Homework Assignment 2

Late homework assignments will not be accepted, unless you have a valid written excuse (medical, etc.). You must do this assignment alone. No team work or "talking with your friends" will be accepted. No copying from the Internet. Cheating means zero.

Create a new Eclipse workspace named "**`Assignment2_1234567890`**" on the desktop of your computer (replace **`1234567890`** with your student ID number). For each question below, create a new project in that workspace. Call each project by its question number: "**`Question1`**", "**`Question2`**", etc. Answer all the questions below. At the end of the assignment, create a ZIP archive of the whole workspace folder. The resulting ZIP file must be called "**`Assignment2_1234567890.zip`**" (replace **`1234567890`** with your student ID number). Upload the ZIP file on iSpace.

Here are a few extra instructions:

- Do not forget to write tests for all the code of all the classes.

- Give meaningful names to your variables so we can easily know what each variable is used for in your program.

- Put comments in your code (in English!) to explain WHAT your code is doing and also to explain HOW your program is doing it.

- Make sure all your code is properly indented (formatted). Your code must be beautiful to read.

Failure to follow these instructions will result in you losing points.

Due Date: **23:50 pm on Apr 3rd (Sunday)**.

Question 1

Create a class **`Sport`** with the following UML diagrams.

```
+------------------------------------------+
|              Sport                       |
+------------------------------------------+
| - playerNumber: int                      |
+------------------------------------------+
| + Sport(int playerNumber)                |
| + getPlayerNumber(): int                 |
| + isFun(): boolean                       |
| + testSport(): void                      |
+------------------------------------------+
```

The **`isFun`** method of the **`Sport`** class returns a boolean indicating whether the sport is fun or not: some sports are fun and some sports are not fun. The **`isFun`** method in the Sport class returns **`false`**.

Add a class **`Tennis`** to your program. **`Tennis`** is a **`sport`**. The constructor for the **`Tennis`** class takes no argument. Tennis is usually played by "**`two`**" players and is **`fun`**.

Add a class **Running** to your program. **Running** is a **sport**. The **Running** class has a private double instance variable **length** that describes the length of running, and a **getLength** method. The constructor for the **Running** class takes the sport’s player number and the its **length** as arguments. **Running** is **not fun**.

Add a class **Marathon** to your program. **Marathon** is a special form of **Running**. The **Marathon** class has two constructors: the first constructor takes the **Marathon’ s** player number and the **Marathon**’s**length** as arguments; the second constructor takes the **Marathon’ s** player numbers as argument and always uses **42.195** as its **length**. The second constructor must use the first constructor.

Add a class **HalfMarathon** to your program. **HalfMarathon** is a special form of **Marathon**. The constructor for the **HalfMarathon** class takes no argument. **HalfMarathon** always have player number 99 and length of 21.0975.

Do not forget to add individual **test methods** to the above classes.

Add a class **Start** to your program to test all your classes.

Question 2

Add a class **SportsCourt** to your program with the following UML diagram:

```
+------------------------------------------+
|              SportsCourt                 |
+------------------------------------------+
| - sport: Sport                           |
+------------------------------------------+
| + SportsCourt (Sport sport)              |
| + playSport(): Sport                     |
| + testSportsCourt (): void               |
+------------------------------------------+
```

In the **testSportsCourt** method, create a half marathon object called **hm1**, then create a SportsCourt with this object in it. Then get the sport by calling playSport and store it into a local variable called **hm2** of type **HalfMarathon**. Use the == operator to check that **hm1** and **hm2** are the same half marathon.