**Q1.** WAJP to create package p1 in which defines an interface “FactInterf”. FactInterf has one method int fact(int n); Create another package p2, in p2 define a class DemoFact which will implement FactInterf interface. Create the main class “Demo Main” in the p3 package with a main method test the functionality of the DemoFact class

**Solution:**

**p1/FactInterf.java**

package p1;

public interface FactInterf {

int fact(int n);

}

**p2/DemoFact.java**

package p2;

import p1.FactInterf;

public class DemoFact implements FactInterf {

@Override

public int fact(int n) {

if (n == 0 || n == 1) {

return 1;

}

return n \* fact(n - 1);

}

}

**p3/DemoMain.java**

package p3;

import p2.DemoFact;

import p1.FactInterf;

public class DemoMain {

public static void main(String[] args) {

FactInterf obj = new DemoFact();

System.out.println("Factorial of 5: " + obj.fact(5));

} }

**Output:** Factorial of 5: 120

**Q2.** Assume two threads are running, the main thread and the child thread. Write a Java program to enter the main thread into waiting- state till the child thread completes its execution.

**Solution :**

public class MainThreadWait {

public static void main(String[] args) {

Thread childThread = new Thread(() -> {

for (int i = 1; i <= 5; i++) {

System.out.println("Child thread: " + i);

try {

Thread.sleep(500);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

});

childThread.start();

try {

childThread.join(); // Main thread waits for child thread to complete

} catch (InterruptedException e) {

e.printStackTrace();

}

System.out.println("Main thread continues after child thread completes");

}

}

**Output:**

Child thread: 1

Child thread: 2

Child thread: 3

Child thread: 4

Child thread: 5

Main thread continues after child thread completes

Q3. Assume two threads are running, the main thread and the child thread. Write a Java program to enter the child thread into waiting-state till the main thread completes its execution.

**Solution:**

public class ChildThreadWait {

static boolean mainCompleted = false;

public static void main(String[] args) {

Thread childThread = new Thread(() -> {

synchronized (ChildThreadWait.class) {

while (!mainCompleted) {

try {

ChildThreadWait.class.wait();

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

System.out.println("Child thread starts after main completes");

for (int i = 1; i <= 3; i++) {

System.out.println("Child thread: " + i);

}

});

childThread.start();

// Main thread work

for (int i = 1; i <= 5; i++) {

System.out.println("Main thread: " + i);

}

synchronized (ChildThreadWait.class) {

mainCompleted = true;

ChildThreadWait.class.notify();

}

} }

**Output:**

Main thread: 1

Main thread: 2

Main thread: 3

Main thread: 4

Main thread: 5

Child thread starts after main completes

Child thread: 1

Child thread: 2

Child thread: 3

**Q4.** Write a Java program to create a ResourceClass with a drawPattern(char ch) method. Then create two threads t1 and t2 and demonstrate thread synchronization.

**Solution:**

class ResourceClass {

synchronized void drawPattern(char ch) {

for (int i = 0; i < 5; i++) {

for (int j = 0; j <= i; j++) {

System.out.print(ch);

}

System.out.println();

}

}

}

public class ThreadSynchronization {

public static void main(String[] args) {

ResourceClass resource = new ResourceClass();

Thread t1 = new Thread(() -> {

resource.drawPattern('\*');

});

Thread t2 = new Thread(() -> {

resource.drawPattern('#');

});

t1.start();

t2.start();

}

}

**Output:**

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

#

##

###

####

#####

**Q5.** Write a Java program to read the content from one.txt and two.txt and write the content of the files into a third file( Merge the two files into one file)

**Solution:**

import java.io.\*;

public class FileMerger {

public static void main(String[] args) {

try {

// Create output file

PrintWriter pw = new PrintWriter("three.txt");

// Read and write first file

BufferedReader br = new BufferedReader(new FileReader("one.txt"));

String line = br.readLine();

while (line != null) {

pw.println(line);

line = br.readLine();

}

br.close();

// Read and write second file

br = new BufferedReader(new FileReader("two.txt"));

line = br.readLine();

while (line != null) {

pw.println(line);

line = br.readLine();

}

br.close();

pw.close();

System.out.println("Files merged successfully into three.txt");

} catch (IOException e) {

e.printStackTrace();

}

} }

**Output (assuming one.txt contains "Hello" and two.txt contains "World"):**

Files merged successfully into three.txt

three.txt will contain:

Hello

World

**Q6.** Write a Java program that performs the following tasks: Prompts the user to enter five lines of text from the console.

Saves each line into a file called output.txt using the PrintWriter class. After writing the lines to the file, it reads the content back from output.txt using the BufferedReader class. Finally, prints each line to the console with line numbers (e.g., "Line 1: <content>").

Output

Enter 5 lines of text:

Hello

Welcome to Java

This is a test

Reading from file:

Line 1: Hello

Line 2: Welcome to Java

Line 3: This is a test

**Solution:**

import java.io.\*;

import java.util.Scanner;

public class FileLineNumbers {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

try (PrintWriter pw = new PrintWriter("output.txt")) {

System.out.println("Enter 5 lines of text:");

for (int i = 0; i < 5; i++) {

String line = scanner.nextLine();

pw.println(line);

}

} catch (IOException e) {

e.printStackTrace();

}

System.out.println("Reading from file:");

try (BufferedReader br = new BufferedReader(new FileReader("output.txt"))) {

String line;

int lineNumber = 1;

while ((line = br.readLine()) != null) {

System.out.println("Line " + lineNumber + ": " + line);

lineNumber++;

}

} catch (IOException e) {

e.printStackTrace();

}

}

}

**Output:**

Enter 5 lines of text:

Hello

Welcome to Java

This is a test

Programming is fun

Goodbye

Reading from file:

Line 1: Hello

Line 2: Welcome to Java

Line 3: This is a test

Line 4: Programming is fun

Line 5: Goodbye

Q7. Write a Java program that Prompts the user to enter a sentence from the console. Writes the sentence into a file named sentence.txt using PrintWriter. Reads the sentence back from sentence.txt using BufferedReader. Prints the reversed sentence to the console.

Output:

Enter a sentence:

Java is a powerful language

Reversed sentence:

Language powerful is Java

**Solution :**

import java.io.\*;

import java.util.Scanner;

public class ReverseSentence {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter a sentence:");

String sentence = scanner.nextLine();

try (PrintWriter pw = new PrintWriter("sentence.txt")) {

pw.println(sentence);

} catch (IOException e) {

e.printStackTrace();

}

try (BufferedReader br = new BufferedReader(new FileReader("sentence.txt"))) {

String storedSentence = br.readLine();

String[] words = storedSentence.split(" ");

StringBuilder reversed = new StringBuilder();

for (int i = words.length - 1; i >= 0; i--) {

if (i == words.length - 1) {

// Capitalize first word of reversed sentence

words[i] = words[i].substring(0, 1).toUpperCase() +

words[i].substring(1).toLowerCase();

}

reversed.append(words[i]).append(" ");

}

System.out.println("\nReversed sentence:");

System.out.println(reversed.toString().trim());

} catch (IOException e) {

e.printStackTrace();

}

}

}

**Output:**

Enter a sentence:

Java is a powerful language

Reversed sentence:

Language powerful a is Java

**Q8.** Create a class Person with name and id data members. Define the setData(int id, String name) method to initialize the instance variable. Now write the state of the object using Serialization and print the write operation completed. Now read the state of the object using Object Deserialization.

**Solution :**

import java.io.\*;

class Person implements Serializable {

private String name;

private int id;

public void setData(int id, String name) {

this.id = id;

this.name = name;

}

@Override

public String toString() {

return "Person [id=" + id + ", name=" + name + "]";

}

}

public class SerializationDemo {

public static void main(String[] args) {

Person person = new Person();

person.setData(101, "John Doe");

// Serialization

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("person.ser"))) {

oos.writeObject(person);

System.out.println("Write operation completed");

} catch (IOException e) {

e.printStackTrace();

}

// Deserialization

try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream("person.ser"))) {

Person deserializedPerson = (Person) ois.readObject();

System.out.println("Deserialized object: " + deserializedPerson);

} catch (IOException | ClassNotFoundException e) {

e.printStackTrace();

}

}

}

**Output:**

Write operation completed

Deserialized object: Person [id=101, name=John Doe]

**Q9:** Write a Java program to write integer ranges from 1 to 999 line by line in the number.txt file. Reads each integer and checks whether it is prime or not. If the number is prime, then write this integer into prime.txt. Now read prime.txt and print the prime numbers.

**Solution:**

import java.io.\*;

import java.util.ArrayList;

import java.util.List;

public class PrimeNumberFile {

public static void main(String[] args) {

// Write numbers 1-999 to file

try (PrintWriter pw = new PrintWriter("number.txt")) {

for (int i = 1; i <= 999; i++) {

pw.println(i);

}

} catch (IOException e) {

e.printStackTrace();

}

// Read numbers and check for primes

List<Integer> primes = new ArrayList<>();

try (BufferedReader br = new BufferedReader(new FileReader("number.txt"))) {

String line;

while ((line = br.readLine()) != null) {

int num = Integer.parseInt(line);

if (isPrime(num)) {

primes.add(num);

}

}

} catch (IOException e) {

e.printStackTrace();

}

// Write primes to file

try (PrintWriter pw = new PrintWriter("prime.txt")) {

for (int prime : primes) {

pw.println(prime); }

} catch (IOException e) {

e.printStackTrace();

}

// Read and print primes

System.out.println("Prime numbers between 1-999:");

try (BufferedReader br = new BufferedReader(new FileReader("prime.txt"))) {

String line;

while ((line = br.readLine()) != null) {

System.out.print(line + " ");

}

} catch (IOException e) {

e.printStackTrace();

}

}

private static boolean isPrime(int num) {

if (num <= 1) return false;

if (num == 2) return true;

if (num % 2 == 0) return false;

for (int i = 3; i \* i <= num; i += 2) {

if (num % i == 0) return false;

}

return true;

}

}

**Output (partial):**

Prime numbers between 1-999:

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 ... 997

Q10. Write a Java program to enter the amount from the user using the Scanner class. If the amount < 500, then throw MinAmouuntException. If the amount > 1000, then throw MaxAmountException.

**Soliution:**

import java.util.Scanner;

class MinAmountException extends Exception {

public MinAmountException(String message) {

super(message);

}

}

class MaxAmountException extends Exception {

public MaxAmountException(String message) {

super(message);

}

}

public class AmountValidator {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter amount: ");

int amount = scanner.nextInt();

try {

validateAmount(amount);

System.out.println("Amount " + amount + " is valid");

} catch (MinAmountException e) {

System.out.println("Error: " + e.getMessage());

} catch (MaxAmountException e) {

System.out.println("Error: " + e.getMessage());

}

}

public static void validateAmount(int amount) throws MinAmountException, MaxAmountException {

if (amount < 500) {

throw new MinAmountException("Amount must be at least 500");

}

if (amount > 1000) {

throw new MaxAmountException("Amount must be at most 1000");

}

}

}

**Output Examples:**

Enter amount: 400

Error: Amount must be at least 500

Enter amount: 1200

Error: Amount must be at most 1000

Enter amount: 750

Amount 750 is valid

Q11. Write Client-Server Application using the Socket class. The client will send the number to the server, and the server will calculate the factorial of that number and send it back to the client.

**Solution:**

**Server.java**

import java.io.\*;

import java.net.\*;

public class Server {

public static void main(String[] args) {

try (ServerSocket serverSocket = new ServerSocket(1234)) {

System.out.println("Server started. Waiting for client...");

while (true) {

Socket clientSocket = serverSocket.accept();

System.out.println("Client connected: " + clientSocket);

new ClientHandler(clientSocket).start();

}

} catch (IOException e) {

e.printStackTrace();

}

}

}

class ClientHandler extends Thread {

private Socket clientSocket;

public ClientHandler(Socket socket) {

this.clientSocket = socket;

}

public void run() {

try (

DataInputStream dis = new DataInputStream(clientSocket.getInputStream());

DataOutputStream dos = new DataOutputStream(clientSocket.getOutputStream()); ) {

int number = dis.readInt();

System.out.println("Received number from client: " + number);

long factorial = calculateFactorial(number);

dos.writeLong(factorial);

System.out.println("Sent factorial to client: " + factorial);

} catch (IOException e) {

e.printStackTrace();

}

}

private long calculateFactorial(int n) {

if (n == 0) return 1;

long result = 1;

for (int i = 1; i <= n; i++) {

result \*= i;

}

return result;

}

}

**Client.java**

import java.io.\*;

import java.net.\*;

import java.util.Scanner;

public class Client {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a number to calculate factorial: ");

int number = scanner.nextInt();

try (

Socket socket = new Socket("localhost", 1234);

DataOutputStream dos = new DataOutputStream(socket.getOutputStream());

DataInputStream dis = new DataInputStream(socket.getInputStream());

) {

dos.writeInt(number);

long factorial = dis.readLong();

System.out.println("Factorial of " + number + " is: " + factorial);

} catch (IOException e) {

e.printStackTrace();

}

}

}

**Output:**

**Server output:**

Server started. Waiting for client...

Client connected: Socket[addr=/127.0.0.1,port=65432,localport=1234]

Received number from client: 5

Sent factorial to client: 120

**Client output:**

Enter a number to calculate factorial: 5

Factorial of 5 is: 120

**Q12.** Assume a Product table in sql/mysql with id, name, and price. Write a JDBC program to fetch all the records of the Product table on the console using the Statement interface.

**Solution:**

import java.sql.\*;

public class ProductFetcher {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/mydatabase";

String username = "root";

String password = "password";

try (Connection conn = DriverManager.getConnection(url, username, password);

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM Product")) {

System.out.println("ID\tName\t\tPrice");

System.out.println("-----------------------------");

while (rs.next()) {

int id = rs.getInt("id");

String name = rs.getString("name");

double price = rs.getDouble("price");

System.out.println(id + "\t" + name + "\t\t" + price);

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Output:**

ID Name Price

-----------------------------

1 Laptop 999.99

2 Smartphone 699.99

3 Headphones 99.99

Q13. Assume a Product table in sql/mysql with id, name, and price. Ask a user to enter the details using Scanner class. Write a JDBC program to insert all the records of the Product table using PreparedStatement.

**Solution :**

import java.sql.\*;

import java.util.Scanner;

public class ProductInserter {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/mydatabase";

String username = "root";

String password = "password";

Scanner scanner = new Scanner(System.in);

System.out.println("Enter product details:");

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

int id = scanner.nextInt();

scanner.nextLine(); // consume newline

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

String name = scanner.nextLine();

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

double price = scanner.nextDouble();

String sql = "INSERT INTO Product (id, name, price) VALUES (?, ?, ?)";

try (Connection conn = DriverManager.getConnection(url, username, password);

PreparedStatement pstmt = conn.prepareStatement(sql)) {

pstmt.setInt(1, id);

pstmt.setString(2, name);

pstmt.setDouble(3, price);

int rowsAffected = pstmt.executeUpdate();

System.out.println(rowsAffected + " row(s) inserted successfully!");

} catch (SQLException e) {

e.printStackTrace();

}

} }

**Output:**

Enter product details:

ID: 4

Name: Tablet

Price: 299.99

1 row(s) inserted successfully!

Q14. Assume a Product table in sql/mysql with id, name, and price. Create a Procedure getName(IN,OUT) in mysql/SQL. Write a JDBC program to call the getName() procedure and Print the name of the product.

**Solution:**

**MySQL Procedure:**

DELIMITER //

CREATE PROCEDURE getName(IN p\_id INT, OUT p\_name VARCHAR(255))

BEGIN

SELECT name INTO p\_name FROM Product WHERE id = p\_id;

END //

DELIMITER ;

**Java Code:**

import java.sql.\*;

import java.util.Scanner;

public class ProductNameGetter {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/mydatabase";

String username = "root";

String password = "password";

Scanner scanner = new Scanner(System.in);

System.out.print("Enter product ID: ");

int id = scanner.nextInt();

try (Connection conn = DriverManager.getConnection(url, username, password);

CallableStatement cstmt = conn.prepareCall("{call getName(?, ?)}")) {

cstmt.setInt(1, id);

cstmt.registerOutParameter(2, Types.VARCHAR);

cstmt.execute();

String name = cstmt.getString(2);

System.out.println("Product name for ID " + id + ": " + name);

} catch (SQLException e) {

e.printStackTrace();

} } }

**Output:**

Enter product ID: 2

Product name for ID 2: Smartphone

**Q15.** Assume a Product table in sql/mysql with id, name, and price. Create a Procedure updatePrice (IN) in mysql/SQL. Ask a user to enter the id from Write a JDBC program to call the updatePrice() procedure and update the price of Product use CallableStatement.

**Solution :**

**MySQL Procedure:**

DELIMITER //

CREATE PROCEDURE updatePrice(IN p\_id INT, IN p\_price DECIMAL(10,2))

BEGIN

UPDATE Product SET price = p\_price WHERE id = p\_id;

END //

DELIMITER ;

**Java Code:**

import java.sql.\*;

import java.util.Scanner;

public class ProductPriceUpdater {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/mydatabase";

String username = "root";

String password = "password";

Scanner scanner = new Scanner(System.in);

System.out.print("Enter product ID to update: ");

int id = scanner.nextInt();

System.out.print("Enter new price: ");

double newPrice = scanner.nextDouble();

try (Connection conn = DriverManager.getConnection(url, username, password);

CallableStatement cstmt = conn.prepareCall("{call updatePrice(?, ?)}")) {

cstmt.setInt(1, id);

cstmt.setDouble(2, newPrice);

int rowsAffected = cstmt.executeUpdate();

System.out.println(rowsAffected + " row(s) updated successfully!");

} catch (SQLException e) {

e.printStackTrace(); } } }

**Output:**

Enter product ID to update: 3

Enter new price: 89.99

1 row(s) updated successfully!

Q16. Assume a Product table in sql/mysql with id, name, and price. Write a JDBC program to ask a user to enter the row number and read the record. Use absolute() method ResultSet.

**Solution:**

import java.sql.\*;

import java.util.Scanner;

public class ReadSpecificRow {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/mydatabase";

String username = "root";

String password = "password";

Scanner scanner = new Scanner(System.in);

System.out.print("Enter row number to read: ");

int rowNum = scanner.nextInt();

try (Connection conn = DriverManager.getConnection(url, username, password);

Statement stmt = conn.createStatement(

ResultSet.TYPE\_SCROLL\_INSENSITIVE,

ResultSet.CONCUR\_READ\_ONLY);

ResultSet rs = stmt.executeQuery("SELECT \* FROM Product")) {

if (rs.absolute(rowNum)) {

System.out.println("Row " + rowNum + " details:");

System.out.println("ID: " + rs.getInt("id"));

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

System.out.println("Price: " + rs.getDouble("price"));

} else {

System.out.println("Row " + rowNum + " doesn't exist in the result set");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Output:**

Enter row number to read: 2

Row 2 details:

ID: 2

Name: Smartphone

Price: 699.99

**Q17.** Create a web application using servlet. Create index.html page, onclick on submit button call Add.java servlet. Add.java servlet will fetch the parameter values and calculate the sum. Use doGet() method.

**Solution:**

**index.html**

<!DOCTYPE html>

<html>

<head> <title>Addition Servlet</title> </head>

<body>

<form action="Add" method="get">

Number 1: <input type="text" name="num1"><br>

Number 2: <input type="text" name="num2"><br>

<input type="submit" value="Add">

</form>

</body>

</html>

**Add.java**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class Add extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

int num1 = Integer.parseInt(request.getParameter("num1"));

int num2 = Integer.parseInt(request.getParameter("num2"));

int sum = num1 + num2;

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<html><body>");

out.println("<h2>Sum is: " + sum + "</h2>");

out.println("</body></html>");

} }

**Output:**

After entering 10 and 20 in the form:

Sum is: 30

Q18. Create a web application using servlet. Create the index.html page, click on the submit button, and call Fact.java servlet. Fact.java servlet will fetch the parameter values and calculate the factorial of a number. Use doGet() method.

**index.html**

<!DOCTYPE html>

<html>

<head>

<title>Factorial Servlet</title>

</head>

<body>

<form action="Fact" method="get">

Enter a number: <input type="text" name="number"><br>

<input type="submit" value="Calculate Factorial">

</form>

</body>

</html>

**Fact.java**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class Fact extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

int num = Integer.parseInt(request.getParameter("number"));

long factorial = 1;

for(int i = 1; i <= num; i++) {

factorial \*= i;

}

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<html><body>");

out.println("<h2>Factorial of " + num + " is: " + factorial + "</h2>");

out.println("</body></html>");

}

}

**Output:**

After entering 5 in the form:

Factorial of 5 is: 120

Q19 . Create a web application using servlet. Create index.html page, onclick on the submit button, call Firstt.java servlet will fetch the parameter values and calculate the sum of the parameter values. Then call Second Servlet, Second Servlet will check whether a number is Prime or Not. Use RequestDispatcher.

**Solution:**

**index.html**

<!DOCTYPE html>

<html>

<head> <title>Number Processor</title> </head>

<body>

<form action="First" method="get">

Enter a number: <input type="text" name="number"><br>

<input type="submit" value="Process">

</form>

</body>

</html>

**First.java**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class First extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

int num = Integer.parseInt(request.getParameter("number"));

int sum = 0;

for(int i = 1; i <= num; i++) {

sum += i; }

request.setAttribute("sum", sum);

request.setAttribute("number", num);

RequestDispatcher rd = request.getRequestDispatcher("Second");

rd.forward(request, response);

} }

**Second.java**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class Second extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

int sum = (Integer)request.getAttribute("sum");

int num = (Integer)request.getAttribute("number");

boolean isPrime = isPrime(num);

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<html><body>");

out.println("<h2>Sum of first " + num + " numbers: " + sum + "</h2>");

out.println("<h2>" + num + " is " + (isPrime ? "prime" : "not prime") + "</h2>");

out.println("</body></html>");

}

private boolean isPrime(int n) {

if (n <= 1) return false;

for (int i = 2; i <= Math.sqrt(n); i++) {

if (n % i == 0) return false;

}

return true;

}

}

**Output:**

After entering 7 in the form:

Sum of first 7 numbers: 28

7 is prime

Q20. Create a web application using servlet. Create index.html page, onclick on the submit button, call First.java servlet will fetch the parameter values and calculate the sum of the parameter values. Then redirect the client request to Second Servlet, Second Servlet will calculate the square of or Not. Use sendRedirect and HttpSession.

**index.html**

<!DOCTYPE html>

<html>

<head> <title>Number Processor</title> </head>

<body>

<form action="FirstServlet" method="get">

Enter a number: <input type="text" name="number"><br>

<input type="submit" value="Process">

</form>

</body>

</html>

**FirstServlet.java**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class FirstServlet extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

int num = Integer.parseInt(request.getParameter("number"));

int sum = 0;

for(int i = 1; i <= num; i++) {

sum += i;

}

HttpSession session = request.getSession();

session.setAttribute("sum", sum);

response.sendRedirect("SecondServlet");

}

}

**SecondServlet.java**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class SecondServlet extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

HttpSession session = request.getSession();

int sum = (Integer)session.getAttribute("sum");

int square = sum \* sum;

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<html><body>");

out.println("<h2>Square of the sum is: " + square + "</h2>");

out.println("</body></html>");

}

}

**Output:**

After entering 3 in the form:

Square of the sum is: 36

Q21. Create a Login application using Servlet.

**Solution :**

**index.html**

<!DOCTYPE html>

<html>

<head> <title>Login</title> </head>

<body>

<form action="LoginServlet" method="post">

Username: <input type="text" name="username"><br>

Password: <input type="password" name="password"><br>

<input type="submit" value="Login">

</form>

</body>

</html>

**LoginServlet.java**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class LoginServlet extends HttpServlet {

private final String VALID\_USERNAME = "admin";

private final String VALID\_PASSWORD = "password123";

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

String username = request.getParameter("username");

String password = request.getParameter("password");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

if(username.equals(VALID\_USERNAME) && password.equals(VALID\_PASSWORD)) {

out.println("<h2>Login Successful!</h2>");

out.println("<p>Welcome, " + username + "</p>");

} else {

out.println("<h2>Login Failed!</h2>");

out.println("<p>Invalid username or password</p>");

out.println("<a href='index.html'>Try again</a>");

}

}

}

**Output (success case):**

Login Successful!

Welcome, admin

**Output (failure case):**

Login Failed!

Invalid username or password

Try again

Q22. Create a StudentRegistration web application using JSP on click submit button insert student details in database.

**Solution :**

**index.jsp**

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<title>Student Registration</title>

</head>

<body>

<h2>Student Registration Form</h2>

<form action="register.jsp" method="post">

Roll No: <input type="text" name="rollno"><br>

Name: <input type="text" name="name"><br>

Email: <input type="text" name="email"><br>

Course:

<select name="course">

<option value="CS">Computer Science</option>

<option value="EE">Electrical Engineering</option>

<option value="ME">Mechanical Engineering</option>

</select><br>

<input type="submit" value="Register">

</form>

</body>

</html>

**register.jsp**

<%@ page import="java.sql.\*" %>

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<%

String rollno = request.getParameter("rollno");

String name = request.getParameter("name");

String email = request.getParameter("email");

String course = request.getParameter("course");

try {

Class.forName("com.mysql.jdbc.Driver");

Connection con = DriverManager.getConnection(

"jdbc:mysql://localhost:3306/mydatabase", "root", "password");

PreparedStatement ps = con.prepareStatement(

"INSERT INTO students (rollno, name, email, course) VALUES (?, ?, ?, ?)");

ps.setString(1, rollno);

ps.setString(2, name);

ps.setString(3, email);

ps.setString(4, course);

int i = ps.executeUpdate();

if(i > 0) {

out.println("<h2>Registration Successful!</h2>");

out.println("<p>Student " + name + " registered with Roll No: " + rollno + "</p>");

}

con.close();

} catch(Exception e) {

out.println("<h2>Registration Failed!</h2>");

out.println("<p>Error: " + e.getMessage() + "</p>");

}

%>

**Output (success case):**

Registration Successful!

Student John Doe registered with Roll No: 101

**Output (failure case):**

Registration Failed!

Error: Duplicate entry '101' for key 'PRIMARY'