1. **Write a java program:**
    a. **To calculate and display the area of a rectangle.**
    b. **To multiply two arrays and display the result**
    c. **To sort the elements in ascending and descending order using bubble sort algorithm.**

1a. Java program to calculate and display the area of a rectangle:

```
import java.util.Scanner;

public class RectangleArea {
  public static void
  main(String[] args) {
    Scanner input = new Scanner(System.in);
    System.out.print("Enter the length of the
    rectangle: "); double length =
    input.nextDouble();
    System.out.print("Enter the width of the
    rectangle: "); double width =
    input.nextDouble();

    double area = length * width;
    System.out.println("The area of the rectangle is: " + area);
  }
}
```

1b. Java program to multiply two arrays and display the result:

```
public class ArrayMultiplication
  { public static void
  main(String[] args) {
    int[] array1 = {1, 2, 3, 4, 5};
    int[] array2 = {6, 7, 8, 9,
    10}; int[] result = new
    int[array1.length];
```

```
    for (int i = 0; i < array1.length;
      i++) { result[i] = array1[i] *
      array2[i];
    }

    System.out.print("Result array after
    multiplication: "); for (int num : result) {
      System.out.print(num + " ");
    }
  }
}
```

1c. Java program to sort elements in ascending and descending order using the bubble sort algorithm:

```
public class BubbleSort {
  public static void
  main(String[] args) { int[] arr
  = {5, 3, 1, 4, 2}; int n =
  arr.length;

    // Ascending order for
    (int i = 0; i < n - 1; i++) {
    for (int j = 0; j < n - i - 1;
    j++) { if (arr[j] > arr[j + 1])
    { int temp = arr[j]; arr[j] =
    arr[j + 1]; arr[j + 1] =
    temp;
        }
      }
    }

    System.out.print("Ascending
    Order: "); for (int num : arr) {
      System.out.print(num + " ");
```

```
        }

        // Descending order for
        (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1;
        j++) { if (arr[j] < arr[j + 1])
        { int temp = arr[j]; arr[j] =
        arr[j + 1]; arr[j + 1] =
        temp;
              }
           }
        }

        System.out.print("\nDescending
        Order: "); for (int num : arr) {
           System.out.print(num + " ");
        }
     }
}
```

OUTPUT

1A

Enter the length of the rectangle: 5
Enter the width of the rectangle: 3
The area of the rectangle is: 15.0

1B

Result array after multiplication: 6 14 24 36 50

1C

Ascending Order: 1 2 3 4 5
Descending Order: 5 4 3 2 1

2.Write a java program to create an Employee database which stores following information about the Employee: Name, Emp_id, department, age, and designation. Perform the following operations:

a. **Read and display the details of at least five Employees.**
b. **Calculate and display the sum of salary of all the employees of "sales" department**
c. **.Retrieve the details of "highest paid manager" in the purchase department**

```
import java.util.ArrayList; import java.util.Scanner;

class
   Employee {
   String name;
   int empId;
   String
   department;
   int age;
   String designation;
   double salary;

  public Employee(String name, int empId, String department, int age,
                                                        String designation,
double salary) { this.name
      = name; this.empId =
      empId;
      this.department =
      department; this.age =
      age; this.designation =
      designation;
      this.salary = salary;
   }
}

public class
   EmployeeDatabase { public
```

```
  static void main(String[]
  args) {
    ArrayList<Employee> employeeList = new ArrayList<Employee>();
    Scanner input = new Scanner(System.in);

    // Read and add details of
    employees for (int i = 1; i <= 5;
    i++) {
      System.out.println("Enter details for Employee " + i + ":");
      System.out.print("Name: ");
      String name =
      input.nextLine();
      System.out.print("Emp_
      ID: "); int empId =
      input.nextInt();
      input.nextLine(); // Consume the newline character
      System.out.print("Department: ");
      String department =
      input.nextLine();
      System.out.print("Age: ");
      int age = input.nextInt();
      input.nextLine(); // Consume the newline character
      System.out.print("Designation: ");
      String designation = input.nextLine();
      System.out.print("Salary: "); double
      salary = input.nextDouble();
      input.nextLine(); // Consume the
      newline character

      employeeList.add(new Employee(name, empId, department, age,
      designation,
salary));
    }

    System.out.println("Details of
    Employees:"); for (Employee emp :
    employeeList) {
```

```
        System.out.println("Name: " + emp.name + ", Emp_ID: " +
        emp.empId + ",
Department: " + emp.department + ", Age: " + emp.age + ",
Designation: " + emp.designation + ", Salary: " + emp.salary);
    }

    // Calculate and display the sum of salaries of all employees in the
    "Sales" department double salesDepartmentSalarySum = 0.0; for
    (Employee emp : employeeList) { if
    (emp.department.equals("Sales")) {
        salesDepartmentSalarySum += emp.salary;
      }
    }
    System.out.println("Total Salary of Sales Department: " +
    salesDepartmentSalarySum);

    // Retrieve details of the highest-paid manager in the
    "Purchase" department Employee highestPaidManager =
    null; double maxSalary = 0.0;

    for (Employee emp : employeeList) { if
      (emp.department.equals("Purchase") &&
      emp.designation.equals("Manager")) {
        if (emp.salary > maxSalary) {
          maxSalary = emp.salary;
          highestPaidManager =
          emp;
        }
      }
    }

    if (highestPaidManager != null) {
      System.out.println("Details of the Highest-Paid Manager in
      Purchase Department:");
      System.out.println("Name: " + highestPaidManager.name + ",
Emp_ID: " + highestPaidManager.empId + ", Department: " +
highestPaidManager.department + ", Age: " + highestPaidManager.age
+ ", Designation: " + highestPaidManager.designation + ", Salary:
```

```
" + highestPaidManager.salary);
      } else {
         System.out.println("No Manager found in Purchase
         Department.");
      }
   }
}
```

Output

```
Enter details for Employee 1:
Name: John Doe
Emp_ID: 101
Department: Sales
Age: 30
Designation: Salesperson
Salary: 50000

Enter details for Employee 2:
Name: Jane Smith
Emp_ID: 102
Department: IT
Age: 35
Designation: Developer
Salary: 60000

Enter details for Employee 3:
Name: Bob Johnson
Emp_ID: 103
Department: Purchase
Age: 40
Designation: Manager
Salary: 75000

Enter details for Employee 4:
Name: Alice Williams
Emp_ID: 104
Department: Sales
Age: 28
```

Designation: Salesperson
Salary: 55000

Enter details for Employee 5:
Name: Mike Brown
Emp_ID: 105
Department: Purchase
Age: 45
Designation: Manager
Salary: 80000

Details of Employees:
Name: John Doe, Emp_ID: 101, Department: Sales, Age: 30, Designation: Salesperson, Salary: 50000
Name: Jane Smith, Emp_ID: 102, Department: IT, Age: 35, Designation: Developer, Salary: 60000
Name: Bob Johnson, Emp_ID: 103, Department: Purchase, Age: 40, Designation: Manager, Salary: 75000
Name: Alice Williams, Emp_ID: 104, Department: Sales, Age: 28, Designation: Salesperson, Salary: 55000
Name: Mike Brown, Emp_ID: 105, Department: Purchase, Age: 45, Designation: Manager, Salary: 80000

Total Salary of Sales Department: 105000.0

Details of the Highest-Paid Manager in Purchase Department:
Name: Mike Brown, Emp_ID: 105, Department: Purchase, Age: 45, Designation: Manager, Salary: 80000

3.Write a Java program using encapsulation and constructors to create a class to represent a complex number and perform the following operations:

a. **Addition of two complex numbers**
b. **Subtraction of two complex numbers**
c. **Compare two complex numbers.**

```
import java.util.Scanner;

public class Complex {
   static class ComplexNumber {
      int real, image;

      public ComplexNumber(int r, int i) {
         this.real = r;
         this.image = i;
      }

      public static ComplexNumber add(ComplexNumber n1,
ComplexNumber n2) {
         return new ComplexNumber(n1.real + n2.real, n1.image +
n2.image);
      }

      public static ComplexNumber sub(ComplexNumber n1,
ComplexNumber n2) {
         return new ComplexNumber(n1.real - n2.real, n1.image -
n2.image);
      }

      public static ComplexNumber mult(ComplexNumber n1,
ComplexNumber n2) {
         return new ComplexNumber(n1.real * n2.real - n1.image *
n2.image,
               n1.real * n2.image + n1.image * n2.real);
```

```
    }

    public static boolean compare(ComplexNumber n1,
ComplexNumber n2) {
        return n1.real == n2.real && n1.image == n2.image;
    }
  }

  public static void main(String[] args) {
    int nr1, ni1, nr2, ni2;
    Scanner input = new Scanner(System.in);

    System.out.println("Enter the real number of complex 1");
    nr1 = input.nextInt();
    System.out.println("Enter the imaginary of complex 1");
    ni1 = input.nextInt();
    System.out.println("Enter the real number of complex 2");
    nr2 = input.nextInt();
    System.out.println("Enter the imaginary of complex 2");
    ni2 = input.nextInt();

    ComplexNumber c1 = new ComplexNumber(nr1, ni1);
    ComplexNumber c2 = new ComplexNumber(nr2, ni2);

    System.out.println("Choose operation: 1 - Add, 2 - Subtract, 3 -
Multiply, 4 - Compare");
    int choice = input.nextInt();

    switch (choice) {
      case 1:
        ComplexNumber addition = ComplexNumber.add(c1, c2);
        System.out.println("Addition: " + addition.real + " + " +
addition.image + "i");
        break;
      case 2:
        ComplexNumber subtraction = ComplexNumber.sub(c1, c2);
```

```
            System.out.println("Subtraction: " + subtraction.real + " - " +
subtraction.image + "i");
            break;
        case 3:
            ComplexNumber multiplication = ComplexNumber.mult(c1,
c2);
            System.out.println("Multiplication: " + multiplication.real + " * "
+ multiplication.image + "i");
            break;
        case 4:
            boolean comparison = ComplexNumber.compare(c1, c2);
            System.out.println("Comparison: " + (comparison ? "Equal" :
"Not Equal"));
            break;
        default:
            System.out.println("Invalid choice");
      }
   }
}
```

OUTPUT:-

```
Enter the real number of complex 1: 2
Enter the imaginary of complex 1: 3
Enter the real number of complex 2: 1
Enter the imaginary of complex 2: 2
Choose operation: 1 - Add, 2 - Subtract, 3 - Multiply, 4 - Compare: 3
Multiplication: -4 + 7i
```

4**. Write a java program to define a base class called person which stores information about a person such as name, age, gender. Derive two new classes employee and student and extend the base class methods in derived class to read and display the details that are specific to student and employee. Display the details of at least five students and five employees.**

```
import java.util.Scanner;

class Person {
    String name;
    int age;
    char gender;

    void readDetails() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter name: ");
        name = scanner.nextLine();
        System.out.print("Enter age: ");
        age = scanner.nextInt();
        System.out.print("Enter gender (M/F): ");
        gender = scanner.next().charAt(0);
    }

    void displayDetails() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Gender: " + gender);
    }
}

class Employee extends Person {
    String employeeId;

    @Override
    void readDetails() {
```

```
        super.readDetails();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter employee ID: ");
        employeeId = scanner.nextLine();
    }

    @Override
    void displayDetails() {
        super.displayDetails();
        System.out.println("Employee ID: " + employeeId);
    }
}

class Student extends Person {
    String studentId;

    @Override
    void readDetails() {
        super.readDetails();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter student ID: ");
        studentId = scanner.nextLine();
    }

    @Override
    void displayDetails() {
        super.displayDetails();
        System.out.println("Student ID: " + studentId);
    }
}

public class Main {
    public static void main(String[] args) {
        // Display details of 5 employees
        System.out.println("Details of Employees:");
        for (int i = 0; i < 5; i++) {
            System.out.println("\nEmployee " + (i + 1) + ":");
```

```
            Employee employee = new Employee();
            employee.readDetails();
            employee.displayDetails();
        }

        // Display details of 5 students
        System.out.println("\nDetails of Students:");
        for (int i = 0; i < 5; i++) {
            System.out.println("\nStudent " + (i + 1) + ":");
            Student student = new Student();
            student.readDetails();
            student.displayDetails();
        }
    }
}
```

Output:-

```
Details of Employees:

Employee 1:
Enter name: John Doe
Enter age: 30
Enter gender (M/F): M
Enter employee ID: E001

Employee 2:
Enter name: Jane Smith
Enter age: 28
Enter gender (M/F): F
Enter employee ID: E002

Employee 3:
Enter name: Alex Johnson
Enter age: 35
Enter gender (M/F): M
Enter employee ID: E003

Employee 4:
```

Enter name: Emily White
Enter age: 25
Enter gender (M/F): F
Enter employee ID: E004

Employee 5:
Enter name: Michael Brown
Enter age: 40
Enter gender (M/F): M
Enter employee ID: E005

Details of Students:

Student 1:
Enter name: Alice Johnson
Enter age: 20
Enter gender (M/F): F
Enter student ID: S001

Student 2:
Enter name: Bob Anderson
Enter age: 22
Enter gender (M/F): M
Enter student ID: S002

Student 3:
Enter name: Sarah Taylor
Enter age: 21
Enter gender (M/F): F
Enter student ID: S003

Student 4:
Enter name: Kevin Lee
Enter age: 23
Enter gender (M/F): M
Enter student ID: S004

Student 5:
Enter name: Emma Davis
Enter age: 19
Enter gender (M/F): F
Enter student ID: S005

5. Write a java program using compile time polymorphism (method overloading) to compare two strings. The program should implement two different versions of strcmp the first version of usrstrcmp () that compares two string the second version should compare only specified number of characters from first string with second string and display the results.

```
import java.util.Scanner;

public class StringComparison {
    // Version 1: Full string comparison
    static boolean strcmp(String str1, String str2) {
        return str1.equals(str2);
    }

    // Version 2: Comparison with specified number of characters
    static boolean strcmp(String str1, String str2, int numChars) {
        return str1.substring(0, numChars).equals(str2.substring(0,
numChars));
    }

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.println("Enter the first string:");
        String str1 = input.nextLine();

        System.out.println("Enter the second string:");
```

```
        String str2 = input.nextLine();

        // Version 1: Full string comparison
        boolean result1 = strcmp(str1, str2);
        System.out.println("Version 1 (Full string comparison): " + result1);

        // Version 2: Comparison with specified number of characters (let's
use 3 for example)
        System.out.println("Enter the number of characters to compare:");
        int numChars = input.nextInt();

        boolean result2 = strcmp(str1, str2, numChars);
        System.out.println("Version 2 (Comparison with " + numChars + "
characters): " + result2);
    }
}
```

OUTPUT

```
Enter the first string:
Hello World
Enter the second string:
Hello World
Version 1 (Full string comparison): true
Enter the number of characters to compare:
5
Version 2 (Comparison with 5 characters): true
```

6.Write a JAVA program to define a base class bank, which holds various details of customers such as name, account number, balance and member functions to read, display and an abstract method to calculate rate of interest earned by all the account holders. Derive three classes namely City-Bank, SBI-bank, Canara-bank from this base class, which are offering different rate of interests. Extend the calculate method of base class with in these derived classes to calculate and display the interest earned by all the account holders of these banks.

```
import java.util.Scanner;

abstract class Bank {
   String name;
   long accountNumber;
   double balance;

   public Bank(String name, long accountNumber, double balance) {
      this.name = name;
      this.accountNumber = accountNumber;
      this.balance = balance;
   }

   abstract double calculateInterest();

   void displayDetails() {
      System.out.println("Name: " + name);
      System.out.println("Account Number: " + accountNumber);
      System.out.println("Balance: $" + balance);
   }
}

class CityBank extends Bank {
   final double INTEREST_RATE = 0.05;

   public CityBank(String name, long accountNumber, double balance) {
      super(name, accountNumber, balance);
```

```
    }

    @Override
    double calculateInterest() {
        return balance * INTEREST_RATE;
    }
}

class SBIBank extends Bank {
    final double INTEREST_RATE = 0.07;

    public SBIBank(String name, long accountNumber, double balance) {
        super(name, accountNumber, balance);
    }

    @Override
    double calculateInterest() {
        return balance * INTEREST_RATE;
    }
}

class CanaraBank extends Bank {
    final double INTEREST_RATE = 0.06;

    public CanaraBank(String name, long accountNumber, double balance) {
        super(name, accountNumber, balance);
    }

    @Override
    double calculateInterest() {
        return balance * INTEREST_RATE;
    }
}

public class BankInterestProgram {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
```

```
        CityBank cityBank = new CityBank("City Bank", 123456, 5000);
        SBIBank sbiBank = new SBIBank("SBI Bank", 789012, 8000);
        CanaraBank canaraBank = new CanaraBank("Canara Bank", 345678, 6000);

        System.out.println("City Bank Details:");
        cityBank.displayDetails();
        System.out.println("Interest Earned: $" + cityBank.calculateInterest());

        System.out.println("\nSBI Bank Details:");
        sbiBank.displayDetails();
        System.out.println("Interest Earned: $" + sbiBank.calculateInterest());

        System.out.println("\nCanara Bank Details:");
        canaraBank.displayDetails();
        System.out.println("Interest Earned: $" + canaraBank.calculateInterest());
    }
}
```

Output

```
City Bank Details:
Name: City Bank
Account Number: 123456
Balance: $5000.0
Interest Earned: $250.0

SBI Bank Details:
Name: SBI Bank
Account Number: 789012
Balance: $8000.0
Interest Earned: $560.0

Canara Bank Details:
Name: Canara Bank
Account Number: 345678
Balance: $6000.0
Interest Earned: $360.0
```

7. Write a multi-threaded java program to illustrate producer consumer problem.

```
import java.util.LinkedList;

class SharedBuffer {
   private LinkedList<Integer> buffer = new LinkedList<>();
   private final int capacity = 5;

   public void produce() throws InterruptedException {
      int item = (int) (Math.random() * 100);
      synchronized (this) {
         while (buffer.size() == capacity) {
            System.out.println("Buffer is full. Producer is waiting...");
            wait();
         }

         System.out.println("Producer produced: " + item);
         buffer.add(item);
         notify();
      }
      Thread.sleep(1000); // Simulating production time
   }

   public void consume() throws InterruptedException {
      synchronized (this) {
         while (buffer.isEmpty()) {
            System.out.println("Buffer is empty. Consumer is waiting...");
            wait();
         }

         int consumedItem = buffer.removeFirst();
         System.out.println("Consumer consumed: " + consumedItem);
         notify();
      }
      Thread.sleep(1000); // Simulating consumption time
   }
}
```

```
class Producer extends Thread {
    private SharedBuffer sharedBuffer;

    public Producer(SharedBuffer sharedBuffer) {
        this.sharedBuffer = sharedBuffer;
    }

    @Override
    public void run() {
        try {
            while (true) {
                sharedBuffer.produce();
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}

class Consumer extends Thread {
    private SharedBuffer sharedBuffer;

    public Consumer(SharedBuffer sharedBuffer) {
        this.sharedBuffer = sharedBuffer;
    }

    @Override
    public void run() {
        try {
            while (true) {
                sharedBuffer.consume();
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
```

```
}

public class ProducerConsumerExample {
    public static void main(String[] args) {
        SharedBuffer sharedBuffer = new SharedBuffer();

        // Creating producer and consumer threads
        Producer producer = new Producer(sharedBuffer);
        Consumer consumer = new Consumer(sharedBuffer);

        // Starting threads
        producer.start();
        consumer.start();
    }
}
```

OUTPUT:-

```
Producer produced: 84
Producer produced: 51
Producer produced: 12
Consumer consumed: 84
Producer produced: 72
Producer produced: 55
Consumer consumed: 51
Producer produced: 9
Producer produced: 25
Consumer consumed: 12
Producer produced: 88
Producer produced: 4
Consumer consumed: 72
Producer produced: 31
Consumer consumed: 55
```

8. Write a java program to read two positive integers and perform the division operation on them and display the result if a user enters a positive integer and non-zero denominator. Else, If the input is negative or the denominator is zero, generate negative number input and divide by zero exception to handle the scenario.

```
import java.util.Scanner;

public class DivisionProgram {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      try {
         // Read the numerator
         System.out.println("Enter a positive numerator:");
         int numerator = scanner.nextInt();

         if (numerator <= 0) {
            throw new IllegalArgumentException("Numerator must be a positive
integer.");
         }

         // Read the denominator
         System.out.println("Enter a non-zero denominator:");
         int denominator = scanner.nextInt();

         if (denominator == 0) {
            throw new ArithmeticException("Denominator cannot be zero.");
         }

         // Perform the division
         double result = (double) numerator / denominator;

         // Display the result
         System.out.println("Result of division: " + result);
```

```
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }
}
```

OUTPUT:-

```
Enter a positive numerator:
10
Enter a non-zero denominator:
2
Result of division: 5.0

Enter a positive numerator:
-5
Error: Numerator must be a positive integer.


Enter a positive numerator:
8
Enter a non-zero denominator:
0
Error: Denominator cannot be zero.
```

9. Write a java program to illustrate the concept of packages and interfaces in java.

```
import java.util.Scanner;

public class Conversion {

    // Interface for conversion computation
    interface Compute {
        double convert(double value);
    }

    // Concrete class for kilobyte to gigabyte conversion
    static class KiloToGiga implements Compute {
        public double convert(double kilobyte) {
            return kilobyte / (1024 * 1024); // Conversion formula
        }
    }

    // Concrete class for euro to rupee conversion
    static class EuroToRupees implements Compute {
        public double convert(double euro) {
            return euro * 89.70; // Conversion formula
        }
    }

    public static void main(String[] args) {
        // Creating instances of conversion classes
        Compute kiloToGiga = new KiloToGiga();
        Compute euroToRupees = new EuroToRupees();

        Scanner input = new Scanner(System.in);

        // Getting input for kilobytes
        System.out.println("Enter the value of kilobyte:");
        double kb = input.nextDouble();

        // Getting input for euros
        System.out.println("Enter the value of euro:");
        double er = input.nextDouble();
```

```
        // Performing conversions
        double inGiga = kiloToGiga.convert(kb);
        double inRupees = euroToRupees.convert(er);

        // Displaying the results
        System.out.println("After conversion of kilobyte to gigabyte is: " +
inGiga + " gigabytes");
        System.out.println("After conversion of euro to Rupees is: " +
inRupees + " Rupees");

        input.close();
    }
}
```

OUTPUT

```
Enter the value of kilobyte:
2048
Enter the value of euro:
50.5
After conversion of kilobyte to gigabyte is: 0.001953125 gigabytes
After conversion of euro to Rupees is: 4523.85 Rupees
```

10. Write a java **program that connects to a database using JDBC and does add deletes, modify and retrieve operations.**

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class JdbcExample {
   private static final String JDBC_URL =
"jdbc:mysql://localhost:3306/students_details";
   private static final String USERNAME = "root";
   private static final String PASSWORD = "";

   public static void main(String[] args) {
      try {
         // Establishing the database connection
         try (Connection connection = DriverManager.getConnection(JDBC_URL,
USERNAME, PASSWORD)) {
            // Retrieving student details
            String query = "SELECT * FROM students";
            try (PreparedStatement preparedStatement =
connection.prepareStatement(query);
                 ResultSet resultSet = preparedStatement.executeQuery()) {

               // Displaying student details
               while (resultSet.next()) {
                  String name = resultSet.getString("name");
                  String usn = resultSet.getString("usn");
                  int semester = resultSet.getInt("semester");
                  int age = resultSet.getInt("age");
                  String course = resultSet.getString("course");
                  String dob = resultSet.getString("dob");
                  String address = resultSet.getString("address");

                  System.out.println("Name: " + name + ", USN: " + usn + ",
Semester: " + semester +
```

```
                    ", Age: " + age + ", Course: " + course + ", DOB: " + dob + ",
Address: " + address);
                }
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
  }
}
```

OUTPUT

Name: John Doe, USN: ABC123, Semester: 3, Age: 20, Course: Computer Science, DOB: 2001-01-15, Address: 123 Main St
Name: Jane Smith, USN: XYZ456, Semester: 2, Age: 21, Course: Electrical Engineering, DOB: 2000-05-20, Address: 456 Oak St

11. Write a java servlet program which displays cookie id.

A web application that takes name and age from an HTML page. If the age is less than 18, it should send a page with “Hello , you are not authorized to visit the site” message, where should be replaced with the entered name. Otherwise it should send “Welcome to this site” message.

votersrv.java

```
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class VoterSrv extends HttpServlet
{
 public void service(HttpServletRequest req, HttpServletResponse res) throws
IOException,ServletException
 {
 //set response content type
 res.setContentType("text/html");
 //get printWrite obj
 PrintWriter pw = res.getWriter();
 //read form data from page as request parameter
 String name = req.getParameter("name");
 int age = Integer.parseInt(req.getParameter("age"));
 if (age>=18)
 {
 pw.println("<font color='green' size='4'>Welcome "+name+" to this site</font>");
 }
 else
```

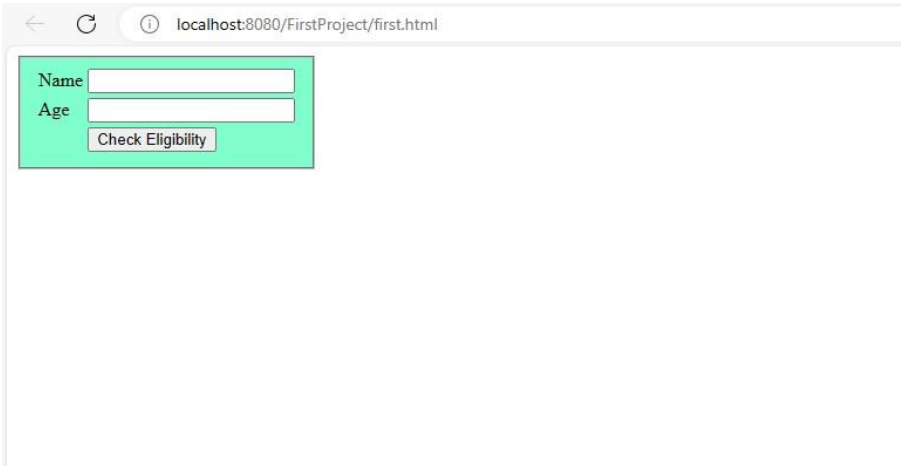
```
 pw.println("<font color='red' size='4'>Hello "+name+", you are not authorized to
visit the site</font>");
 pw.println("<br><br><a href= 'index.html'>back</a>");
 //close the stream
 pw.close();
 }
}
```

index.html

```
<html
<head>
 <title>VoterApp</title>
 </head>
 <body>
 <form action= "http://localhost:8080/Hello1/check" method="get">
 <fieldset style="width:20%; background-color:#80ffcc">
 <table>
 <tr><td>Name</td><td><input type="text" name="name"></td></tr>
 <tr><td>Age</td><td><input type="text" name="age"></td></tr>
 <tr><td></td>
 <td><input type = "submit" value="Check Eligibility"></td></tr>
 </table>
 </fieldset>
 </form>
 </body>
</html>
```

web.xml

```
<web-app>
 <servlet>
 <servlet-name>abc</servlet-name>
 <servlet-class>VoterSrv</servlet-class>
 </servlet>
 <servlet-mapping>
 <servlet-name>abc</servlet-name>
 <url-pattern>/check</url-pattern>
 </servlet-mapping>
</web-app>
```



12. Create a registration form using JSP, servelet, JDBC, MySQL.

A web application that takes a name as input and on submit it shows a hello page where name is taken from the request. It shows the start time at the right top corner of the page and provides a logout button. On clicking this button, it should show a logout page with Thank You message with the duration of usage (hint: Use session to store name and time).

Sessionjsp.html

```
<html>
<head> <title> SESSION LOGIN </title> </head>
<body>
<center>
<form action="http://localhost:8080/jsp/Session1.jsp" method="get">
Enter Name: <input type="text" name="uname"> <br>
<input type="submit" value="LOGIN" name="register">
</form>
</center>
</body>
</html>
```

Session1.jsp

```
<%@page language="java" import="java.util.*" errorPage=""%>
<form method="get" action="http://localhost:8080/jsp/Session2.jsp">
```

```
<%
Date d=new Date();
%>
<p align="right"> Time;<%=d.getTime()%></p>
<%
String un=request.getParameter("uname");
session.setAttribute("user",un);
session.setAttribute("time",d.getTime());
%>
Hello <%=un%>
<br><br>
<input type="submit" value="logout">
</form>
```

Session2.jsp

```
<%@page language="java" import="java.util.*" errorPage=""%>
<%
Date d2=new Date();
String un=(String)session.getAttribute("user");
Long t1=(Long)session.getAttribute("time");
Long t2=d2.getTime();
%>
Thank you <%=un%>
<br><br>
Session duration: <%=(t2-t1)/(60*60)%> seconds
<% session.invalidate();%>
```

Thank you jaya

Session duration: 7 seconds