IT22049

1] Classes and object:

```
class Student {          // student is class
    String name;         // field
    int rollnumber;
    void study(){        // method
        System.out.println(name + "is studying...");
    }
}
public class Main {
    public static void main (String[] args) {
        Student student1 = new Student();
        student1.name = "Alice"  //object creation
        student1.rollnumber = 101;
        student1.study();
    }
}
```

2] Access modifier

```
class Book {
    private String title; // Private variable
    public void setTitle(String t) { // using this to
        title = t;                   access & safely
    }                                title.
    public String getTitle(){
        return title;
    }
```

```
public class Main {

    public static void main (String[] args) {
        Book myBook = new Book();
        myBook.setTitle ("The Alchemist");
        System.out.println (myBook.getTitle());
    }
}
```

3] Inheritance and Protected Access

```
class Employee {
    protected String role = "Employee";
    void work() {
        System.out.println ("Employee is working");
    }
}
class Manager extends Employee {
    void Manage() {
        System.out.println (role + " is managing the team");
    }
}
public class Main {
    public static void main (String[] args) {
        Manager m = new Manage();
```

```
        m.work();
        m.manage();
    }
}
```

Encapsulation:

```
class Student{
        private ~~String~~ int age;
        public void setage(int age){
            if (age>0) this.age=age;
        }
        public int getAge(){
            return age;
        }
    }
public static void main(String[] args){
        Student student = new Student();
        student.set~~Name~~Age(~~"John"~~20);

    }
}
```

5. Abstract Class:-

```
abstract class Animal {
    abstract void makeSound();
    void sleep() {
        System.out.println("Sleeping...");
    }
}

class Dog extends Animal {
    void makeSound() {
        System.out.println("Bark Bark");
    }
}

public class Main {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.makeSound();
        d.sleep();
    }
}
```

6. Interface

```
interface Animal {
    void sound();
}
class Dog implements Animal {
    public void sound() {
        System.out.println("woof");
    }
}
public class main {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.sound();
    }
}
```