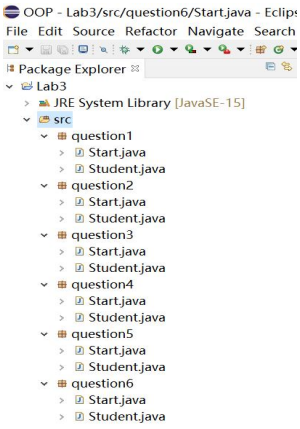
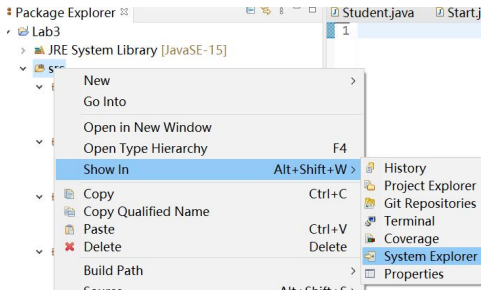
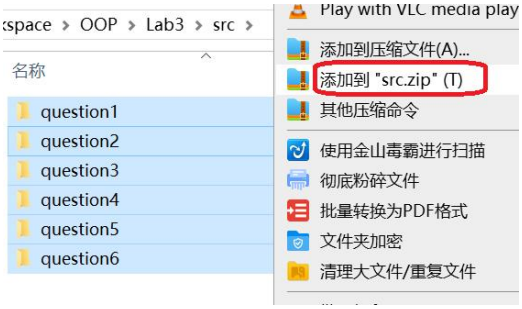
Lab 3 Requirements

Create a new Eclipse workspace named "**OOP**" on the desktop of your computer. Create a Java Project named “Lab3”. Then for each question below, create a new package (right click the src -> New -> Package). Call the package "**question1**", "**question2**", "**question3**", etc. Note: the letter q is lowercase! Answer all the questions below. Note: a question may need answers of the previous question(s), just copy paste the previous code to your new question. At the end of the lab, create a ZIP archive of all the src. In Eclipse, right click src -> Show In -> System Explorer, then zip up all folders inside the src folder. In Lab3 case, zip up question1, question2, question3, question4, question5, and question6 folders as below.







Extract your zip file to confirm that your zip file contains folders named question1, question2, question3, question4, question5, and question6, under each there are Student.java and Start.java. Rename your src.zip to "**Lab3_1234567890.zip**" (replace **1234567890** with your student ID number). Upload the ZIP file onto AutoLab. Refer to the “How to upload to AutoLab” on iSpace.

question 1 (Reviewed)

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):

```
+--------------------------+
|          Student         |
+--------------------------+
| - ID: int                |
+--------------------------+
| + Student(int ID)        |
| + getID(): int           |
| + testStudent(): void    |
+--------------------------+
```

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?

Test case: (study, type, and learn!)

```
public static void testStudent() {
      // Testing the constructor and the getID method
      Student s = new Student(9999999);
      System.out.println(s.getID() == 9999999);
}
```

question 2 (Mandatory)

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?

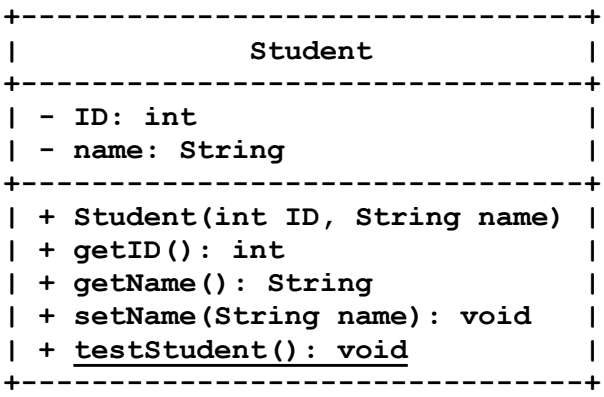
Test case: (study, type, and learn!)

```
public static void testStudent() {
      // Testing the constructor and the getID method
      Student s1 = new Student(9999999);
      System.out.println(s1.getID() == 9999999);

      // Testing the 'if' test in the constructor
      Student s2 = new Student(-9999999);
      System.out.println(s2.getID() == 0);
      Student s3 = new Student(0);
      System.out.println(s3.getID() == 0);
}
```

question 3(Mandatory)

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.

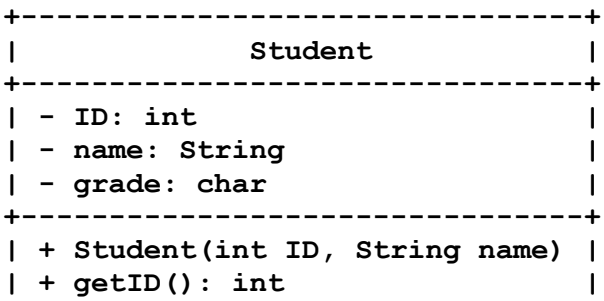
Test case: (study, type, and learn!)

```
public static void testStudent() {
      // Testing the constructor and the getID method
      Student s1 = new Student(9999999, "Philippe");
      System.out.println(s1.getID() == 9999999);

      // Testing the getName and setName methods
      System.out.println(s1.getName() == "Philippe");
      s1.setName("Meunier");
      System.out.println(s1.getName() == "Meunier");

      // Testing the 'if' test in the constructor
      Student s2 = new Student(-9999999, "Unknown");
      System.out.println(s2.getID() == 0);
      Student s3 = new Student(0, "Unknown");
      System.out.println(s3.getID() == 0);
}
```

question 4 (Mandatory)

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:



```
| + 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. In Java you can use **==** to compare characters.

```
public static void testStudent() {
      // Testing the constructor and the getID method
      Student s1 = new Student(9999999, "Philippe");

      System.out.println(s1.getID() == 9999999);

      // Testing the getName and setName methods
      System.out.println(s1.getName() == "Philippe");
      s1.setName("Meunier");
      System.out.println(s1.getName() == "Meunier");

      // Testing the getGrade and setGrade methods
      System.out.println(s1.getGrade() == 'A');
      s1.setGrade('F');
      System.out.println(s1.getGrade() == 'F');

      // Testing the 'if' test in the constructor
      Student s2 = new Student(-9999999, "Unknown");
      System.out.println(s2.getID() == 0);
      Student s3 = new Student(0, "Unknown");
      System.out.println(s3.getID() == 0);
}
```

question 5 (Mandatory)

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.

```
public static void testStudent() {
      // Testing the first constructor and the getID method
      Student s1 = new Student(9999999, "Philippe");
      System.out.println(s1.getID() == 9999999);

      // Testing the getName and setName methods
      System.out.println(s1.getName() == "Philippe");
      s1.setName("Meunier");
      System.out.println(s1.getName() == "Meunier");

      // Testing the getGrade and setGrade methods
      System.out.println(s1.getGrade() == 'A');
      s1.setGrade('F');
      System.out.println(s1.getGrade() == 'F');

      // Testing the 'if' test in the first constructor
      Student s2 = new Student(-9999999, "Unknown");
      System.out.println(s2.getID() == 0);
      Student s3 = new Student(0, "Unknown");
      System.out.println(s3.getID() == 0);

      // Testing the second constructor
      Student s4 = new Student(9999999, "Philippe", 'B');
      System.out.println(s4.getID() == 9999999);
      System.out.println(s4.getName() == "Philippe");
      System.out.println(s4.getGrade() == 'B');

            // Testing the 'if' test in the second constructor
      Student s5 = new Student(-9999999, "Unknown", 'C');
      System.out.println(s5.getID() == 0);
      Student s6 = new Student(0, "Unknown", 'C');
      System.out.println(s6.getID() == 0);
}
```

question 6 (Optional)

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

```
| - ID: int                                          |
| - name: String                                     |
| - grade: char                                      |
| - sleeping: boolean                                |
+----------------------------------------------------+
| + 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                       |
| + isSleeping(): boolean                            |
| + goToSleep(): void                                |
| + wakeUp(): void                                   |
| + testStudent(): void                              |
+----------------------------------------------------+
```

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

```java
public static void testStudent() {
      // Testing the first constructor and the getID method
      Student s1 = new Student(9999999, "Philippe");
      System.out.println(s1.getID() == 9999999);

      // Testing the getName and setName methods
      System.out.println(s1.getName() == "Philippe");
      s1.setName("Meunier");
      System.out.println(s1.getName() == "Meunier");

      // Testing the getGrade and setGrade methods
      System.out.println(s1.getGrade() == 'A');
      s1.setGrade('B');
      System.out.println(s1.getGrade() == 'B');

      // Testing the isSleeping, goToSleep, and wakeUp methods
      // As part of these tests we need to check that the grade
      // correctly changes when the students goes to sleep and
      // when the student wakes up.
      System.out.println(s1.isSleeping() == false);
      s1.goToSleep();
      System.out.println(s1.isSleeping() == true);
      System.out.println(s1.getGrade() == 'C');
      s1.wakeUp();
      System.out.println(s1.isSleeping() == false);
      System.out.println(s1.getGrade() == 'C');

      // The 'if' statement of the goToSleep method has two
      // branches: one for the 'A', 'B', and 'C' cases, and one
      // for the 'D', 'F', and 'other' cases.  We tested the first
      // branch just above, so here we test the other branch
      s1.setGrade('X');
      s1.goToSleep();
      System.out.println(s1.isSleeping() == true);
      System.out.println(s1.getGrade() == 'F');
      s1.wakeUp();
      System.out.println(s1.isSleeping() == false);
      System.out.println(s1.getGrade() == 'F');

      // Testing the 'if' test in the first constructor
      Student s2 = new Student(-9999999, "Unknown");
      System.out.println(s2.getID() == 0);
      Student s3 = new Student(0, "Unknown");
      System.out.println(s3.getID() == 0);

      // Testing the second constructor
      Student s4 = new Student(9999999, "Philippe", 'B');
      System.out.println(s4.getID() == 9999999);
      System.out.println(s4.getName() == "Philippe");
      System.out.println(s4.getGrade() == 'B');

      // Testing the 'if' test in the second constructor
      Student s5 = new Student(-9999999, "Unknown", 'C');
      System.out.println(s5.getID() == 0);
      Student s6 = new Student(0, "Unknown", 'C');
      System.out.println(s6.getID() == 0);
}
```