MEGHNAD SAHA INSTITUTE OF TECHNOLOGY

*Techno Complex, Madurdaha,Beside NRI Complex, Post-Uchhepota, Kolkata 700 150*

LABORATORY NOTE BOOK

MAKAUT EVEN SEMESTER 2025



[MASTERS OF COMPUTER APPLICATION]

[OBJECT ORIENTED PROGRAMMING LAB USING JAVA (MCAN-293)]

[RUPAK SARKAR]

ROLL NO: 14271024036 REGN. NO.: 241420510045

STREAM: MCA SEMESTER: II (2ND)

YEAR: 1ST YearSESSION: 2024-2026



MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY



MEGHNAD SAHA INSTITUTE OF TECHNOLOGY

*Techno Complex,. Madurdaha,Beside NRI Complex, Post-Uchhepota, Kolkata 700 150*

“LIST OF ASSIGNMENT/EXPERIMENT SUBMISSION DETAILS”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SL.**  **NO.** | **ASSIGNMENT / EXPERIMENT NAME** | **DATE OF EXPERIMENT** | **DATE OF SUBMISION** | **CHECKED BY** | **REMARKS (ANY DEVIATION REGARDING SUBMISSION DATES, CONTENT, FORMAT, ETC)** |
| 1. | Create class Box and define all constructors. | 03/03/2025 | 10/03/2025 |  |  |
| 2. | Create a class employee with age and sal and calculate max sal. | 03/03/2025 | 10/03/2025 |  |  |
| 3. | Create a class Complex and add two Complex numbers using ‘this’ keyword. | 03/03/2025 | 10/03/2025 |  |  |
| 4. | Create a class Stack and implement push and pop. | 03/03/2025 | 10/03/2025 |  |  |
| 5. | Create a class Time and add two Time objects. | 03/03/2025 | 10/03/2025 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

|  |
| --- |
| OBSERVATIONS / COMMENTS ON THE OVERALL PERFORMANCE: |

Signature in full with date Signature in full with date

**Faculty / Technical Assistant Lab Examiner**

**Q.1. Write a class Box and describe all the constructors of the class, include a volume method that will return the volume of object Box (Overloading Constructor).**

**Code:**

class Box

{

    double height;

    double width;

    double depth;

    Box()

    {

        height = 0.0;

        width = 0.0;

        depth = 0.0;

    }

    Box (double h, double w, double d)

    {

        height = h;

        width = w;

        depth = d;

    }

    Box (double h, double w)

    {

        height = h;

        width = w;

        depth = 10;

    }

    Box (Box b)

    {

        height = b.height;

        width = b.width;

        depth = b.depth;

    }

    double vol()

    {

        return height\*width\*depth;

    }

}

class BoxDemo1

{

    public static void main(String args[])

    {

        Box b1 = new Box();

        Box b2 = new Box(1,2,3);

        Box b3 = new Box(4,5);

        Box b4 = new Box(b3);

        System.out.println("Volume of b1: "+b1.vol());

        System.out.println("Volume of b2: "+b2.vol());

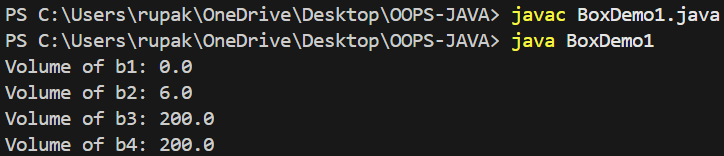
        System.out.println("Volume of b3: "+b3.vol());

        System.out.println("Volume of b4: "+b4.vol());

    }

}

**Output:**



**Q.2. Create an Employee class age and sal using constructor, take input and find the highest salaried employee.**

**Code:**

import java.util.Scanner;

class Emp {

    int age;

    float sal;

    Emp() {

        age = 0;

        sal = 0.0f;

    }

    Emp(int a, float s) {

        age = a;

        sal = s;

    }

}

public class EmpDemo {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the number of employees: ");

        int n = sc.nextInt();

        Emp[] employees = new Emp[n];

        for (int i = 0; i < n; i++) {

            System.out.println("Enter details for employee " + (i + 1) + ":");

            System.out.print("Age: ");

            int age = sc.nextInt();

            System.out.print("Salary: ");

            float sal = sc.nextFloat();

            employees[i] = new Emp(age, sal);

        }

        Emp highestPaid = employees[0];

        for (int i = 1; i < n; i++) {

            if (employees[i].sal > highestPaid.sal) {

                highestPaid = employees[i];

            }

        }

        System.out.println("\nHighest Salaried Employee:");

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

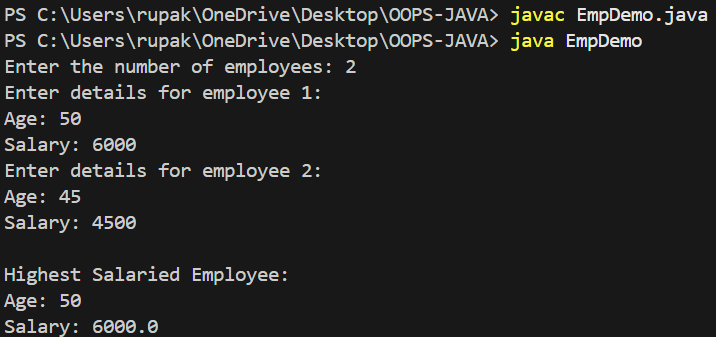
        System.out.println("Salary: " + highestPaid.sal);

        sc.close();

    }

}

**Output:**



**Q.3. Create a class Complex and add two complex objects using ‘this’ keyword.**

**Code:**

class Complex

{

    double real, imaginary;

    Complex()

    {

        real = 0.0;

        imaginary = 0.0;

    }

    Complex(double r, double i)

    {

        real = r;

        imaginary = i;

    }

    Complex Add(Complex x, Complex y)

    {

       this.real = x.real + y.real;

       this.imaginary = x.imaginary + y.imaginary;

       return this;

    }

    void display()

    {

        System.out.println(real+" +i"+imaginary);

    }

}

class AddComplex1

{

    public static void main(String args[])

    {

        Complex C1 = new Complex(2,3);

        Complex C2 = new Complex(5,6);

        Complex C3 = new Complex();

        C3=C3.Add(C1,C2);

        C1.display();

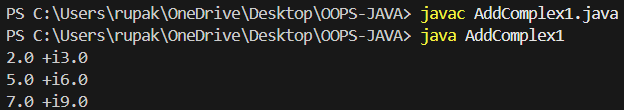
        C2.display();

        C3.display();

    }

}

**Output:**



**Q.4. Create a class Stack and implement push and pop operations.**

**Code:**

class Stack

{

    int top;

    int stck[] = new int[10];

    Stack()

    {

        top = -1;

    }

    void push(int item)

    {

        if(top == (stck.length-1))

        {

            System.out.println("Stack is Full!");

        }

        else

        {

            stck[++top]=item;

        }

    }

    int pop()

    {

        if(top==-1)

        {

            System.out.println("Stack is Empty!");

            return 0;

        }

        else

        {

            return stck[top--];

        }

    }

}

class StackDemo

{

    public static void main(String args[])

    {

        Stack s1 = new Stack();

        int k=10;

        for (int i=0;i<10;i++)

        {

            s1.push(k);

            k++;

        }

        for (int i=0;i<10;i++)

        {

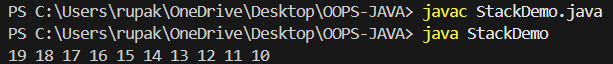
            System.out.print(s1.pop()+" ");

        }

    }

}

**Output:**



**Q.5. Create a class Time, Add two Time objects using method add() that will return an object of Time class after adding them.**

**Code:**

class Time

{

    int hr, min, sec;

    Time()

    {

        hr = 0;

        min = 0;

        sec = 0;

    }

    Time(int h, int m, int s)

    {

        hr = h;

        min = m;

        sec = s;

    }

    Time add(Time T1, Time T2)

    {

        this.sec = T1.sec + T2.sec;

        this.min = T1.min + T2.min + (this.sec / 60);

        this.sec %= 60;

        this.hr = T1.hr + T2.hr + (this.min / 60);

        this.min %= 60;

        return this;

    }

    void display()

    {

        System.out.println(hr + " hr " + min + " min " + sec + " sec");

    }

}

public class TimeDemo

{

    public static void main(String args[])

    {

        Time T1 = new Time(2, 45, 50);

        Time T2 = new Time(1, 20, 30);

        Time T3 = new Time();

        T3 = T3.add(T1, T2);

        System.out.print("Sum of T1 and T2: ");

        T3.display();

    }

}

**Output:**

