Java Programming I: Exercise 10

The deadline for submissions is one week after the exercise.

The aim of this exercise is to learn the concept of interface.

Source code of the lecture examples

- Relatable interface (one file implementation): OneFile.java
- Relatable interface (multiple file implementation): Relatable.java Point.java RectanglePlus.java TestRelatable.java

[Core Set, Problem 1] The Relatable Interface

(3 points)

Download the full version of the *Relatable* interface from the course Web site OneFile.java and run on your machine using Jeliot3 to understand it clearly. What should you change to use the perimeter of a *Rectangle* instead of the area, when you want to compare the "size" of objects? Please apply these changes to the *MyRectangle* class. You should propose a dataset (an array of objects of the *MyRectangle* class) to test your solution and create a demonstration program (*TestRelatable.java*). Your program should find the largest object in the array and print the value of its perimeter.

Your Java code for each class should be saved to a separate file.

[Core Set, Problem 2] The Relatable Interface: A BMI Application

(5 points)

You should create a *Student* class with fields as follows: name, age, weight, and height. This class should implement the *Relatable* interface. To compare the "size" of two objects use the body mass index (BMI). BMI is a measure of body fat based on height and weight that applies to adult men and women. The formula to calculate BMI is:

$$BMI = \frac{weight\ (kg)}{height^2\ (m^2)}$$

Fig. 1 The formula to calculate BMI

To test your solution, create a program comparing your BMI with data of your brother / sister / friend. The name of the file is *TestBMI.java* .

Your Java code for each class should be saved to a separate file.

[Advanced Set, Problem 3] Area Values of the Figure Objects

(10 points)

Your task is to write classes for figures: *Point*, *Circle*, and *Rectangle*. They have to implement the following interface:

```
interface Shape {
   public double getArea();
}
```

There are different approaches to design classes for the aforementioned graphical figures. One approach was

introduced in Exercise 6. Another one is presented in Chapter 4 of your text book. You are free to choose the approach.

The figures can be characterized as follows:

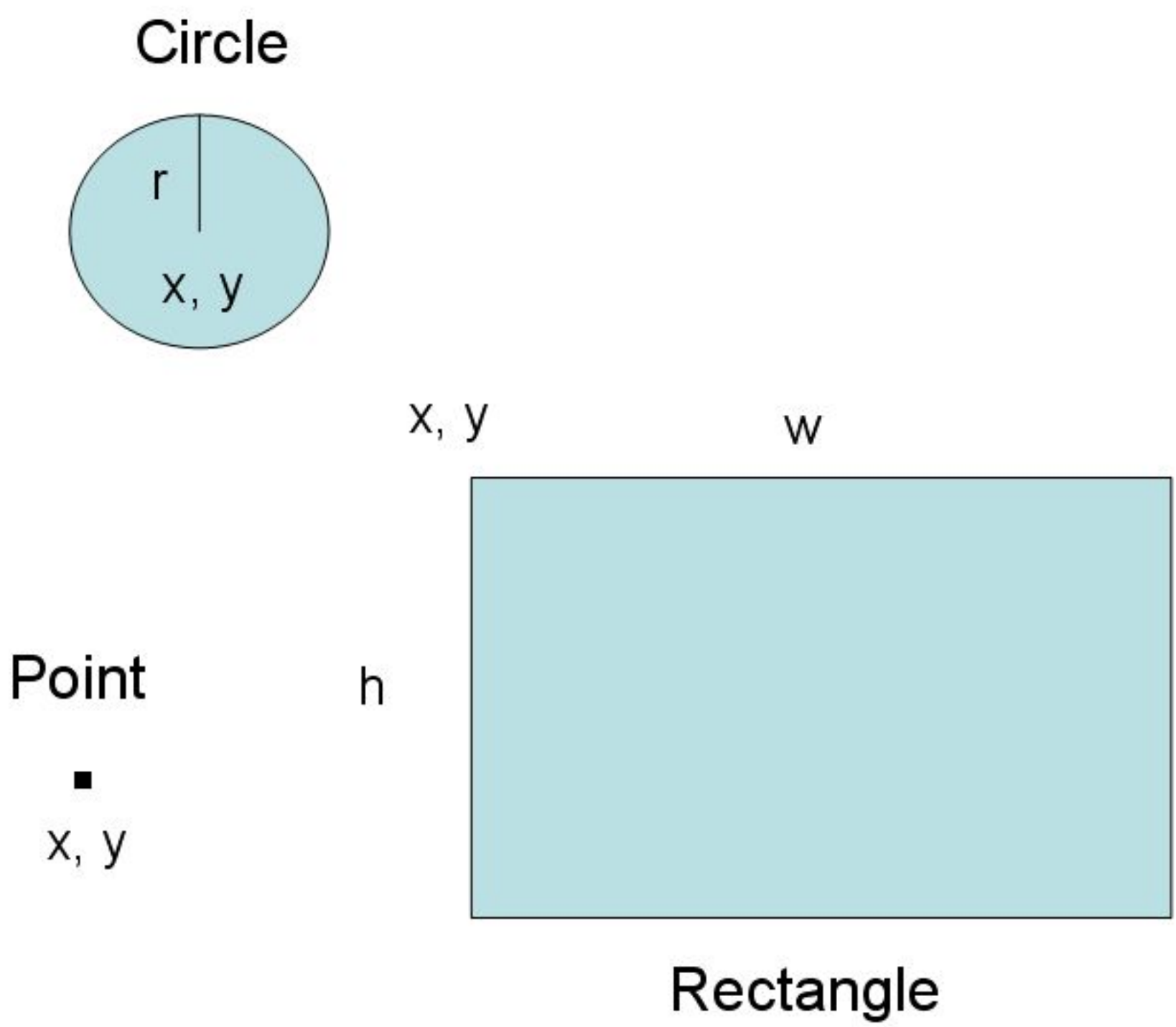

Fig. 2 Figure objects

Your application should create 3 objects of the *Point* class, 4 objects of the *Circle* class, and 5 objects of the *Rectangle* class. Then it should store created objects in an array of Objects, find the sum of area values of the figure objects, and print them out.

The file name for your application is *TestShape.java*. Your Java code for each class should be saved to a separate file.

Note: The part of your application responsible for finding the sum should not know the places of objects in the array.