# **object oriented programming and design**

**Class, Object, Methods, Constructors, Static Keyword**

1. **Write a java program to print student details using class and objects**

class Student {

String name;

int age;

void displayInfo() {

System.out.println("Name: " + name);

System.out.println("Age: " + age);

}

}

public class Main1 {

public static void main(String[] args) {

Student s1 = new Student();

s1.name = "Ramya";

s1.age = 20;

s1.displayInfo();

}

}

1. **Write a java program to calculate the area of the rectangle using class and objects**

class Rectangle {

int length;

int width;

void calculateArea() {

int area = length \* width;

System.out.println("Area of Rectangle: " + area);

}

}

public class Main2 {

public static void main(String[] args) {

Rectangle r = new Rectangle();

r.length = 5;

r.width = 4;

r.calculateArea();

}

}

1. **Write a java program to illustrate Simple Bank Account using class and object**

class BankAccount {

String accountHolder;

double balance;

void deposit(double amount) {

balance += amount;

System.out.println(amount + " deposited. New balance: " + balance);

}

void displayBalance() {

System.out.println("Account Holder: " + accountHolder);

System.out.println("Balance: " + balance);

}

}

public class Main3 {

public static void main(String[] args) {

BankAccount acc = new BankAccount();

acc.accountHolder = "Ravi";

acc.balance = 1000;

acc.deposit(500);

acc.displayBalance();

}

}

1. **Write a Java program to overload a method add() that adds: two integers and two double values.**

class Calculator {

void add(int a, int b) {

System.out.println("Sum of integers: " + (a + b));

}

void add(double a, double b) {

System.out.println("Sum of doubles: " + (a + b));

}

public static void main(String[] args) {

Calculator calc = new Calculator();

calc.add(5, 10); // calls int version

calc.add(2.5, 3.7); // calls double version

}

}

1. **Write a Java program that calculates the area of different shapes (circle, square, rectangle, and triangle) using method overloading.**

class AreaCalculator {

void area(double radius) {

double result = 3.1416 \* radius \* radius;

System.out.println("Area of Circle: " + result);

}

void area(int side) {

int result = side \* side;

System.out.println("Area of Square: " + result);

}

void area(int length, int breadth) {

int result = length \* breadth;

System.out.println("Area of Rectangle: " + result);

}

void area(double base, double height) {

double result = 0.5 \* base \* height;

System.out.println("Area of Triangle: " + result);

}

public static void main(String[] args) {

AreaCalculator ac = new AreaCalculator();

ac.area(5.0); // Circle

ac.area(4); // Square

ac.area(5, 10); // Rectangle

ac.area(6.0, 3.0); // Triangle

}

}

1. **Write a Java program to demonstrate non-parameterized and parameterized constructors.**

class Student {

String name;

int age;

Student() {

name = "Ramya";

age = 0;

System.out.println("Non-Parameterized Constructor Called");

}

Student(String n, int a) {

name = n;

age = a;

System.out.println("Parameterized Constructor Called");

}

void display() {

System.out.println("Name: " + name + ", Age: " + age);

}

public static void main(String[] args) {

Student s1 = new Student();

Student s2 = new Student("Ram", 20);

s1.display();

s2.display();

}

}

1. **Write a Java program to demonstrate the use of a copy constructor.**

class Book {

String title;

int pages;

Book(String t, int p) {

title = t;

pages = p;

}

Book(Book b) {

title = b.title;

pages = b.pages;

}

void display() {

System.out.println("Title: " + title + ", Pages: " + pages);

}

public static void main(String[] args) {

Book book1 = new Book("Java Basics", 250);

Book book2 = new Book(book1);

System.out.println("Original Book:");

book1.display();

System.out.println("Copied Book:");

book2.display();

}

}

1. **Write a Java program to illustrate constructor overloading using a Box class.**

class Box {

double length, width, height;

Box() {

length = 1;

width = 1;

height = 1;

}

Box(double side) {

length = width = height = side;

}

Box(double l, double w, double h) {

length = l;

width = w;

height = h;

}

void displayVolume() {

double volume = length \* width \* height;

System.out.println("Volume: " + volume);

}

public static void main(String[] args) {

Box box1 = new Box(); // default constructor

Box box2 = new Box(5); // cube

Box box3 = new Box(3, 4, 5); // rectangular box

box1.displayVolume();

box2.displayVolume();

box3.displayVolume();

}

}

1. **Static Data Member Example**

class Employee {

int empId;

String name;

static String company = "TechCorp";

Employee(int id, String n) {

empId = id;

name = n;

}

void display() {

System.out.println(empId + " " + name + " " + company);

}

public static void main(String[] args) {

Employee e1 = new Employee(1, "Alice");

Employee e2 = new Employee(2, "Bob");

e1.display();

e2.display();

}

}

1. **Static Method Example**

class MathUtility {

static int square(int x) {

return x \* x;

}

public static void main(String[] args) {

int result = MathUtility.square(5);

System.out.println("Square of 5 is: " + result);

}

}

1. **Static Block Example**

class Hello {

static {

System.out.println("Static block executed before main method.");

}

public static void main(String[] args) {

System.out.println("Main method executed.");

}

}

1. **Write a Java program to demonstrate the use of a static data member, static method, and static block using a Student class.**

class Student {

int rollNo;

String name;

static String collegeName;

static {

collegeName = "ABC College";

System.out.println("Static block executed: College name set");

}

Student(int r, String n) {

rollNo = r;

name = n;

}

static void changeCollege(String newCollege) {

collegeName = newCollege;

}

void display() {

System.out.println(rollNo + " " + name + " " + collegeName);

}

public static void main(String[] args) {

Student s1 = new Student(101, "Ramya");

Student s2 = new Student(102, "Priya");

s1.display();

s2.display();

Student.changeCollege("XYZ Institute");

s1.display();

s2.display();

}

}