

#11. WAP to implement the concept of method overloading and method overriding. And Abstract Class

```
class ani{
    public void cat(){
        System.out.println("She is a cat");
    }
    public void cat(String s){
        System.out.println("She is a " + s);
    }
}
class Ocean{
    public void fish() {
        System.out.println("This is tasty");
    }
}
class Aqua{
    public void fish(){
        System.out.println("This is living being");
    }
}
abstract class kurama {
    abstract public void kurama();
    void naruto(){
        System.out.println("I have Rassingaan");
    }
}

public class lab1 {
    public static void main(String[] args){
        ani a = new ani();
        a.cat();
        a.cat("Kitty");
        Ocean b = new Ocean();
        b.fish();
        Aqua z = new Aqua();
        z.fish();
        kurama k = new kurama() {
            @Override
            public void kurama() {
                System.out.println("I have levitating powers");
            }
        };
        k.kurama();
        k.naruto();
    }
}
```

12. WAP to Create a class Employee having members as follows:

```
private int empNo
```

```
private String empName
```

```
private int empBasic
```

Parameterized constructor to initialize members.

Getter methods for all instance variables

```
class emp{
    private int empNo;
    private String empName;
    private int empBasic;
    emp(int s,String name,int sal){
        empNo = s;
        empName = name;
        empBasic = sal;
    };
    void getter(){
        System.out.println(empName);
        System.out.println(empBasic);
        System.out.println(empNo);
    }
}
```

```
public class Employee {
    public static void main(String[] args){
        emp p = new emp(0021,"hritik",25000);
        p.getter();
    }
}
```

14. WAP to create a class named Shape and create three subclasses Circle, Triangle and Square, each class has two-member functions named draw () and erase (). Implement this concept using polymorphism.

// Base class

```
class Shape {
    void draw() {
        System.out.println("Drawing a Shape");
    }

    void erase() {
        System.out.println("Erasing a Shape");
    }
}
```

```
}
```

```
// Subclass 1
```

```
class Circle extends Shape {
```

```
    @Override
```

```
    void draw() {
```

```
        System.out.println("Drawing a Circle");
```

```
    }
```

```
    @Override
```

```
    void erase() {
```

```
        System.out.println("Erasing a Circle");
```

```
    }
```

```
}
```

```
// Subclass 2
```

```
class Triangle extends Shape {
```

```
    @Override
```

```
    void draw() {
```

```
        System.out.println("Drawing a Triangle");
```

```
    }
```

```
    @Override
```

```
    void erase() {
```

```
        System.out.println("Erasing a Triangle");
```

```
    }
```

```
}
```

```
// Subclass 3
```

```
class Square extends Shape {
```

```
    @Override
```

```

void draw() {
    System.out.println("Drawing a Square");
}

@Override
void erase() {
    System.out.println("Erasing a Square");
}
}

```

```

// Main class
public class Main {
    public static void main(String[] args) {
        Shape s;

        s = new Circle(); // Polymorphism
        s.draw();
        s.erase();

        s = new Triangle(); // Polymorphism
        s.draw();
        s.erase();

        s = new Square(); // Polymorphism
        s.draw();
        s.erase();
    }
}

```

13. Create a class WriteEmployee having a main method. Ask the user to enter the details of an employee and store them in an Employee object. After storing, retrieve the details from the object itself and display those details.

```

import java.util.Scanner;

```

```
// Employee class
class Employee {
    private int id;
    private String name;
    private String department;
    private double salary;

    // Constructor
    public Employee(int id, String name, String department, double salary) {
        this.id = id;
        this.name = name;
        this.department = department;
        this.salary = salary;
    }

    // Getters
    public int getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public String getDepartment() {
        return department;
    }

    public double getSalary() {
        return salary;
    }
}
```

```
}  
}
```

```
// Main class
```

```
public class WriteEmployee {
```

```
    public static void main(String[] args) {
```

```
        Scanner sc = new Scanner(System.in);
```

```
        // Taking input
```

```
        System.out.print("Enter Employee ID: ");
```

```
        int id = sc.nextInt();
```

```
        sc.nextLine(); // consume newline
```

```
        System.out.print("Enter Employee Name: ");
```

```
        String name = sc.nextLine();
```

```
        System.out.print("Enter Employee Department: ");
```

```
        String department = sc.nextLine();
```

```
        System.out.print("Enter Employee Salary: ");
```

```
        double salary = sc.nextDouble();
```

```
        // Creating Employee object
```

```
        Employee emp = new Employee(id, name, department, salary);
```

```
        // Retrieving and displaying details
```

```
        System.out.println("\nEmployee Details:");
```

```
        System.out.println("ID      : " + emp.getId());
```

```
        System.out.println("Name    : " + emp.getName());
```

```
        System.out.println("Department: " + emp.getDepartment());
```

```
        System.out.println("Salary   : " + emp.getSalary());
```

```
        sc.close();  
    }  
}
```