Assignment 1

Create a new Eclipse workspace named "**OOP**" on the desktop of your computer. Create a Java Project named "Assignment1". Download the **assignment1-sample-code-for-student.zip** from iSpace and unzip it. Next, we will import the sample code to you project (right click the src -> Import… -> File System -> locate to the unzipped folder -> import all the questions to your project). Answer all the questions below. At the end, 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. Rename your src.zip to "**Assignment1_1234567890.zip**" (replace **1234567890** with your student ID number). Upload the ZIP file onto AutoLab. For more details, refer to the file ***How to Use Eclipse and submit to Autolab for student.pdf*** on iSpace. Note: you **must** use the sample code, so AutoLab can work correctly.

Here are a few extra instructions:

- 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 should be beautiful to read.

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

question 1

You'll apply your knowledge of classes to build a Java program that utilizes an **Animal** class (Animal). It contains 4 private properties : name (Name of the animal), fastOrslow(fast or slow running animal), nLegs (number of legs it has), topSpeed (max speed it attain), endangered (whether it is an endangered animal).

Write an **`Animal`** class with the following UML specification:

```
+-----------------------------------------+
|                 Animal                  |
+-----------------------------------------+
| - name: String                          |
| - fastOrslow: String                    |
| - nLegs: int                            |
| - topSpeed: int                         |
| - endangered: boolean                   |
+-----------------------------------------+
| + Animal()                              |
| + getname(): String                     |
| + setname(String name): void            |
| + getnLegs(): int                       |
| + setnLegs(int nLegs): void             |
| + gettopSpeed(): int                    |
| + settopSpeed(int topSpeed): void       |
| + getendangered(): boolean              |
| + setendangered(boolean endangered): void |
| + getfastOrslow(): String               |
| + info(): void                          |
| + testAnimal(): void                    |
+-----------------------------------------+
```

where:

- **name** is a private String variable contains the name of the animal.
- **fastOrslow** is a private String variable describing whether it is a fast (**topSpeed** >30 or slow (**topSpeed** =<30)
- **nLegs** is a private int variable describing number of legs the animal has.
- **topSpeed** is a private int variable describing max running speed of the animal.
- **endangered** is a private boolean variable describing whether the animal is endangered or not.
- **Animal** is a public constructor that creates an **Animal** object. E.g. Animal("Elephant",4,25,true) means construct an animal object Elephant, with 4 legs, topSpeed = 25 and is an endangered animal. Also, set the fastOrslow to "fast" if the topSpeed >30, otherwise to "slow".
- **getname** is a public String method that returns the name of the animal.
- **setname** is a public method that sets the name of the animal.
- **getnLegs** is a public int method that returns the number of legs of the animal.
- **setnLegs** is a public method that sets the number of legs of the animal.
- **gettopSpeed** is a public int method that returns the max running speed of the animal.
- **settopSpeed** is a public method that sets the max running speed of the animal. If the topSpeed is >30, set fastOrslow to "fast", otherwise "slow".
- **getfastOrslow** is a public String method that returns fastOrslow status of the animal.
- **getendangered** is a public boolean method that returns whether the animal is endangered or not.
- **setendangered** is a public method that sets whether the animal is endangered or not.
- **info** is a public method that printout the information about the animal in the format. E.g. The Elephant has a top speed of 25 mph, a slow moving animal.
- **testAnimal** is a public static method that tests all the code in your **Animal** class. Test all your methods from the simplest one first to the most complicated one last.

Once you have written the **Animal** class, you can test it by adding the following code in a separate class:

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

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

question 2

You are a software engineer who has to write the software of an air conditioner (AC). The AC has three modes: cool mode, heat mode, and dry mode. The software is designed to be able to turn on / off the AC, set the mode and the temperature.

Write a **AirConditioner** class with the following UML specification:

```
+-----------------------------------------+
|              AirConditioner             |
+-----------------------------------------+
| - powerOn: boolean                      |
| - mode: int                             |
| - temperature: int                      |
+-----------------------------------------+
| + AirConditioner()                      |
```

```
| + turnOn(): void                          |
| + turnOff(): void                         |
| + getPower(): boolean                     |
| + setMode(int mode): void                 |
| + getMode(): int                          |
| + setTemperature(int temp): void          |
| + getTemperature(): int                   |
| + testAirConditioner(): void              |
+-------------------------------------------+
```

where:

- **powerOn** is a private instance variable describing whether the power of the AC is on or not. While true means power on, false means power off.
- **mode** is a private instance variable describing the current mode of the AC. To be specific, the codes for the cool mode, the heat mode and the dry mode are 0, 1, 2, respectively. Any other code is invalid.
- **temperature** is a private instance variable describing the temperature in degree Celsius that is set for the AC.
- **AirConditioner** is a public constructor that creates an **AirConditioner** object. When an AC is created, its power is off, mode is cool, and temperature is 16.
- **turnOn** is a public method that turns the AC on.
- **turnOff** is a public method that turns the AC off.
- **getPower** is a public method that returns the status of the power.
- **setMode** is a public method that sets the mode of the AC to *mode*.
    - If the AC is power off, this function should print "**Please turn on the AC first.**" and return.
    - If the code is invalid, **setMode** should print "**Invalid mode.**" and return.
- **getMode** is a public method that returns the current mode of the AC.
- **setTemperature** is a public method that set the temperature of the AC to *temp*.
    - If the AC is power off, this function should print "**Please turn on the AC first.**" and return.
    - If the AC is power on but is on the dry mode, **setTemperature** should print "**Cannot set the temperature in the dry mode.**" and return.
    - The valid range of temperature is from 16 to 30 degrees. If *temp* is invalid, **setTemperature** should print "**Invalid temperature.**" and return.
- **getTemperature** is a public method that returns the current temperature of the AC.
- **testAirConditioner** is a public static method that tests all the code in your **AirConditioner** class. Test all your methods from the simplest one first to the most complicated one last.

Once you have written the **AirConditioner** class, you can test it by adding the following code in a separate class:

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

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