Lab 3 Requirements

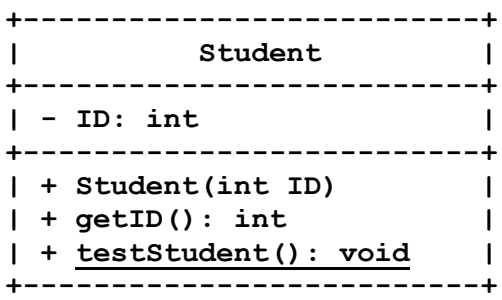
Create a new Eclipse workspace named "**Lab3_1234567890**" on the desktop of your computer (replace **1234567890** with your student ID number). For each question below, create a new project in that workspace. Call each project by its question number: "**Question1**", "**Question2**", etc. If you do not remember how to create a workspace or projects, read the "*Introduction to Eclipse*" document which is on iSpace. Answer all the questions below. At the end of the lab, create a ZIP archive of the whole workspace folder. The resulting ZIP file must be called "**Lab3_1234567890.zip**" (replace **1234567890** with your student ID number). Upload the ZIP file on iSpace.

Question 1

Create a class **Student** that contains the following things:

- a private integer ID number;
- a constructor that takes as argument an ID number and uses this ID number to initialize the object's ID number;
- a public method **getID** that takes zero argument and returns the ID number of the student;
- a public static method **testStudent** that contains tests for your class.

Here is the corresponding UML diagram (remember that **+** means public and **−** means private):



The **Student** class does not have a **setID** method because the ID number of a student never changes.

How many tests should the **testStudent** method contain?

Once you have written the **Student** class, you can test it by adding the following code in a new class called **Start** in the same project:

```
public class Start {
	public static void main(String[] args) {
		Student.testStudent();
	}
}
```

This code calls the static **testStudent** method of the **Student** class, which should then run all your tests.

What is the problem if you make the **ID** instance variable public?

Question 2

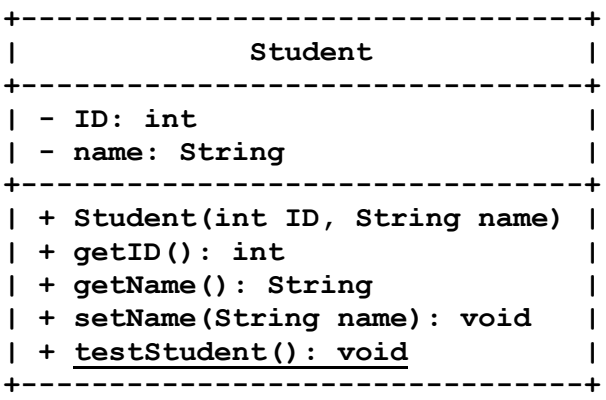
Modify the constructor of the **`Student`** class so that negative ID numbers are not allowed when creating a **`Student`** object. If the constructor is given a negative ID number as argument then the constructor should use **`0`** for the ID number.

Modify your **`testStudent`** method to test the new code. How many tests should the **`testStudent`** method now contain?

Question 3

Add to your **`Student`** class a string representing the name of the student. The constructor for the class should now take the name of the student as an extra argument. Also add to your class two methods **`getName`** and **`setName`** to get and change the name of a student (a student is allowed to change name).

Here is the corresponding UML diagram:



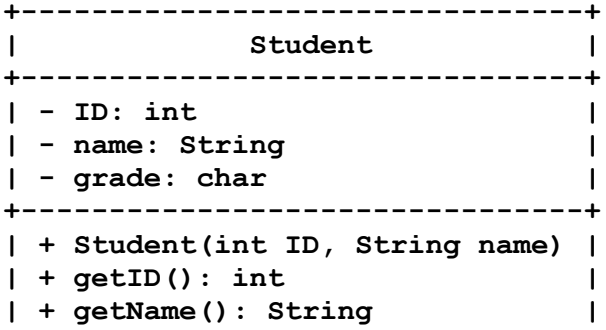
Note: we have not talked much about the **`String`** type in class yet. For now just use it like you use any other type (like **`int`** or **`float`**).

Do not forget to modify your **`testStudent`** method to test the new code. In Java you can use **`==`** to compare constant strings.

Question 4

Add to your **`Student`** class a character representing the grade of the student. The default grade when a student is created is **`'A'`**. Also add to your class two methods **`getGrade`** and **`setGrade`** to get and change the grade of a student.

Here is the corresponding UML diagram:



```
| + setName(String name): void   |
| + getGrade(): char             |
| + setGrade(char grade): void   |
| + testStudent(): void          |
+--------------------------------+
```

Do not forget to modify your **testStudent** method to test the new code. In Java you can use **==** to compare characters.

Question 5

Add a new constructor to your **Student** class that takes three arguments as input: an ID number, a name, and an initial grade. Using this second constructor it becomes possible to create **Student** objects with an initial grade which is different from **'A'**.

Here is the corresponding UML diagram:

```
+-------------------------------------------+
|                  Student                  |
+-------------------------------------------+
| - ID: int                                 |
| - name: String                            |
| - grade: char                             |
+-------------------------------------------+
| + Student(int ID, String name)            |
| + Student(int ID, String name, char grade)|
| + getID(): int                            |
| + getName(): String                       |
| + setName(String name): void              |
| + getGrade(): char                        |
| + setGrade(char grade): void              |
| + testStudent(): void                     |
+-------------------------------------------+
```

Do not forget to modify your **testStudent** method to test the new code.

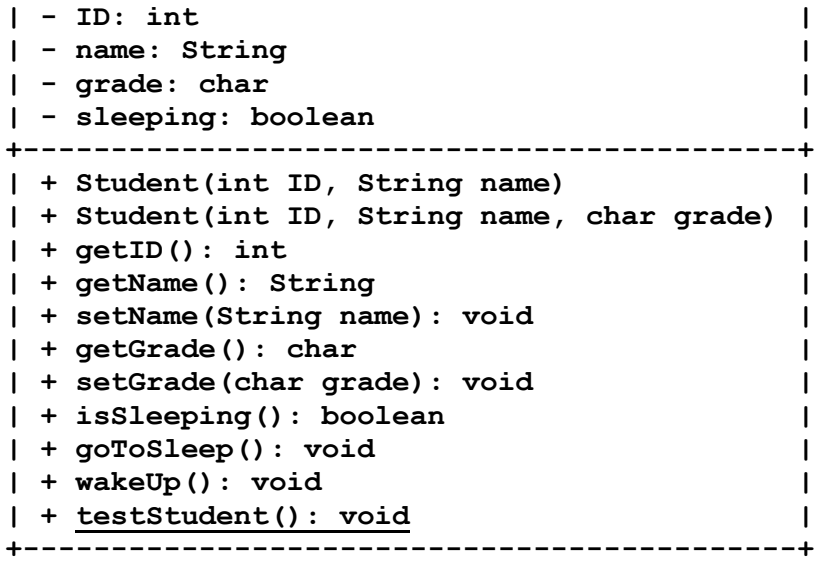
Question 6

Add to your **Student** class a boolean indicating whether the student is currently sleeping in the lab or not. When a student is created, the student is awake. Also add to your class three methods:

- one method **isSleeping** that returns a boolean indicating whether the student is currently sleeping or not.
- one method **goToSleep** that makes the student fall asleep. When a student falls asleep, the grade of the student decreases by one letter grade (**'A'** becomes **'B'**, **'B'** becomes **'C'**, **'C'** becomes **'D'**, **'D'** becomes **'F'**, **'F'** stays an **'F'**, and any other grade becomes **'F'** too).
- one method **wakeUp** that makes the student wake up. The grade of a student does not go up when the student wakes up.

Here is the corresponding UML diagram:

```
+-------------------------------------------+
|                  Student                  |
+-------------------------------------------+
```



Do not forget to modify your **`testStudent`** method to test the new code.