Practical No. 1

1.1 Generic Class

Aim: Write a java program to demonstrate a generic class

Code:

package pract1;

class Test <T>{

T obj;

Test(T obj){this.obj=obj;}

public T getobject() {return this.obj;}

}

class genericClass{

public static void main(String[] args) {

Test<Integer> iobj = new Test<Integer>(20);

System.out.println("The value of Integer :"+iobj.getobject());

Test<String> sobj = new Test<String>("XYZ");

System.out.println("The value of String :"+sobj.getobject());

}

}

(1.2) Generic Method

Aim: Write a java program to demonstrate a generic method

Code:

package practical1;

public class GenericMethod {

void display() {

System.out.println("generic Method example");

}

<T> void gdisplay(T e) {

System.out.println(e.getClass().getName()+"="+e);

}

public static void main(String[] args) {

GenericMethod g1= new GenericMethod();

g1.display();

g1.gdisplay(1);

g1.gdisplay("XYZ");

g1.gdisplay(11.0);

}

}

(1.3) Wildcard

Aim: Write a java program to demonstrate wildcard in java generic.

Code:

package practical1;

import java.util.\*;

public class Wildcard {

//upper bound

private static double sum(List<? extends Number>list) {

double sum=0.0;

for(Number i : list) {

sum=sum+i.doubleValue();

}

return sum;

}

// Lower Bound

private static void show(List <? super Integer>list) {

list.forEach((x)->{

System.out.println(x+" ");

});

}

public static void main(String[] args) {

// TODO Auto-generated method stub

System.out.println("Upper Bound");

List<Integer> list1=Arrays.asList(4,2,7,5,1,9);

System.out.println("List 1 sum :"+sum(list1));

List<Double> list2=Arrays.asList(4.7,2.4,7.3,5.4,1.5,9.2);

System.out.println("\nLower Bound");

System.out.println("List 2 sum :"+sum(list2));

List<Integer> list3=Arrays.asList(4,2,7,5,1,9);

System.out.println("Only classes with integer Superclass will be Accepted :");

show(list3);

}

}

Practical No. 2

(2.1) List

Aim: Write a Java program to create a List containing a list of items of type String

and use for each loop to print the items of the list.

Code:

package practical;

import java.util.ArrayList;

public class Practical\_2\_1 {

public static void main(String[] args) {

ArrayList <String> list = new ArrayList<>();

list.add("maths");

list.add("Ajava");

list.add("Adbms");

list.add("Spm");

System.out.println(list);

System.out.println("Traversing list through for each loop:");

for(String subject : list ) {

System.out.println(subject);

}

}

}

(2.2) List Iterator

Aim: Write a Java program to create List containing list of items and use List Iterator

Interface to print items present in the list. Also, print the list in reverse/ backward

Direction.

Code:

package practical;

import java.util.ArrayList;

import java.util.List;

import java.util.ListIterator;

public class Practical\_2\_2 {

public static void main(String[] args) {

List<String> Products= new ArrayList<>();

Products.add("Product 1");

Products.add("Product 2");

Products.add("Product 3");

Products.add("Product 4");

Products.add("Product 5");

System.out.println("printing products in Forward direction using listiterator");

ListIterator <String> frowardIterator=Products.listIterator();

while(frowardIterator.hasNext()) {

System.out.println(frowardIterator.next());

listiterator");

}

System.out.println("printing products in Backward direction using

ListIterator <String> backwardIterator=Products.listIterator(Products.size());

while(backwardIterator.hasPrevious()) {

System.out.println(backwardIterator.previous());

}

}

}

Practical No.3

(3.1) SETS

Aim: Write a Java program to create a Set containing list of items of type String and

print the items in the list using Iterator interface. Also, print the list in reverse /

backward direction.

Code:

package practical\_3;

import java.util.LinkedList;

import java.util.List;

import java.util.ListIterator;

public class Set {

public static void main(String[] args) {

// TODO Auto-generated method stub

List <String> names=new LinkedList<>();

names.add("Anish");

names.add("XYZ");

names.add("PQR");

names.add("ABC");

ListIterator<String> listiterator =names.listIterator();

System.out.println("forward Direction Iterator");

while(listiterator.hasNext()) {

System.out.println(listiterator.next());

}

System.out.println("Backward Direction Iterator");

while(listiterator.hasPrevious()) {

System.out.println(listiterator.previous());

}

}

}

(3.2) Set Operations

Aim: Write a Java program using Set interface containing list of items and perform

the following operations:

a. Add items in the set.

b. Insert items of one set into another set.

c. Remove items from the set

d. Search the specified item in the set

Code:

package practical\_3;

import java.util.HashSet;

import java.util.Set;

;

public class Set2 {

public static void main(String[] args) {

// TODO Auto-generated method stub

Set<String> set1= new HashSet<>();

set1.add("Anish");

set1.add("XYZ");

set1.add("ABC");

set1.add("Rajan");

System.out.println("Element ofset 1 :"+set1);

Set<String> set2= new HashSet<>();

set2.add("Jay");

set2.add("Hind");

set2.add("Surya");

System.out.println("Element ofset 2 :"+set2);

//adding Set 2 element into set 1

set1.addAll(set2);

System.out.println("Adding set 2 element into set1");

System.out.println(set1);

//Removing an element from a set

set1.remove("Hind");

System.out.println("Removing element from set");

System.out.println(set1);

//Search for an item in a set

boolean isPresent=set1.contains("Parth");

System.out.println("Is Parth present in set :"+isPresent);

}

}

Practical No. 4

Aim: Write a Java program using Map interface containing list of items having keys,

associated values, and perform the following operations:

a. Add items in the map.

b. Remove items from the map

c. Search specific key from the map

d. Get value of the specified key

e. Insert map elements of one map into another map.

f. Print all keys and values of the map.

Code:

package prcartical\_4;

import java.util.HashMap;

import java.util.Map;

public class map {

public static void main(String[] args) {

// TODO Auto-generated method stub

Map<Integer,String>map1=new HashMap<>();

map1.put(1, "XYZ");

map1.put(2, "ABC");

map1.put(3, "Suraj");

map1.put(4, "Raju");

map1.put(5, "PQR");

System.out.println("Map1: "+map1);

Map<Integer,String>map2=new HashMap<>();

map2.put(6, "Anish");

map2.put(7, "Sahil");

map2.put(8, "ashwini");

System.out.println("Map2: "+map2);

map1.putAll(map2);

System.out.println("Map 1 after adding item from map2: "+map1);

map1.remove(3);

System.out.println("Map 1 after removing an element: "+map1);

boolean isPresent=map1.containsKey(4);

System.out.println("Is key 4 present in map1: "+isPresent);

String value=map1.get(2);

System.out.println("The value of key 2 is : "+value);

System.out.println("Printing all key and value of map1");

for(Map.Entry<Integer,String> entry:map1.entrySet()) {

System.out.println("Key:"+entry.getKey()+" Value:"+entry.getValue() );

}

}

}

Practical 5.1

Aim: Write a Java program using Lambda Expression to print ”Hello World”.

Code:

package pract5;

public class practical\_5\_1 {

public static void main(String[] args) {

// TODO Auto-generated method stub

Runnable helloWorld=()->System.out.println("Hello World");

helloWorld.run();

}

}

Practical 5.2

Aim: Write a Java program using Lambda Expression with single parameters.

Code:

package pract5;

import java.util.Arrays;

import java.util.List;

public class practical\_5\_2 {

public static void main(String[] args) {

// TODO Auto-generated method stub

List<String> names=Arrays.asList("XYZ", "ABC", "PQR","Raju");

names.forEach(name->System.out.println("Hello "+name+ " !"));

}

}

Practical 5.3

Aim: Write a Java program using Lambda Expression with multiple parameters to add two

numbers.

Code:

package pract5;

public class practical\_5\_3 {

public static void main(String[] args) {

// TODO Auto-generated method stub

A obj=(i,j)->i+j;

int result=obj.add(5,4);

System.out.println(result);

}

}

interface A{

int add (int i,int j);

}

Practical 5.4

Aim: Write a Java program using Lambda Expression to calculate the following:

a. Convert Fahrenheit to Celcius

b. Convert Kilometers to Miles.

Code:

package pract5;

public class prcatical\_5\_4 {

public static void main(String[] args) {

// TODO Auto-generated method stub

Converter c=f->(f-32)\*5/9;

double result1=c.convert(88);

System.out.println("88 dregree fahrenheit is "+ result1 +" Degeree Celsicus");

Converter k=km->km/1.609344;

double result2=k.convert(90);

System.out.println("90 kilometer is "+ result2 +" miles");

}

}

interface Converter{

double convert (double input);

}

Practical 5.5

Aim: Write a Java program using Lambda Expression with or without return keyword.

Code:

package pract5;

public class prcatical\_5\_5 {

public static void main(String[] args) {

// TODO Auto-generated method stub

Calculator obj1 = (i,j)->i+j;

int result=obj1.cal(5, 4);

System.out.println("Adding "+result);

Calculator obj2 = (i,j)->{return i+j;};

int result1=obj2.cal(7, 4);

System.out.println("Adding "+result1);

}

}

interface Calculator{

int cal (int x,int y);

}

Practical 5.6:

Aim: Write a Java program using Lambda Expression to concatenate two strings.

Code:

package pract5;

public class practcal\_5\_6 {

public static void main(String[] args) {

// TODO Auto-generated method stub

Concatentor c=(s1,s2)->s1+s2;

String result= c.con("Hello", " XYZ");

System.out.println(result);

}

}

interface Concatentor{

String con (String str1,String str2);

}

**Practical No. 6**

**(6.1)**

**Aim:** Create a Telephone directory using JSP and store all the information within a

database, so that later could be retrieved as per the requirement. Make your own assumptions.

**Code:  Display\_contact.jsp:**

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

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

<%@ page import=*"demo.DBConnection"* %>

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<meta name=*"viewport"* content=*"width=device-width, initial-scale=1"*>

<link href=*"https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"* rel=*"stylesheet"* integrity=*"sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"* crossorigin=*"anonymous"*>

<title>Display data</title>

</head>

<body>

<div class=*"container-fluid"*>

<h1 >Registered Users</h1>

<button class=*"btn btn-info"* type=*"button"*><a href=*"add\_contact.jsp"* >Add data</a></button><br><br>

<table class=*"table table-bordered"*>

<tr>

<th>User ID</th>

<th>name</th>

<th>phone</th>

<th>Update</th>

</tr>

<%

**try** {

// Establish a database connection

Connection connection = DBConnection.getConnection();

// Create and execute an SQL SELECT statement

String selectQuery = "SELECT \* FROM contacts";

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(selectQuery);

**while** (resultSet.next()) {

**int** userId = resultSet.getInt("id");

String username = resultSet.getString("name");

String phone = resultSet.getString("phone\_number");

%>

<tr>

<td><%= userId %></td>

<td><%= username %></td>

<td><%= phone %></td>

<td>

<a href=*"edit\_contact.jsp?id=*<%= userId %>*"*>Edit</a>

<a href=*"delete\_contact.jsp?id=*<%= userId %>*"*>Delete</a>

</td>

</tr>

<%

}

resultSet.close();

statement.close();

connection.close();

} **catch** (SQLException e) {

// Handle database errors here

out.println("Database error: " + e.getMessage());

}

%>

</table>

</div>

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"* integrity=*"sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL"* crossorigin=*"anonymous"*></script>

</body>

</html>

**Add\_contact.jsp:**

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

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

<%@ page import=*"demo.DBConnection"* %><%

Connection connection = DBConnection.getConnection();

**if** (request.getMethod().equals("POST")) {

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

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

**if** (name != **null** && phone != **null**) {

**try** {

// Insert the new contact into the database

String insertQuery = "INSERT INTO contacts (name, phone\_number) VALUES (?, ?)";

PreparedStatement preparedStatement = connection.prepareStatement(insertQuery);

preparedStatement.setString(1, name);

preparedStatement.setString(2, phone);

preparedStatement.executeUpdate();

preparedStatement.close();

} **catch** (SQLException e) {

e.printStackTrace();

}

}

}

// Close the database connection

**if** (connection != **null**) {

connection.close();

}

%>

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<link href=*"https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"* rel=*"stylesheet"* integrity=*"sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"* crossorigin=*"anonymous"*>

<title>Add new Contact</title>

</head>

<body>

<div class=*"container-fluid"* >

<h1>Add a New Contact</h1><br><br>

<form action=*"add\_contact.jsp"* method=*"post"*>

<label>Name: <input type=*"text"* name=*"name"*></label><br><br>

<label>Phone: <input type=*"text"* name=*"phone"*></label><br><br>

<input type=*"submit"* class=*"btn btn-primary"* value=*"Add Contact"*><br><br>

<a href=*"display\_contact.jsp"*>show data</a>

</form>

</div>

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"* integrity=*"sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL"* crossorigin=*"anonymous"*></script>

</body>

</html>

**Edit\_contact.jsp:**

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

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

<%@ page import=*"demo.DBConnection"* %>

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<link href=*"https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"* rel=*"stylesheet"* integrity=*"sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"* crossorigin=*"anonymous"*>

<title>Update Data</title>

</head>

<body>

<div class=*"container-fluid"*>

<h1>Edit Contact</h1>

<%

Connection connection = **null**;

PreparedStatement preparedStatement = **null**;

ResultSet resultSet = **null**;

**try** {

// Establish a database connection

connection = DBConnection.getConnection();

// Check if an ID parameter is provided in the URL

String idParam = request.getParameter("id");

**int** contactId = -1;

**if** (idParam != **null**) {

contactId = Integer.parseInt(idParam);

// Retrieve the contact's current details

String selectQuery = "SELECT \* FROM contacts WHERE id=?";

preparedStatement = connection.prepareStatement(selectQuery);

preparedStatement.setInt(1, contactId);

resultSet = preparedStatement.executeQuery();

**if** (resultSet.next()) {

String currentName = resultSet.getString("name");

String currentPhone = resultSet.getString("phone\_number");

%>

<form action=*"update\_contacts.jsp?id=*<%=contactId %>*"* method=*"post"*>

<input type=*"hidden"* name=*"id"* placeholder=*"*<%= contactId %>*"*>

Name: <input type=*"text"* name=*"name"* <%= currentName %>><br><br>

Phone: <input type=*"text"* name=*"phone"* <%= currentPhone %>><br><br>

<input type=*"submit"* class=*"btn btn-warning"*value=*"Update Contact"*>

</form>

<%

}

}

} **catch** (SQLException e) {

e.printStackTrace();

} **finally** {

// Close database resources individually

**if** (resultSet != **null**) resultSet.close();

**if** (preparedStatement != **null**) preparedStatement.close();

**if** (connection != **null**) connection.close();

}

%>

</div>

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"* integrity=*"sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL"* crossorigin=*"anonymous"*></script>

</body>

</html>

**Update\_contact.jsp:**

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

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

<%@ page import=*"demo.DBConnection"* %>

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Update contact</title>

</head>

<body>

<h1>Update Contact</h1>

<%

Connection connection = **null**;

PreparedStatement preparedStatement = **null**;

**try** {

// Establish a database connection

connection = DBConnection.getConnection();

// Get the data submitted from the form

**int** contactId = Integer.parseInt(request.getParameter("id"));

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

String newPhone = request.getParameter("phone");

// Update the contact's information in the database

String updateQuery = "UPDATE contacts SET name=?, phone\_number=? WHERE id=?";

preparedStatement = connection.prepareStatement(updateQuery);

preparedStatement.setString(1, newName);

preparedStatement.setString(2, newPhone);

preparedStatement.setInt(3, contactId);

preparedStatement.executeUpdate();

%>

<p>Contact updated successfully.</p>

<a href=*"display\_contact.jsp"*>Back to Contacts</a>

<%

} **catch** (SQLException e) {

e.printStackTrace();

%>

<p>An error occurred while updating the contact: <%= e.getMessage() %></p>

<%

} **finally** {

// Close database resources individually

**if** (preparedStatement != **null**) preparedStatement.close();

**if** (connection != **null**) connection.close();

}

%>

</body>

</html>

**Delete\_contact.jsp:**

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

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

<%@ page import=*"demo.DBConnection"* %>

<%

Connection connection = DBConnection.getConnection();

// Check if an ID parameter is provided in the URL

String idParam = request.getParameter("id");

**if** (idParam != **null**) {

**int** contactId = Integer.parseInt(idParam);

**try** {

// Delete the contact from the database

String deleteQuery = "DELETE FROM contacts WHERE id=?";

PreparedStatement preparedStatement = connection.prepareStatement(deleteQuery);

preparedStatement.setInt(1, contactId);

preparedStatement.executeUpdate();

preparedStatement.close();

} **catch** (SQLException e) {

e.printStackTrace();

}

}

// Close the database connection

**if** (connection != **null**) {

connection.close();

}

%>

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Delete data</title>

</head>

<body>

<h1>Contact Deleted</h1>

<p>The contact has been deleted successfully.</p>

<a href=*"display\_contact.jsp"*>Back to Contacts</a>

</body>

</html>

**(6.2)**

**Aim:** Write a JSP page to display the Registration form.

**Code: login.jsp:**

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

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

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Login Page</title>

</head>

<body>

<center>

<h1>Registration Form</h1>

<form action=*"databse.jsp"* method=*"post"*>

First name : <input type=*"text"* name=*"firstname"* required><br><br>

Last name : <input type=*"text"* name=*"lastname"* required><br><br>

Age : <input type=*"number"* name=*"age"* required><br><br>

Address : <input type=*"text"* name=*"address"* required><br><br>

Phone contact : <input type=*"number"* name=*"phone"* required><br><br>

<input type=*"submit"* name=*"sumbit"* value=*"Submit"* required><br><br>

</form>

</center>

</body>

</html>

**database.jsp:**

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

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

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Insert title here</title>

</head>

<body>

<%

**try** {

String connectionURL = "jdbc:mysql://localhost:3306/prcat6";

Connection connection = DriverManager.getConnection(connectionURL, "root", "root");

**if**(!connection.isClosed())

%>

<%

out.println("Successfully Register");

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

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

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

String address= request.getParameter("age");

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

String insertquery="Insert into mysqltest(firstname,lastname,age,address,phone) values(?,?,?,?,?)";

PreparedStatement prepared= connection.prepareStatement(insertquery);

prepared.setString(1,firstname);

prepared.setString(2,lastname);

prepared.setString(3,age);

prepared.setString(4,address);

prepared.setString(5,phone);

prepared.executeUpdate();

connection.close();

}

**catch**(Exception ex){

%>

<%

out.println("Unable to connect to database.");

}

%>

</body>

</html>

**(6.3)**

**Aim:**Write a JSP program to add, delete and display the records from StudentMaster

(RollNo, Name, Semester, Course) table.

**Code:**

**Dbcon.java:**

**package com.example;**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.SQLException;**

**public class Dbcon {**

**public static Connection getConnection() throws SQLException {**

**String url = "jdbc:mysql://localhost:3306/studentmaster";**

**String username = "root";**

**String password = "root";**

**Connection connection = null;**

**try {**

**connection = DriverManager.*getConnection*(url, username, password);**

**} catch (SQLException e) {**

**e.printStackTrace();**

**throw e; // You may want to handle this exception more gracefully in a real application**

**}**

**return connection;**

**}**

**}**

**AddStudent.jsp:**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

**<%@ page import=*"com.example.Dbcon"* %>**

**<%**

**Connection connection = Dbcon.getConnection();**

**boolean ia = false;**

**if (request.getMethod().equals("POST")) {**

**String RollNo =request.getParameter("RollNo");**

**String Name = request.getParameter("Name");**

**String Semester = request.getParameter("Semester");**

**String Course = request.getParameter("Course");**

**if (RollNo !=null && Name != null && Semester != null && Course != null) {**

**try {**

**// Insert the new contact into the database**

**String insertQuery = "INSERT INTO students (RollNo,Name,Semester, Course) VALUES (?, ?, ?,?)";**

**PreparedStatement preparedStatement = connection.prepareStatement(insertQuery);**

**preparedStatement.setString(1, RollNo);**

**preparedStatement.setString(2, Name);**

**preparedStatement.setString(3, Semester);**

**preparedStatement.setString(4, Course);**

**ia = preparedStatement.executeUpdate() > 0;**

**preparedStatement.close();**

**} catch (SQLException e) {**

**e.printStackTrace();**

**}**

**}**

**}**

**// Close the database connection**

**if (connection != null) {**

**connection.close();**

**}**

**%>**

**<html>**

**<body>**

**<h1>Add a New student</h1>**

**<form action=*"AddStudent.jsp"* method=*"post"*>**

**<label>RollNO: <input type=*"number"* name=*"RollNo"*></label><br><br>**

**<label>Name: <input type=*"text"* name=*"Name"*></label><br><br>**

**<label>Semester: <input type=*"text"* name=*"Semester"*></label><br><br>**

**<label>Course: <input type=*"text"* name=*"Course"*></label><br><br>**

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

**<a href=*"DisplayStudent.jsp"*>show data</a>**

**</form>**

**<div>**

**<% if (ia) { %>**

**<p>Data added successfully!</p>**

**<% } %>**

**</div>**

**</body>**

**</html>**

**DisplayStudent.jsp:**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

**<%@ page import=*"com.example.Dbcon"* %>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Display Student Info</title>**

**</head>**

**<body>**

**<h1>Registered Students </h1>**

**<table border=*"1"*>**

**<tr>**

**<th>User ID</th>**

**<th>RollNo</th>**

**<th>Name</th>**

**<th>Semester</th>**

**<th>Course</th>**

**<th>Update</th>**

**</tr>**

**<%**

**try {**

**// Establish a database connection**

**Connection connection = Dbcon.getConnection();**

**// Create and execute an SQL SELECT statement**

**String selectQuery = "SELECT \* FROM students";**

**Statement statement = connection.createStatement();**

**ResultSet resultSet = statement.executeQuery(selectQuery);**

**while (resultSet.next()) {**

**int userId = resultSet.getInt("id");**

**Integer RollNo=resultSet.getInt("RollNo");**

**String username = resultSet.getString("Name");**

**String semester = resultSet.getString("Semester");**

**String course = resultSet.getString("Course");**

**%>**

**<tr>**

**<td><%= userId %></td>**

**<td><%= RollNo %></td>**

**<td><%= username %></td>**

**<td><%= semester %></td>**

**<td><%= course %></td>**

**<td>**

**<a href=*"EditStudent.jsp?id=*<%= userId %>*"*>Edit</a>**

**<a href=*"Delete.jsp?id=*<%= userId %>*"*>Delete</a>**

**</td>**

**</tr>**

**<%**

**}**

**resultSet.close();**

**statement.close();**

**connection.close();**

**} catch (SQLException e) {**

**// Handle database errors here**

**out.println("Database error: " + e.getMessage());**

**}**

**%>**

**</table><br><br>**

**Back to register:<a href=*"AddStudent.jsp"*>link</a>**

**</body>**

**</html>**

**EditStudent.jsp:**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

**<%@ page import=*"com.example.Dbcon"* %>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"UTF-8"*>**

**<title>Edit Student Info</title>**

**</head>**

**<body>**

**<h1>Edit Student Info</h1>**

**<%**

**Connection connection = null;**

**PreparedStatement preparedStatement = null;**

**ResultSet resultSet = null;**

**try {**

**// Establish a database connection**

**connection = Dbcon.getConnection();**

**// Check if an ID parameter is provided in the URL**

**String idParam = request.getParameter("id");**

**int contactId = -1;**

**if (idParam != null) {**

**contactId = Integer.parseInt(idParam);**

**// Retrieve the contact's current details**

**String selectQuery = "SELECT \* FROM students WHERE id=?";**

**preparedStatement = connection.prepareStatement(selectQuery);**

**preparedStatement.setInt(1, contactId);**

**resultSet = preparedStatement.executeQuery();**

**if (resultSet.next()) {**

**Integer RollNo=resultSet.getInt("RollNo");**

**String username = resultSet.getString("Name");**

**String semester = resultSet.getString("Semester");**

**String course = resultSet.getString("Course");**

**%>**

**<form action=*"UpdateStudent.jsp"* method=*"post"*>**

**<input type=*"hidden"* name=*"id"* value=*"*<%= contactId %>*"*>**

**RollNo: <input type=*"number"* name=*"RollNo"* value=*"*<%= RollNo %>*"*><br><br>**

**Name: <input type=*"text"* name=*"Name"* value=*"*<%= username %>*"*><br><br>**

**Semester: <input type=*"text"* name=*"Semester"* value=*"*<%= semester %>*"*><br><br>**

**Course: <input type=*"text"* name=*"Course"* value=*"*<%= course %>*"*><br><br>**

**<input type=*"submit"* value=*"Update Student"*>**

**</form>**

**<%**

**}**

**}**

**} catch (SQLException e) {**

**e.printStackTrace();**

**} finally {**

**// Close database resources individually**

**if (resultSet != null) resultSet.close();**

**if (preparedStatement != null) preparedStatement.close();**

**if (connection != null) connection.close();**

**}**

**%>**

**</body>**

**</html>**

**UpdateStudent.jsp**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

**<%@ page import=*"com.example.Dbcon"* %>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Update Student Info</title>**

**</head>**

**<body>**

**<h1>Update Student Info</h1>**

**<%**

**Connection connection = null;**

**PreparedStatement preparedStatement = null;**

**try {**

**// Establish a database connection**

**connection = Dbcon.getConnection();**

**// Get the data submitted from the form**

**int contactId = Integer.parseInt(request.getParameter("id"));**

**String newRollNo =request.getParameter("RollNo");**

**String newName = request.getParameter("Name");**

**String newsemester = request.getParameter("Semester");**

**String newcourse = request.getParameter("Course");**

**// Update the contact's information in the database**

**String updateQuery = "UPDATE students SET RollNo=?,Name=?, Semester=?, Course=? WHERE id=?";**

**preparedStatement = connection.prepareStatement(updateQuery);**

**preparedStatement.setString(1, newRollNo);**

**preparedStatement.setString(2, newName);**

**preparedStatement.setString(3, newsemester);**

**preparedStatement.setString(4, newcourse);**

**preparedStatement.setInt(5, contactId);**

**preparedStatement.executeUpdate();**

**%>**

**<p>Student Info updated successfully.</p>**

**Back to Registered Students:<a href=*"DisplayStudent.jsp"*>link</a>**

**<%**

**} catch (SQLException e) {**

**e.printStackTrace();**

**%>**

**<p>An error occurred while updating the contact: <%= e.getMessage() %></p>**

**<%**

**} finally {**

**// Close database resources individually**

**if (preparedStatement != null) preparedStatement.close();**

**if (connection != null) connection.close();**

**}**

**%>**

**</body>**

**</html>**

**DeleteStudent.jsp:**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

**<%@ page import=*"com.example.Dbcon"* %>**

**<%**

**Connection connection = Dbcon.getConnection();**

**// Check if an ID parameter is provided in the URL**

**String idParam = request.getParameter("id");**

**if (idParam != null) {**

**int contactId = Integer.parseInt(idParam);**

**try {**

**// Delete the contact from the database**

**String deleteQuery = "DELETE FROM students WHERE id=?";**

**PreparedStatement preparedStatement = connection.prepareStatement(deleteQuery);**

**preparedStatement.setInt(1, contactId);**

**preparedStatement.executeUpdate();**

**preparedStatement.close();**

**} catch (SQLException e) {**

**e.printStackTrace();**

**}**

**}**

**// Close the database connection**

**if (connection != null) {**

**connection.close();**

**}**

**%>**

**<html>**

**<body>**

**<h1>Contact Deleted</h1>**

**<p>The Student data has been deleted successfully.</p>**

**Back to registered students:<a href=*"DisplayStudent.jsp"*>link</a>**

**</body>**

**</html>**

**(6.4)**

**Aim:** Design loan calculator using JSP which accepts Period of Time (in years) and

Principal Loan Amount. Display the payment amount for each loan and then list the

loan balance and interest paid for each payment over the term of the loan for the

following time period and interest rate:

a. 1 to 7 year at 5.35%

b. 8 to 15 year at 5.5%

c. 16 to 30 year at 5.75%.

**Code:**

<!DOCTYPE html>

<html>

<head>

<title>Simple Interest Calculator</title>

</head>

<body>

<h1>Simple Interest Calculator</h1>

<form>

Principal Amount: <input type=*"text"* id=*"principal"* required><br>

Rate of Interest (per annum): <input type=*"text"* id=*"rate"* required><br>

Time (in years): <input type=*"text"* id=*"time"* required><br>

<input type=*"button"* value=*"Calculate"* onclick="calculateSimpleInterest()">

</form>

<p id=*"result"*></p>

<script>

**function** calculateSimpleInterest() {

**var** principal = parseFloat(document.getElementById("principal").value);

**var** rate = parseFloat(document.getElementById("rate").value);

**var** time = parseFloat(document.getElementById("time").value);

**var** simpleInterest = (principal \* rate \* time) / 100;

// Display the result in a traditional way

**var** resultMessage = "Simple Interest: " + simpleInterest;

**var** resultElement = document.getElementById("result");

resultElement.innerHTML = resultMessage;

}

</script>

</body>

</html>

**(6.5)**

**Aim:** Write a program using JSP that displays a webpage consisting Application form for

change of Study Center which can be filled by any student who wants to change his/

her study center. Make necessary assumptions

**Code:**  
<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Study Center</title>

</head>

<body>

<center>

<h1>Change of Study Center</h1>

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

<table>

<tr>

<td> UId No.</td>

<td><input type=*"text"* name=*"uid"* required/> </td>

</tr>

<tr>

<td> Current Center</td>

<td>

<select name=*"currentCenter"* required>

<option selected disabled hidden/>

<option value=*"Mumbai"*>Mumbai</option>

<option value=*"Pune"*>Pune</option>

<option value=*"Gujrat"*>Gujrat</option>

</select>

</td>

</tr>

<tr>

<td>New Center</td>

<td>

<select name=*"newCenter"* required>

<option selected disabled hidden/>

<option value=*"Mumbai"*>Mumbai</option>

<option value=*"Pune"*>Pune</option>

<option value=*"Gujrat"*>Gujrat</option>

</select>

</td>

</tr>

</table>

<input type=*"submit"* valu=*"Submit"*/>

</form>

</center>

<%**if**(request.getParameter("uid")!=**null** &&

request.getParameter("currentCenter")!=**null** &&

request.getParameter("newCenter")!=**null**){

out.println("<center> <br> Your request to change Study Center from <br>"

+request.getParameter("currentCenter")+" to "+

request.getParameter("newCenter")+"<br> has been seen to the Administrator.</center>");}

%>

</body>

</html>

**(6.6)**

**Aim:** Write a JSP program that demonstrates the use of JSP declaration, scriptlet,

directives, expression, header and footer.

**Code: Main.jsp:**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Jsp Declaration, Spriptlet,directive,expression,header and footer example</title>

</head>

<body>

<%@ include file="header.jsp" %>

<center>

<%! **int** data=50; %>

<%= "value of the variable is: "+data %>

<%! **double** circle(**int** n){**return** 3.14\*n\*n;}%>

<br>

<%= "Area of Circle is: "+circle(5) %>

<br>

<%! **int** rectangle (**int** l, **int** b){**return** l\*b;}%>

<%="Area of rectangle is: "+rectangle(4,6) %>

<br>

<%! **int** perimeter(**int** x,**int** y)

{

**int** peri=2\*(x+y);

**return** peri;

}%>

<br>

<%= "Perimeter of rectangle: "+ perimeter(5,6)%>

<br>

<p> Thanks for visting my page</p>

</center>

<%@ include file="footer.jsp" %>

</body>

</html>

**Header.jsp:**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<%! **int** pageCount=0;

**void** addcount(){pageCount++;}%>

<% addcount();%>

<title>Jsp Declaration, Spriptlet,directive,expression,header and footer example</title>

</head>

<body>

<center>

<h2>

The include Directive Example

</h2>

<p><b>This site been visited <%= pageCount %>.</b></p>

</center>

</body>

</html>

**Footer.jsp:**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<center>

<b>Suraj Gupta </b><br>

<b>23-Mca-25</b>

</center>

</body>

</html>

**Practical No.7**

**(7.1)**

**Aim:**Write a program to print “Hello World” using spring framework.

**Code:**

**Prac71new2Application.java:**

package com.example.demo;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

@SpringBootApplication

public class Prac71new2Application {

public static void main(String[] args) {

try (AnnotationConfigApplicationContext context = new AnnotationConfigApplicationContext(HelloWorldConfig.class)) {

            HelloWorldService helloWorldService = context.getBean(HelloWorldService.class);

            helloWorldService.sayHello();

        }

}

}

**HelloWorldConfig.java:**

package com.example.demo;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class HelloWorldConfig {

   @Bean

   public HelloWorldService helloWorldService() {

       return new HelloWorldService();

   }

}

**HelloWorldService.java:**

package com.example.demo;

public class HelloWorldService {

   public void sayHello() {

       System.*out*.println("Hello World");

   }

}

**(7.2)**

**Aim:** Write a program to demonstrate dependency injection via setter method

**Code:**

**Helloapplication.java:**

package com.example.demo;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication

@ComponentScan(basePackages = "com.example.demo")

public class Hello1Application {

public static void main(String[] args) {

SpringApplication.run(Hello1Application.class, args);

}

@Bean

public CommandLineRunner demo(Car car) {

        return args -> {

            // Start the Car

         car.setModel("Honda");

         Driver dr=new Driver();

         dr.setName("Manoranjan");

         car.setDriver(dr);

            car.start();

        };

    }

}

@RestController

class HelloController {

    private final Car car;

    public HelloController(Car car) {

        this.car = car;

    }

    @GetMapping("/hello")

    public String hello() {

        car.start();

        return car.getModel();

    }

}

**Driver.java:**

package com.example.demo;

import org.springframework.stereotype.Component;

@Component

public class Driver {

   private String name;

   public String getName() {

       return name;

   }

   public void setName(String name) {

       this.name = name;

   }

}

**Car.java:**

package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

@Component

public class Car {

   private String model;

   private Driver driver;

   public String getModel() {

       return model;

   }

   public void setModel(String model) {

       this.model = model;

   }

   public Driver getDriver() {

       return driver;

   }

   @Autowired

   public void setDriver(Driver driver) {

       this.driver = driver;

   }

   public void start() {

       System.*out*.println("Car model: " + model);

       System.*out*.println("Driver: " + driver.getName());

       System.*out*.println("Car is starting...");

   }

}

**(7.3)**

**Aim:** Write a program to demonstrate dependency injection via Constructor

**Code:**

**Helloapplication.java:**

package com.example.demo;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication

@ComponentScan(basePackages = "com.example.demo")

public class Hello1Application {

public static void main(String[] args) {

SpringApplication.run(Hello1Application.class, args);

}

@Bean

public CommandLineRunner demo(Car car) {

        return args -> {

            // Start the Car

         car.setModel("Honda");

         Driver dr=new Driver();

         dr.setName("Manoranjan");

         car.setDriver(dr);

            car.start();

        };

    }

}

@RestController

class HelloController {

    private final Car car;

    public HelloController(Car car) {

        this.car = car;

    }

    @GetMapping("/hello")

    public String hello() {

        car.start();

        return car.getModel();

    }

}

**Driver.java:**

package com.example.demo;

import org.springframework.stereotype.Component;

@Component

public class Driver {

   private String name;

   public String getName() {

       return name;

   }

   public void setName(String name) {

       this.name = name;

   }

}

**Car.java:**

package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

@Component

public class Car {

   private String model;

   private Driver driver;

   public String getModel() {

       return model;

   }

   public void setModel(String model) {

       this.model = model;

   }

   public Driver getDriver() {

       return driver;

   }

   @Autowired

   public void setDriver(Driver driver) {

       this.driver = driver;

   }

   public void start() {

       System.*out*.println("Car model: " + model);

       System.*out*.println("Driver: " + driver.getName());

       System.*out*.println("Car is starting...");

   }

}

**Practical No.8**

**(8.1)**

**Aim:** Write a program to demonstrate Spring AOP – before advice.

**Code:** **MainApp:**

**package** com.example;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.ConfigurableApplicationContext;

@SpringBootApplication

**public** **class** MainApp {

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

// **TODO** Auto-generated method stub

ConfigurableApplicationContext context= SpringApplication.*run*(MainApp.**class**, args);

MyService service= context.getBean(MyService.**class**);

service.show();

}

}

**MyServices.class:**

**package** com.example;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** MyService {

**void** show() {

System.***out***.println("My Services Class.");

}

}

**MyAspect.class:**

**package** com.example;

**import** org.aspectj.lang.annotation.Aspect;

**import** org.aspectj.lang.annotation.Before;

**import** org.springframework.stereotype.Component;

@Aspect

@Component

**public** **class** MyAspect {

@Before("execution(\* com.example.MyService.\*(..))")

**void** Logging() {

System.***out***.println("Logging Before My Service Class.");

}

}

**(8.2)**

**Aim:Write a program to demonstrate Spring AOP – after advice.**

**Code:** **MainApp:**

**package** com.example;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.ConfigurableApplicationContext;

@SpringBootApplication

**public** **class** MainApp {

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

// **TODO** Auto-generated method stub

ConfigurableApplicationContext context= SpringApplication.*run*(MainApp.**class**, args);

MyService service= context.getBean(MyService.**class**);

service.show();

}

}

**MyServices.class:**

**package** com.example;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** MyService {

**void** show() {

System.***out***.println("My Services Class.");

}

}

**MyAspect:**

**package** com.example;

**import** org.aspectj.lang.annotation.After;

**import** org.aspectj.lang.annotation.Aspect;

**import** org.springframework.stereotype.Component;

@Aspect

@Component

**public** **class** MyAspect {

@After("execution(\* com.example.MyService.\*(..))")

**void** AfterLogging() {

System.***out***.println("Logging After My Services Class.      ");

}

}

**(8.3)**

**Aim:Write a program to demonstrate Spring AOP – around advice.**

**Code:** **MainApp:**

**package** com.example;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.ConfigurableApplicationContext;

@SpringBootApplication

**public** **class** MainApp {

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

// **TODO** Auto-generated method stub

ConfigurableApplicationContext context= SpringApplication.*run*(MainApp.**class**, args);

MyService service= context.getBean(MyService.**class**);

service.show();

}

}

**MyServices.class:**

**package** com.example;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** MyService {

**void** show() {

System.***out***.println("My Services Class.");

}

}

**MyAspect:**

**package** com.example;

**import** org.aspectj.lang.ProceedingJoinPoint;

**import** org.aspectj.lang.annotation.Around;

**import** org.aspectj.lang.annotation.Aspect;

**import** org.springframework.stereotype.Component;

@Aspect

@Component

**public** **class** MyAspect {

@Around("execution(\* com.example.MyService.\*(..))")

**public** Object aroundServiceMethod(ProceedingJoinPoint joinPoint) **throws** Throwable {

        // Code to be executed before the method call

        System.***out***.println("Before method execution: " + joinPoint.getSignature().toShortString());

        // Proceed with the actual method invocation

        Object result = joinPoint.proceed();

        // Code to be executed after the method call

        System.***out***.println("After method execution: " + joinPoint.getSignature().toShortString());

        // Return the result of the method call

**return** result;

    }

}

**(8.4)**

**Aim:Write a program to demonstrate Spring AOP – after returning advice.**

**Code:** **MainApp:**

**package** com.example;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.ConfigurableApplicationContext;

@SpringBootApplication

**public** **class** MainApp {

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

// **TODO** Auto-generated method stub

ConfigurableApplicationContext context= SpringApplication.*run*(MainApp.**class**, args);

MyService service= context.getBean(MyService.**class**);

service.show();

}

}

**MyServices.class:**

**package** com.example;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** MyService {

**void** show() {

System.***out***.println("My Services Class.");

}

}

**MyAspect:**

**package** com.example;

**import** org.aspectj.lang.annotation.AfterReturning;

**import** org.aspectj.lang.annotation.Aspect;

**import** org.springframework.stereotype.Component;

@Aspect

@Component

**public** **class** MyAspect {

@AfterReturning(pointcut = "execution(\* com.example.MyService.\*(..))", returning = "result")

**public** **void** afterReturningServiceMethod(Object result) {

        System.***out***.println("After method execution: Returning result - " + result);

    }

}

**(8.5)**

**Aim:Write a program to demonstrate Spring AOP – after throwing advice.**

**Code:**

**MainApp:**

**package** com.example;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.ConfigurableApplicationContext;

@SpringBootApplication

**public** **class** MainApp {

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

// **TODO** Auto-generated method stub

ConfigurableApplicationContext context=SpringApplication.*run*(MainApp.**class**);

MyService services= context.getBean(MyService.**class**);

**try** {

services.show();

} **catch** (Exception e) {

// **TODO**: handle exception

}

}

}

**MyService**

**package** com.example;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** MyService {

**public** **void** show() {

**throw** **new** RuntimeException("Simulation Exception in MyService");

}

}

**MyAspect**

**package** com.example;

**import** org.aspectj.lang.annotation.AfterThrowing;

**import** org.aspectj.lang.annotation.Aspect;

**import** org.springframework.stereotype.Component;

@Aspect

@Component

**public** **class** MyAspect {

@AfterThrowing(pointcut = "execution(\* com.example.MyService.show())",throwing= "exception")

**public** **void** afterThrowingAdvice(Exception exception) {

System.***out***.println("After Throwing Advice: Exception caught - "+exception.getMessage());

}

}

**(8.6)**

**Aim:Write a program to demonstrate Spring AOP – pointcuts.**

**Code:**

**MainApp**

**package** com.example;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.ConfigurableApplicationContext;

@SpringBootApplication

**public** **class** MainApp {

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

// **TODO** Auto-generated method stub

ConfigurableApplicationContext context=SpringApplication.*run*(MainApp.**class**);

MyService services= context.getBean(MyService.**class**);

services.method1();

services.method2();

}

}

**MyServices**

**package** com.example;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** MyService {

**public** **void** method1() {

System.***out***.println("Executing method1");

}

**public** **void** method2() {

System.***out***.println("Executing method2");

}

}

**MyAspect**

**package** com.example;

**import** org.aspectj.lang.annotation.AfterThrowing;

**import** org.aspectj.lang.annotation.Aspect;

**import** org.aspectj.lang.annotation.Before;

**import** org.aspectj.lang.annotation.Pointcut;

**import** org.springframework.stereotype.Component;

@Aspect

@Component

**public** **class** MyAspect {

@Pointcut("execution(\* com.example.MyService.method1())")

**public** **void** pointcutMethod1() {

}

@Pointcut("execution(\* com.example.MyService.method2())")

**public** **void** pointcutMethod2() {

}

@Before("pointcutMethod1()")

**public** **void** beforeMethod1() {

System.***out***.println("Before advice for method1");

}

@Before("pointcutMethod2()")

**public** **void** beforeMethod2() {

System.***out***.println("Before advice for method2");

}

}

**Practical No. 10**

**(10.1)**

**Aim:Write a program to create a simple Spring Boot application that prints a message.**

**Code:**

**WebApplication.class**

package com.example.demo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class WebApplication {

public static void main(String[] args) {

SpringApplication.run(WebApplication.class, args);

}

}

**helloController.java**

**package** com.example.demo;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.RestController;

@RestController

**public** **class** helloController {

@GetMapping(path="/hello")

**public** String HelloWorld() {

**return** "Hello World";

}

}

**(10.2)**

**Aim:Write a program to demonstrate RESTful Web Services with spring boot.**

**Code:**

**WebApplication:**

package com.example.demo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class WebApplication {

public static void main(String[] args) {

SpringApplication.run(WebApplication.class, args);

}

}

**HelloController.class**

**package** com.example.demo;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.RestController;

@RestController

**public** **class** helloController {

@GetMapping(path="/hello")

**public** String HelloWorld() {

**return** "Hello World";

}

@GetMapping(path="/hello-world-bean")

**public** helloworldbean helloWorldBean() {

**return** **new** helloworldbean("hey");

}

}

Helloworldbean.class

**package** com.example.demo;

**public** **class** helloworldbean {

**private** String message;

**public** String getMessage() {

**return** message;

}

**public** **void** setMessage(String message) {

**this**.message = message;

}

**public** helloworldbean(String message) {

**super**();

**this**.message = message;

}

@Override

**public** String toString() {

**return** "helloworldbean [getMessage()=" + getMessage() + "]";

}

}

**Practical 9**

**Assignment based Spring JDBC**

**9.1**

**Aim:Write a program to insert, update and delete records from the given table.**

**Code:**

**Pract9Application.java**

package com.example;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.Bean;

@SpringBootApplication

public class Pract91Application {

public static void main(String[] args) {

SpringApplication.run(Pract91Application.class, args);

}

@Bean

    CommandLineRunner demo(operationjdbc recordService) {

return args -> {

// // Insert a record

recordService.insertRecord("Raju", 25,1);

// Update the record

recordService.updateRecord("Raju Gupta", 26,1);

//

// // Delete the record

recordService.deleteRecord(1);

};

}

}

**AppConfig.java**

package com.example;

import javax.sql.DataSource;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.jdbc.datasource.DriverManagerDataSource;

@Configuration

public class AppConfig {

@Bean

DataSource dataSource() {

DriverManagerDataSource dataSource = new DriverManagerDataSource();

dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");

dataSource.setUrl("jdbc:mysql://localhost:3306/emp");

dataSource.setUsername("root");

dataSource.setPassword("root");

return dataSource;

}

@Bean

JdbcTemplate jdbcTemplate(DataSource dataSource) {

return new JdbcTemplate(dataSource);

}

}

**Operation.java**

package com.example;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.stereotype.Service;

@Service

public class operationjdbc {

@Autowired

private JdbcTemplate jdbcTemplate;

public void insertRecord(String name, int age ,int id) {

String sql = "INSERT INTO emp (name, age, id) VALUES (?, ?, ?)";

jdbcTemplate.update(sql, name, age,id);

}

public void updateRecord( String name, int age ,int id) {

String sql = "UPDATE emp SET name = ?, age = ? where id=?";

jdbcTemplate.update(sql, name, age,id);

}

public void deleteRecord(int id) {

String sql = "DELETE FROM emp WHERE id = ?";

jdbcTemplate.update(sql, id);

}

}

**9.2**

**Aim:Write a program to demonstrate PreparedStatement in Spring JdbcTemplate**

**Code:**

**Pract92Application.java:**

package com.example;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.Bean;

@SpringBootApplication

public class Pract92Application {

public static void main(String[] args) {

SpringApplication.run(Pract92Application.class, args);

}

@Bean

CommandLineRunner demo(prepared recordService) {

return args ->{

// Insert a record

recordService.insertRecord("Raju", 24,1);

};

}

}

**AppConfig.java:**

**package** com.example;

**import** javax.sql.DataSource;

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.datasource.DriverManagerDataSource;

@Configuration

**public** **class** appconfig {

@Bean

DataSource dataSource() {

DriverManagerDataSource dataSource = **new** DriverManagerDataSource();

dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");

dataSource.setUrl("jdbc:mysql://localhost:3306/emp");

dataSource.setUsername("root");

dataSource.setPassword("root");

**return** dataSource;

}

@Bean

JdbcTemplate jdbcTemplate(DataSource dataSource) {

**return** **new** JdbcTemplate(dataSource);

}

}

**Prepared.java:**

**package** com.example;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.PreparedStatementCreator;

**import** org.springframework.stereotype.Service;

**import** java.sql.Connection;

**import** java.sql.PreparedStatement;

**import** java.sql.SQLException;

@Service

**public** **class** prepared {

@Autowired

**private** JdbcTemplate jdbcTemplate;

**public** **void** insertRecord(String name, **int** age, **int** id) {

**final** String sql = "INSERT INTO emp (name, age, id) VALUES (?, ?, ?)";

jdbcTemplate.update(

**new** PreparedStatementCreator() {

@Override

**public** PreparedStatement createPreparedStatement(Connection

connection) **throws** SQLException {

PreparedStatement ps = connection.prepareStatement(sql);

ps.setString(1, name);

ps.setInt(2, age);

ps.setInt(3, id);

**return** ps;

}

}

);

System.***out***.println("Inserted record with id: " + id);

}

}

**9.3**

**Aim:Write a program in Spring JDBC to demonstrate ResultSetExtractor Interface.**

**Code:**

**Pract93Application.java:**

**package** com.example;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.boot.CommandLineRunner;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.annotation.Bean;

**import** java.util.List;

@SpringBootApplication

**public** **class** Pract93Application **implements** CommandLineRunner {

@Autowired

**private** ResultSetExtraction resultSetExtraction;

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

SpringApplication.*run*(Pract93Application.**class**, args);

}

@Override

**public** **void** run(String... args) **throws** Exception {

// Get all employees

List<ResultSetExtraction.Employee> employees =resultSetExtraction.getAllEmployees();

// Print the result

**for** (ResultSetExtraction.Employee employee : employees) {

System.***out***.println("Employee ID:" + employee.getId() +

", Name: " + employee.getName() +

", Age:" + employee.getAge());

}

}

}

**AppConfig.java:**

**package** com.example;

**import** javax.sql.DataSource;

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.datasource.DriverManagerDataSource;

@Configuration

**public** **class** appconfig {

@Bean

DataSource dataSource() {

DriverManagerDataSource dataSource = **new** DriverManagerDataSource();

dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");

dataSource.setUrl("jdbc:mysql://localhost:3306/emp");

dataSource.setUsername("root");

dataSource.setPassword("root");

**return** dataSource;

}

@Bean

JdbcTemplate jdbcTemplate(DataSource dataSource) {

**return** **new** JdbcTemplate(dataSource);

}

}

**ResultSetExtraction.java:**

**package** com.example;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.ResultSetExtractor;

**import** org.springframework.stereotype.Service;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.ArrayList;

**import** java.util.List;

@Service

**public** **class** ResultSetExtraction {

@Autowired

**private** JdbcTemplate jdbcTemplate;

**public** List<Employee> getAllEmployees() {

String sql = "SELECT id, name, age FROM emp";

**return** jdbcTemplate.query(sql, **new** EmployeeResultSetExtractor());

}

// Employee class for representing the result set

**public** **static** **class** Employee {

**private** Long id;

**public** **void** setId(Long id) {

**this**.id = id;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

**private** String name;

**private** **int** age;

// Constructors, getters, and setters

// Example: Getter methods

**public** Long getId() {

**return** id;

}

**public** String getName() {

**return** name;

}

**public** **int** getAge() {

**return** age;

}

}

// ResultSetExtractor implementation

**private** **static** **class** EmployeeResultSetExtractor **implements**

ResultSetExtractor<List<Employee>> {

@Override

**public** List<Employee> extractData(ResultSet resultSet) **throws** SQLException

{

List<Employee> employees = **new** ArrayList<>();

**while** (resultSet.next()) {

Employee employee = **new** Employee();

employee.setId(resultSet.getLong("id"));

employee.setName(resultSet.getString("name"));

employee.setAge(resultSet.getInt("age"));

employees.add(employee);

}

**return** employees;

}

}

}

**9.4**

**Aim: Write a program to demonstrate RowMapper interface to fetch the records from the**

**Database.**

**Code:**

**Pract94Application.java:**

**package** com.example;

**import** java.util.List;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.boot.CommandLineRunner;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

**public** **class** Pract94Application **implements** CommandLineRunner{

@Autowired

**private** Rowmapper rowMapperDemo;

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

SpringApplication.*run*(Pract94Application.**class**, args);

}

@Override

**public** **void** run(String... args) **throws** Exception {

// Get all employees

List<Rowmapper.Employee> employees = rowMapperDemo.getAllEmployees();

// Print the result

**for** (Rowmapper.Employee employee : employees) {

System.***out***.println("Employee ID: " + employee.getId() +

", Name: " + employee.getName() +

", Age: " + employee.getAge());

}

}

}

**AppConfig.java:**

**package** com.example;

**import** javax.sql.DataSource;

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.datasource.DriverManagerDataSource;

@Configuration

**public** **class** AppConfig {

@Bean

DataSource dataSource() {

DriverManagerDataSource dataSource = **new** DriverManagerDataSource();

dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");

dataSource.setUrl("jdbc:mysql://localhost:3306/emp");

dataSource.setUsername("root");

dataSource.setPassword("root");

**return** dataSource;

}

@Bean

JdbcTemplate jdbcTemplate(DataSource dataSource) {

**return** **new** JdbcTemplate(dataSource);

}

}

**Rowmapper.java:**

**package** com.example;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.RowMapper;

**import** org.springframework.stereotype.Service;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.List;

@Service

**public** **class** Rowmapper {

@Autowired

**private** JdbcTemplate jdbcTemplate;

**public** List<Employee> getAllEmployees() {

String sql = "SELECT id, name, age FROM emp";

**return** jdbcTemplate.query(sql, **new** EmployeeRowMapper());

}

// Employee class for representing the result set

**public** **static** **class** Employee {

**private** Long id;

**public** **void** setId(Long id) {

**this**.id = id;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

**private** String name;

**private** **int** age;

**public** Long getId() {

**return** id;

}

**public** String getName() {

**return** name;

}

**public** **int** getAge() {

**return** age;

}

}

// RowMapper implementation

**private** **static** **class** EmployeeRowMapper **implements** RowMapper<Employee> {

@Override

**public** Employee mapRow(ResultSet resultSet, **int** rowNum) **throws**

SQLException {

Employee employee = **new** Employee();

employee.setId(resultSet.getLong("id"));

employee.setName(resultSet.getString("name"));

employee.setAge(resultSet.getInt("age"));

**return** employee;

}

}

}