`, `Student`, `Employee`, `Faculty`, and `Staff` classes) Design a class named `Person` and its two subclasses named `Student` and `Employee`. Make `Faculty` and `Staff` subclasses of `Employee`.

A person has a name, address, phone number, and email address. A student has a class status (freshman, sophomore, junior, or senior). Define the status as a constant. An employee has an office, salary, and date-hired. Define a class named `MyDate` that contains the fields `year`, `month`, and `day`. A faculty member has office hours and a rank. A staff member has a title. Override the `toString` method in each class to display the class name and the person's name.

## Ex4

Identify the problems in the following classes:

```
1 public class Circle {
2     private double radius;
3
4     public Circle (double radius) {
5         radius = radius;
6     }
7
8     public double getRadius() {
9         return radius;
10    }
11
12    public double getArea() {
13        return radius * radius * Math.PI;
14    }
15 }
16
17 class B extends Circle {
18     private double length;
19
20     B(double radius, double length) {
21         Circle(radius);
22         length = length;
23     }
24
25     /** Override getArea() */
26     public double getArea() {
27         return getArea() * length;
28     }
29 }
```

## Ex5

(The **Fan** class)

Design a class named **Fan** to represent a fan. The class contains:

- Three constants named **SLOW**, **MEDIUM**, and **FAST** with values **1**, **2**, and **3** to denote the fan speed.

- An `int` data field named `speed` that specifies the speed of the fan (default `SLOW`).
- A `boolean` data field named `on` that specifies whether the fan is on (default `false`).
- A `double` data field named `radius` that specifies the radius of the fan (default `5`).
- A string data field named `color` that specifies the color of the fan (default `blue`).
- A no-arg constructor that creates a default fan.
- The accessor and mutator methods for all four data fields.
- A method named `toString()` that returns a string description for the fan. If the fan is on, the method returns the fan speed, color, and radius in one combined string. If the fan is not on, the method returns fan color and radius along with the string "fan is off" in one combined string.

### Ex6 (The `MyInteger` class)

Design a class named `MyInteger`. The class contains:

- An `int` data field named `value` that stores the `int` value represented by this object.
- A constructor that creates a `MyInteger` object for the specified `int` value.
- A get method that returns the `int` value.
- Methods `isEven()`, `isOdd()`, and `isPrime()` that return `true` if the value is even, odd, or prime, respectively.
- Static methods `isEven(int)`, `isOdd(int)`, and `isPrime(int)` that return `true` if the specified value is even, odd, or prime, respectively.
- Static methods `isEven(MyInteger)`, `isOdd(MyInteger)`, and `isPrime(MyInteger)` that return `true` if the specified value is even, odd, or prime, respectively.
- Methods `equals(int)` and `equals(MyInteger)` that return `true` if the value in the object is equal to the specified value.
- A static method `parseInt(int)` that converts a string to an `int` value.