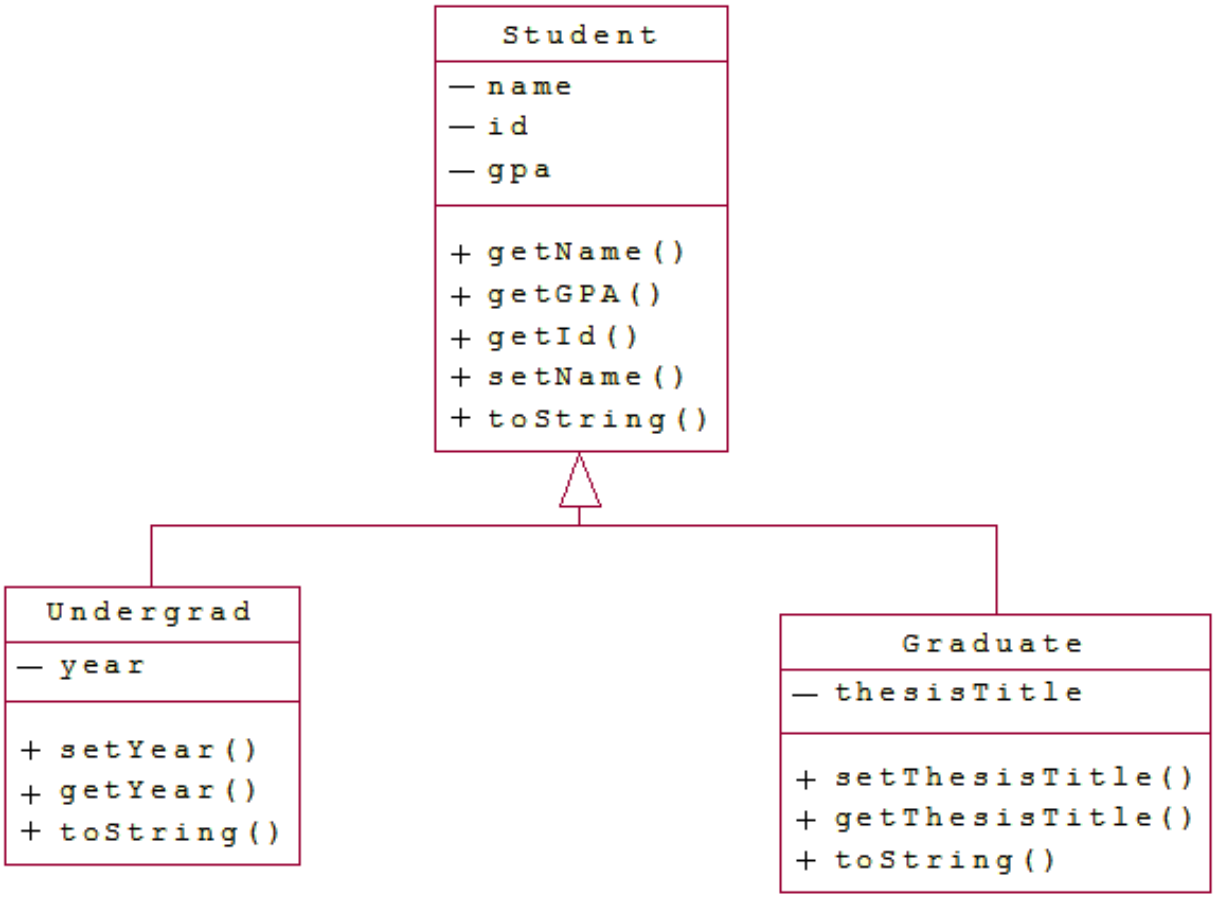
Lab06: Inheritance

In this lab, the following topics will be covered:

1. **Inheritacen (in Java)**
2. **Exercise for practice**

1. Inheritance (in Java)

Inheritance is an important object-oriented concept that allows classes to be reused in order to define similar, but distinct, classes. In this lab we walk through the development of a class hierarchy and a program that makes use of several classes in the hierarchy. We begin by looking at an example of inheritance hierarchy:



The class Student is the parent class. Note that all the variables are private and hence the child classes can only use them through accessor and mutator methods. Also note the use of **overloaded** constructors.

```
/*
 * Student Class
*/
public class Student{
	private String name;
	private int id;
	private double gpa;

	public Student(int id, String name, double gpa) {
		this.id = id;
		this.name = name;
		this.gpa = gpa;
	}

	public Student(int id, double gpa){
		this(id, "", gpa);
	}
	public String getName(){return name;}

	public int getId() {return id;}

	public double getGPA(){return gpa;}

	public void setName(String newName){this.name = newName;	}
	public String toString(){
		return "Student:\nID: "+id+"\nName: "+name+"\nGPA: "+gpa;
	}

 } // end of class Student
```

The class Undergrad **extends** the Student class. Note the **overridden** toString() method

```
/*
 * Undergrad Class
*/
public class Undergrad extends Student
{
	private String year;
	public Undergrad(int id, String name, double gpa, String year)
	{
		super(id, name, gpa);
		this.year = year;
	}
	public String getYear() {return year;}

	public void setYear(String newYear) {this.year = newYear;}

	public String toString() {
		return "Undergraduate "+super.toString()+"\nYear: "+year;
	}
}//end of Cource class
```

The class Graduate **extends** the Student class too. Note the **overridden** toString() method

```
/*
 *  Graduate class
 */
public class Graduate extends Student
{
	private String thesisTitle;
	public Graduate(int id, String name, double gpa, String
                                                        thesisTitle)
	{
		super(id, name, gpa);
		this.thesisTitle = thesisTitle;
	}
	public String getthesisTitle() {
		return thesisTitle;
	}
	public void setThesisTitle(String newthesisTitle) {
		this.thesisTitle = newthesisTitle;
	}
	public String toString() {
		return "Graduate " +super.toString()+"\nThesis:
"+thesisTitle;
	}
}
```

TestStudents is a driver class to test the above classes

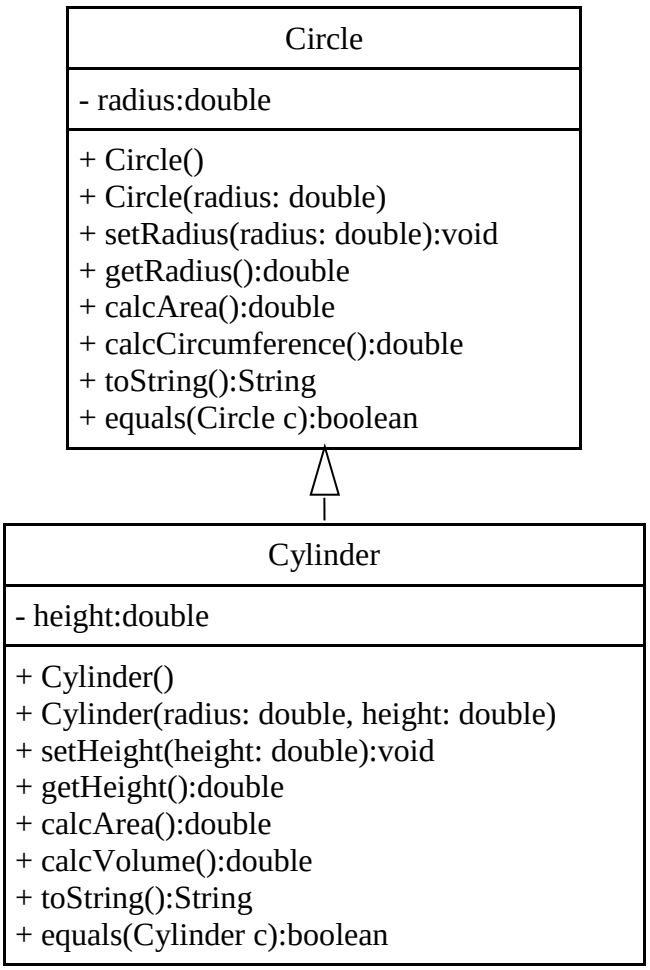
```
/*
 * TestStudents is driver Class
*/
public class TestStudents
{
	public static void main(String[] args)
	{
		Student s1 = new Student(123456, "Tariq", 3.51);
		Student s2 = new Student(234567, 3.22);
		Undergrad u1 = new Undergrad(345678, "Yasser", 2.91,
"Junior");
		Graduate g1 = new Graduate(456789, "Mubin", 3.57,
"Algorithms and Complexity");

		System.out.println(s1);
		System.out.println(s2);
		System.out.println(u1);
		System.out.println(g1);
	}
}
```

Exercises

Exercise 1 *(Circle.java, Cylinder.java Application.java)*

Write the classes below containing the given instance variables and methods, following the inherited hierarchy:



Formulas:

The following formulas may be helpful:

- Area of a circle: πr^2
- Circumference of a circle: $2\pi r$
- Surface area of a cylinder: $2\pi rh + 2\pi r^2$
- Volume of a cylinder: $\pi r^2 h$

Use the following application class to test your code

```
public class Application {
    public static void main(String []args){
        Circle circle1 = new Circle(3);
        Circle circle2 = new Circle(5);
```

```
        System.out.println(circle1);
        System.out.println(circle2);
        if(circle1.equals(circle2))
            System.out.println("Both circles are equal");
        else
            System.out.println("Both circles are different");

        Cylinder cylinder1 = new Cylinder(3,4);
        Cylinder cylinder2 = new Cylinder(4,6);
        Cylinder cylinder3 = new Cylinder(3,4);
        System.out.println(cylinder1);
        System.out.println(cylinder2);
        if(cylinder1.equals(cylinder3))
            System.out.println("Both cylinders are equal");
        else
            System.out.println("Both cylinders are different");
    }
}
```

Exercise 2 *(PurchaseItem.java,WeightedItem.java, CountedItem.java,)*

Consider a superclass PurchaseItem which models customer's purchases. This class has:

- Two private instance variables name (String) and unitPrice (double).
- One constructor to initialize the instance variables.
- A default constructor to initialize name to "no item", and unitPrice to 0. use this()
- A method getPrice that returns the unitPrice.
- Accessor and mutator methods.
- A toString method to return the name of the item followed by @ symbol, then the unitPrice.

Consider two subclasses WeighedItem and CountedItem. WeighedItem has an additional instance variable weight (double) in Kg while CountedItem has an additional variable quantity (int) both private.

- Write an appropriate constructor for each of the classes making use of the constructor of the superclass in defining those of the subclasses.
- Override getPrice method that returns the price of the purchasedItem based on its unit price and weight (WeighedItem), or quantity (CountedItem). Make use of getPrice of the superclass

- Override also toString method for each class making use of the toString method of the superclass in defining those of the subclasses.

toString should return something that can be printed on the receipt.

For example

Banana @ 3.00 1.37 Kg 4.11 PKR (in case of WeighedItem class)

Pens @ 4.5 10 units 45 PKR (in case of CountedItem class)

Class Diagram:

You must draw a class diagram first to start writing your code.

Project

- From last week you have found the relationships between the classes and now find the inheritance in your classes.