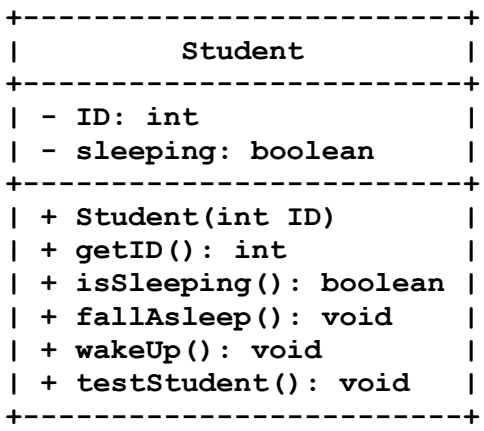
Lab 4 Requirements

Create a new Eclipse workspace named "**Lab4_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 "**Lab4_1234567890.zip**" (replace **1234567890** with your student ID number). Upload the ZIP file on iSpace.

Question 1

Create a class **Student** with the following UML diagram (remember that **+** means public and **–** means private):



ID is the student's ID number. The **sleeping** instance variable indicates whether the student is currently sleeping or not. When a student is created, it is initially awake (not sleeping). The **isSleeping** method indicates as a result whether the student is currently sleeping or not. The **fallAsleep** method makes the student fall asleep (or does nothing if the student is already sleeping). The **wakeUp** method wakes the student up (or does nothing if the student is already awake).

The **testStudent** method is static and is used for testing the **Student** class. Here is the code for this **testStudent** method:

```
public static void testStudent() {
	Student s = new Student(1234567890);

	System.out.println(s.getID() == 1234567890);
	System.out.println(s.isSleeping() == false);
	s.fallAsleep();
	System.out.println(s.isSleeping() == true);
	s.fallAsleep(); // should do nothing because the student is already sleeping
	System.out.println(s.isSleeping() == true);
	s.wakeUp();
	System.out.println(s.isSleeping() == false);
	s.wakeUp(); // should do nothing because the student is already awake
	System.out.println(s.isSleeping() == false);
}
```

And here is the **Start** class to test the **Student** class:

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

Question 2

Add to the **Start** class a new method called **check** that takes a **Student** object as argument and returns a string as a result. This **check** method should return the string "**sweet dreams**" if the student is currently sleeping, and should return the string "**need coffee**" if the student is currently awake.

Should the **check** method be static or not? Why?

Add tests to the **main** method of the **Start** class to test your new **check** method. You can use the **==** operator to compare strings.

Question 3

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

```
+--------------------------+
|         Chicken          |
+--------------------------+
| - weight: double         |
| - sleeping: boolean      |
+--------------------------+
| + Chicken(double weight) |
| + getWeight(): double    |
| + isSleeping(): boolean  |
| + fallAsleep(): void     |
| + wakeUp(): void         |
| + testChicken(): void    |
+--------------------------+
```

The **weight** instance variable indicates the current weight of the chicken. The **sleeping** instance variable indicates whether the chicken is currently sleeping or not. When a chicken is created, it is initially sleeping (not awake). The **isSleeping** method indicates as a result whether the chicken is currently sleeping or not. The **fallAsleep** method makes the chicken fall asleep (or does nothing if the chicken is already sleeping). The **wakeUp** method wakes the chicken up (or does nothing if the chicken is already awake).

The **testChicken** method is static and is used for testing the **Chicken** class. Here is the code for this **testChicken** method:

```
	public static void testChicken() {
		Chicken c = new Chicken(2.3);

		System.out.println(c.getWeight() == 2.3);
		System.out.println(c.isSleeping() == true);
		c.wakeUp();
		System.out.println(c.isSleeping() == false);
		c.wakeUp(); // should do nothing because the chicken is already awake
		System.out.println(c.isSleeping() == false);
		c.fallAsleep();
		System.out.println(c.isSleeping() == true);
```

```
        c.fallAsleep(); // should do nothing because the chicken is already sleeping
        System.out.println(c.isSleeping() == true);
    }
```

Do not forget to modify the **main** method of the **Start** class to test the **Chicken** class too!

Question 4

Add to the **Start** class a new method called **check** that takes a **Chicken** object as argument and returns a string as a result. This **check** method should return the string "**sweet dreams**" if the chicken is currently sleeping, and should return the string "**need coffee**" if the chicken is currently awake.

Should the **check** method be static or not? Why?

Add tests to the **main** method of the **Start** class to test your new check method. You can use the **==** operator to compare strings.