# **PRACTICAL NO-1**

**Aim: Assignments on Java Generics.**

**1. Write a Java Program to demonstrate a Generic Class.**

**Code:**

package generics;

class Main1<T> {

T obj;

void add(T obj) {

this.obj=obj;

}

T get() {

return obj;

}

}

public class generics{

public static void main(String[] args) {

Main1<Integer> nn =new Main1<Integer>();

nn.add(100);

Main1<String> ss =new Main1<String>();

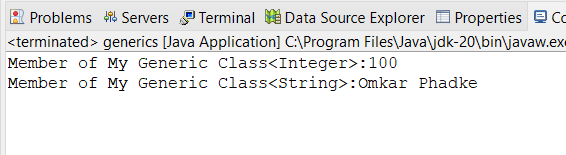
ss.add("Omkar Phadke");

System.out.println("Member of My Generic Class<Integer>:"+nn.get());

System.out.println("Member of My Generic Class<String>:"+ss.get());

}}

**OUTPUT:**

****

**2. Write a Java Program to demonstrate Generic Methods.**

**Code:**

package generics;

public class GenericMethod {

public static <T>void printGenericArray(T[] items) {

for(T item : items) {

System.*out*.print(item+"");

}

System.*out*.println();

}

public static void main(String[] args) {

Integer[] int\_Array= {1,2,4,5,6,11};

Double[] doubleArray = { 1.1, 2.2, 3.3, 4.4 };

Character[] char\_Array = {'O','M','K','A','R'};

System.*out*.println("Integer Array contents: ");

*printGenericArray*(int\_Array);

System.*out*.println("\nArray doubleArray contains:");

*printGenericArray*(doubleArray);

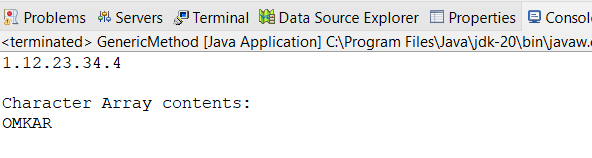
System.*out*.println("\nCharacter Array contents: ");

*printGenericArray*(char\_Array );

}

}

**Output:**

****

**3. Write a Java Program to demonstrate Wildcards in Java Generics.**

**Code:**

package wildcard;

import java.util.\*;

public class Wildcard {

public static void printList(List<?> list) {

for (Object elem : list)

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

System.out.println();

}

// method that sums the elements of a list of numbers

public static double sumOfList(List<? extends Number> list) {

double s = 0.0;

for (Number n : list)

s += n.doubleValue();

return s;

}

public static void main(String[] args) {

List<Integer> integerList = Arrays.asList(1, 2, 3, 4, 5);

System.out.println("integerList contains: ");

printList(integerList);

System.out.println("The sum is: " + sumOfList(integerList));

List<Double> doubleList = Arrays.asList(1.1, 2.2, 3.3, 4.4, 5.5);

System.out.println("\ndoubleList contains: ");

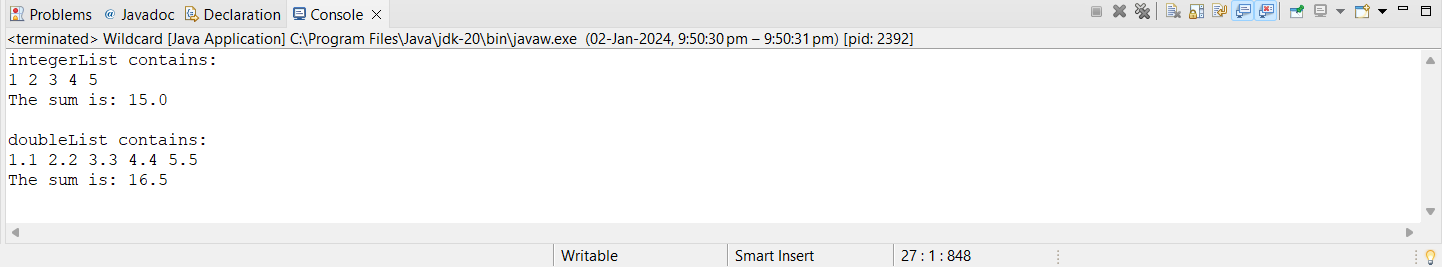
printList(doubleList);

System.out.println("The sum is: " + sumOfList(doubleList));

}

}

**Output:**



# **PRACTICAL NO-2**

**Aim: Assignments on List Interface.**

**1. Write a Java program to create List containing list of items of type String and use for- --each loop to print the items of the list.**

**Code:**

package listinterface;

import java.util.\*;

public class Listinterface {

public static void main(String args[]) {

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

list.add("Hello");

list.add("Omkar");

list.add("How Are You?");

System.out.println("ArrayList:" + list);

//System.out.println("root is at index: " + list.indexOf(10));

for(String s:list){

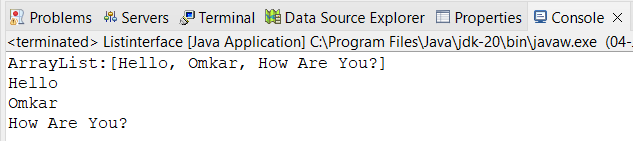
System.out.println(s);

}

}

}

**Output:**



**2. Write a Java program to create List containing list of items and use ListIterator interface to print items present in the list. Also print the list in reverse/ backword direction.**

**Code:**

package listinterface;

import java.util.ArrayList;

import java.util.Iterator;

public class ListInterface2 {

public static void main(String[] args) {

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

arr.add("Hello");

arr.add("Omkar");

arr.add("How Are You");

System.out.println(arr);

Iterator<String> it= arr.iterator();

for (int i = arr.size()-1; i >=0; i--) {

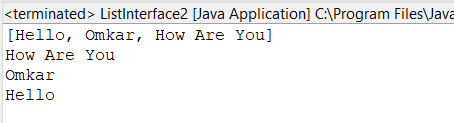
System.out.println(arr.get(i));

}

}

}

**Output:**

****

# **PRACTICAL NO-3**

**Aim:** **Assignments on Set Interface.**

**1. 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 in to other set.**

**c. Remove items from the set**

**d. Search the specified item in the set**

**Code:**

package setinterface;

import java.util.\*;

public class Setinterface {

public static void main(String[] args) {

Set<String> items = new HashSet<>(Arrays.asList("Nishant", "Ashish",

"Anirudh", "Manav", "Shreyas","Omkar","Harsh","Sen","Siddharth"));

// Get an Iterator for the set

Iterator<String> iterator = items.iterator();

System.out.println("Printing items in the set:");

while (iterator.hasNext()) {

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

}

// Convert the Set to a List

List<String> itemList = new ArrayList<>(items);

// Reverse the list

Collections.reverse(itemList);

System.out.println("\nPrinting items in reverse order:");

for (String item : itemList) {

System.out.println(item);

}

}

}

**Output:**

# 

# **B. Write a Java program using Set interface containing list of items and perform the following operations:**

* 1. **Add items in the set.**
  2. **Insert items of one set in to other set.**
  3. **Remove items from the set**
  4. **Search the specified item in the set**

**Code:**

package setinterface;

import java.util.\*;

public class Setinterface1 {

public static void main(String[] args) {

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

//Add Function

hset.add("Apple");

hset.add("Mango");

hset.add("Banana");

hset.add("Orange");

hset.add("Grapes");

System.out.println("After Using Add Function: "+hset);

//Remove Function

hset.remove("Mango");

System.out.println("After Using Remove Function: "+hset);

String check="Grapes";

System.out.println(hset);

System.out.println("Contain "+" "+ check+" "+"is "+hset.contains(check)+".");

Set<String> hset1 = new HashSet<String>();

hset1.addAll(Arrays.asList(new String[] { "ABC", "XYZ", "MNO", "EFG" }));

Set<String> hset2 = new HashSet<String>();

hset2.addAll(Arrays.asList(new String[] { "KLM", "HIJ", "PQR", "STY" }));

Set<String> union = new HashSet<String>(hset1);

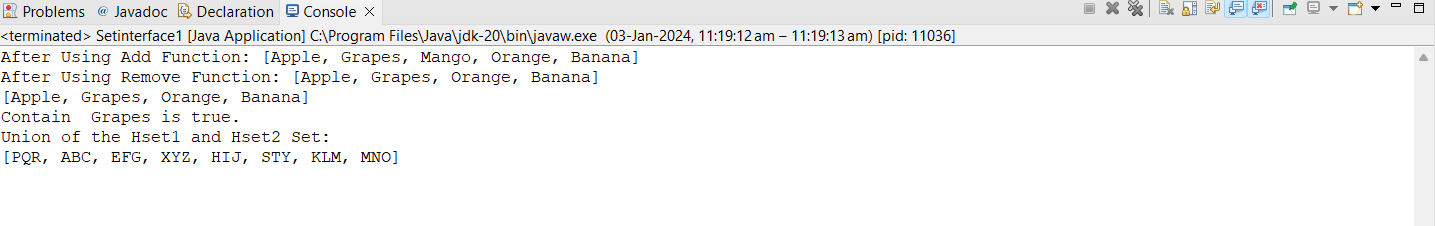
union.addAll(hset2);

System.out.println("Union of the Hset1 and Hset2 Set: ");

System.out.println(union);

}

}

**Output:**

# **PRACTICAL NO-4**

**Aim: Assignments on Map Interface**

**Write a Java program using Map interface containing list of items having keys and 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 in to other map.**

**f. Print all keys and values of the map**

**Code:**

package mapinterface;

import java.util.\*;

public class MapInterface {

public static void main(String[] args) {

// a. Add items in the map

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

map1.put("Omkar", "Phadke");

map1.put("Satyam", "yadav");

map1.put("Yash", "Parab");

System.out.println("A]\tMap1: " + map1);

// b. Remove items from the map

map1.remove("Yash");

System.out.println("B]\tAfter removing 'Yash' from map1, map1: " +

map1);

// c. Search specific key from the map

String key = "SHREYAS";

if (map1.containsKey(key)) {

System.out.println("C]\t'" + key + "' is found in the map.");

} else {

System.out.println("C]\t'" + key + "' is not found in the map.");

}

// d. Get value of the specified key

String value = map1.get(key);

System.out.println("D]\tThe value of '" + key + "' is: " + value);

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

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

map2.put("OMKAR", "PHADKE");

map2.put("SEN", "SABU");

map1.putAll(map2);

System.out.println("E]\tAfter adding map2 into map1, map1: " + map1);

// f. Print all keys and values of the map

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

System.out.println("F]\tKey: " + entry.getKey() + ", Value: " +

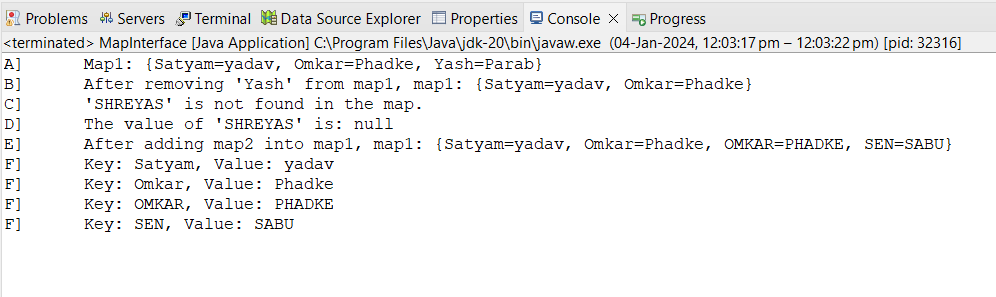
entry.getValue());

}

}

}

**Output:**

****

# **PRACTICAL NO-5**

**Aim:** **Assignments on Lambda Expression.**

**1. Write a Java program using Lambda Expression to print ”Hello World”.**

**Code:**

package lambdaExpression;

//Q1. Write a java program to execute Lambda Expression without parameter.

public class LambdaExpression {

interface Hello {

void hello();

}

public static void main(String[] args) {

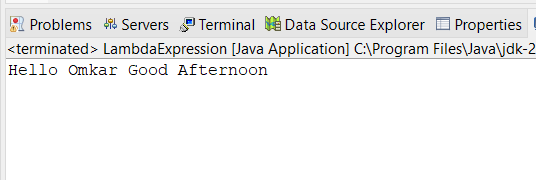
Hello obj = () -> System.out.println("Hello Omkar Good Afternoon");

obj.hello();

}

}

**Output:**

****

**2. Write a Java program using Lambda Expression with single parameters.**

**Code:**

package lambdaExpression;

//Q2. Write a java program to execute Lambda Expression with single parameter.

interface Sayable {

public String say(String name);

}

public class SingleParameter {

public static void main(String[] args) {

Sayable S1 = (name) -> {

return "Hello," + name;

};

System.*out*.println(S1.say("Omkar Phadke"));

Sayable S2 = (name) -> {

return "Omkar your " + name;

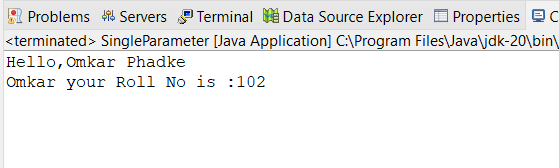
};

System.*out*.println(S2.say("Roll No is :102"));

}

}

**Output:**

****

**3. Write a Java program using Lambda Expression with multiple parameters to add two numbers.**

**Code:**

package lambdaExpression;

//Q3. Write a java program to execute Lambda Expression with multiple parameter.

interface Addition{

int add(int a, int b);

}

public class MultipleParameter {

public static void main(String[] args) {

Addition ad1=(a,b)->(a+b);

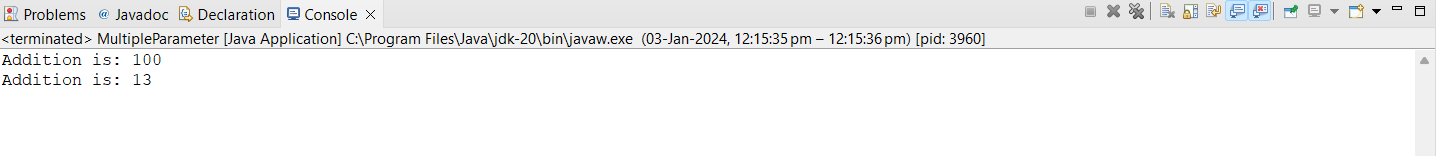
System.*out*.println("Addition is: "+ad1.add(50, 50));

Addition ad2=(int x,int y)->(x+y);

System.*out*.println("Addition is: "+ad2.add(5,8));

}

}

**Output:**

**4. Write a Java program using Lambda Expression to calculate the following:**

**a. Convert Fahrenheit to Celcius**

**Code:**

package LambdaExpression;

import java.util.Scanner;

class FahrenheitToCelsius

{

double celsius(double f)

{

return (f-32)\*5/9;

}

public static void main(String arg[])

{

double a,c;

Scanner sc=new Scanner(System.in);

System.out.print("Enter Fahrenheit temperature: ");

a=sc.nextDouble();

FahrenheittoCelsius fah=new FahrenheittoCelsius( );

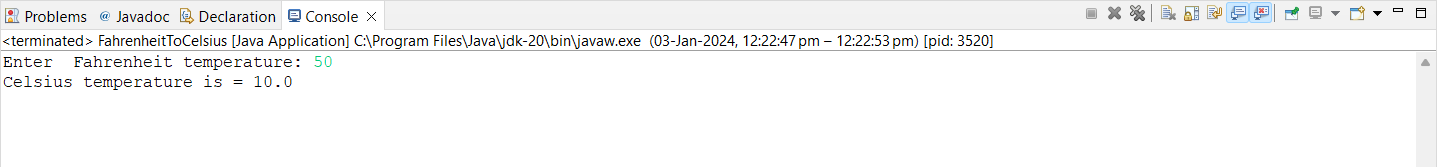
double result=fah.celsius(a);

System.out.println("Celsius temperature is = "+result);

}

}

**Output:**

****

**b. Convert Kilometers to Miles.**

**Code:**

package LambdaExpression;

import java.util.Scanner;

public class KilometerToMiles {

public static void main(String[] args) {

double kilometers, miles;

double conversionFactor = 1.609344;

System.out.print("Enter distance value in kilometer:");

Scanner input = new Scanner(System.in);

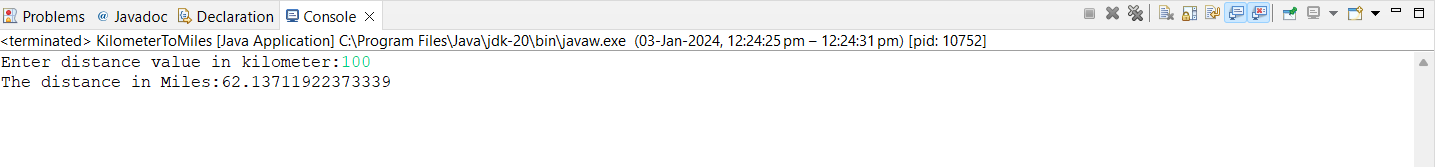
kilometers = input.nextDouble();

miles = kilometers / conversionFactor;

System.out.println("The distance in Miles:" + miles);

}

}

**Output:**

**5. Write a Java program using Lambda Expression with or without return keyword.**

**Code:**

package lambdaExpression;

@FunctionalInterface

interface Adder2 {

int add(int a, int b);

}

@FunctionalInterface

interface Subtractor {

int subtract(int a, int b);

}

public class WithoutReturnKeyword {

public static void main(String[] args) {

Adder2 adder = (a, b) -> a + b;

Subtractor subtractor = (a, b) -> a - b;

int sum = adder.add(10, 9);

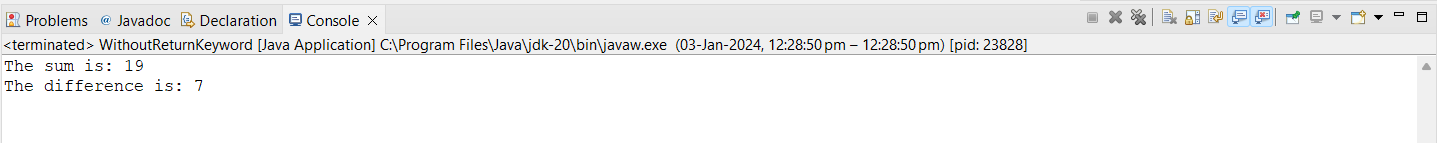
System.*out*.println("The sum is: " + sum);

int difference = subtractor.subtract(9, 2);

System.*out*.println("The difference is: " + difference);

}

}

**Output:**

**6. Write a Java program using Lambda Expression to concatenate two strings.**

**Code:**

package lambdaExpression;

@FunctionalInterface

interface ConcatenateInterface {

String concatenate(String a, String b);

}

public class Concatenate {

public static void main(String[] args) {

ConcatenateInterface concat = (String a, String b) -> a + b;

String str1 = "Anup Kumar ";

String str2 = "Bouns ka Badsha";

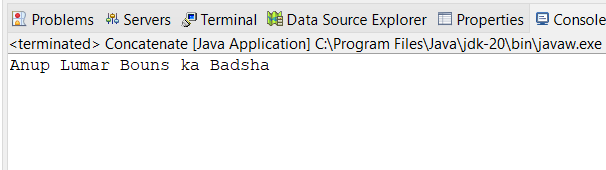
String result = concat.concatenate(str1, str2);

System.out.println(result);

}

}

**Output:**

****

# **PRACTICAL NO-6**

**Aim: Assignments based on web application development using JSP.**

**1. Write Programs to demonstrate different Implicit Objects**

**a. OUT**

**Code:**

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

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

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

<title>Implicit OUT</title>

</head>

<body>

<% int num1=100; int num2=200;

out.println("num1 is "+num1);

out.println("<br>num2 is "+num2);

%>

</body>

</html>

**Output:**

**b. Request**

**Code:**

**RequestObj.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>Request Object</title>

</head>

<body>

<form action="form1.jsp">

<input type="text" name="Username">

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

</form>

</body>

</html>

**form1.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>

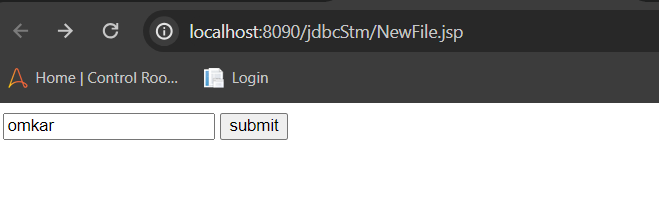
<% String Username = request.getParameter("Username");

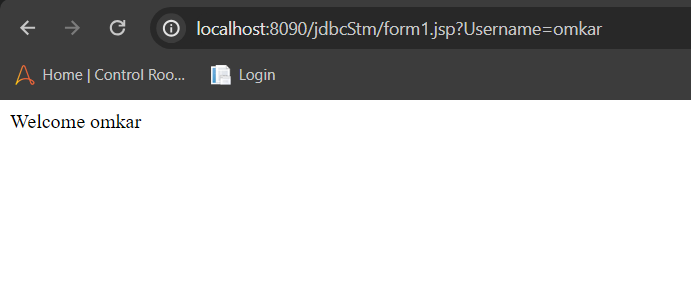
out.println("Welcome "+Username);%>

</body>

</html>

**Output:**





**c. Session Code:**

**Session.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>Session</title>

</head>

<body>

<% session.setAttribute("User", "Nishant Rane"); %>

<a href="Implicit.jsp">Click here to here User name</a>

</body>

</html>

**Implicit.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>

<% String name=(String)session.getAttribute("user");

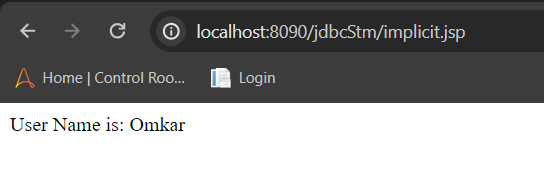
out.println("User Name is: "+name);

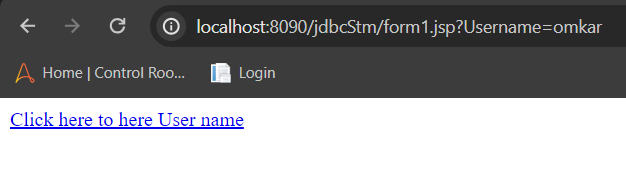
%>

</body>

</html>

**Output:**

****

****

**2. Write Programs to demonstrate temporary storage using Bean.**

**Code:**

**PersonBean.java**

package Person;

public class PersonBean{

private String FirstName;

private String LastName;

private String Email;

public String getFirstName() {

return FirstName;

}

public void setFirstName(String FirstName) {

this.FirstName = FirstName;

}

public String getLastName() {

return LastName;

}

public void setLastName(String LastName) {

this.LastName = LastName;

}

public String getEmail() {

return Email;

}

public void setEmail(String Email) {

this.Email = Email;

}

}

**Person.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>Person Bean Test</title>

</head>

<body>

<%

if("POST".equalsIgnoreCase(request.getMethod())&&request.getParameter("Submit")!= null)

%>

<jsp:useBean id=*"PersonBean"* class=*"Person.PersonBean"* scope=*"session"*>

<jsp:setProperty name=*"PersonBean"* property=*"\*"*/>

</jsp:useBean>

<%

out.println("First Name: "+PersonBean.getFirstName());

out.println("<br>Last Name: "+PersonBean.getLastName());

out.println("<br>Email: "+PersonBean.getEmail());

%>

</body>

</html>

**Form.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>

<form method=*"POST"* action=*"person.jsp"*>

<h1>Person Form</h1>

First Name: <input type=*"text"* name=*"FirstName"*>

<br><br>

Last Name: <input type=*"text"* name=*"LastName"*>

<br><br>

Email: <input type=*"text"* name=*"Email"*>

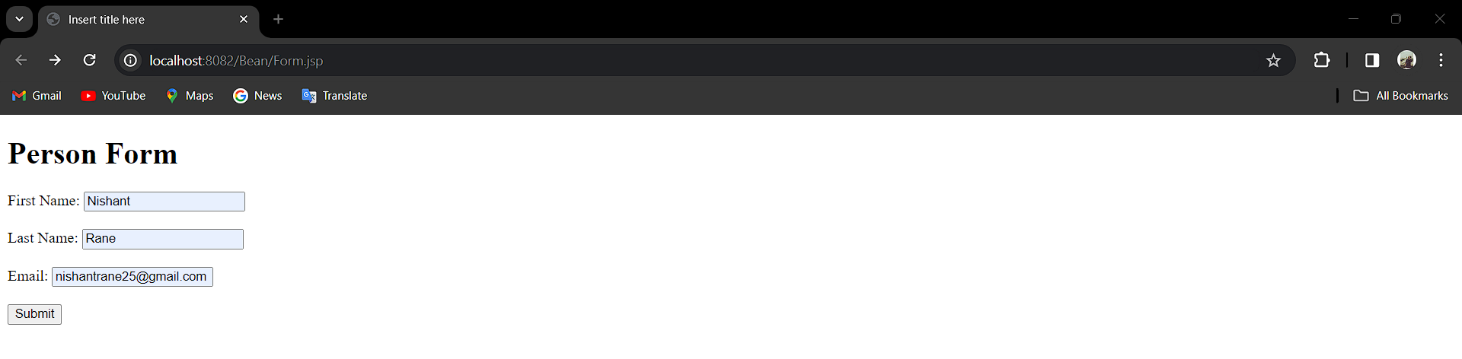
<br><br>

<Button type=*"Submit"* name=*"Submit"*>Submit</Button>

</form>

</body>

</html>

**Output:**

**3. Write a program to demonstrate Standard Action tags**

**Code:**

**Action\_cookie\_main.jsp**

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

pageEncoding=*"UTF-8"*%>

<!DOCTYPE html>

<%

Cookie username = new Cookie("username",request.getParameter("username"));

Cookie email = new Cookie("email",request.getParameter("email"));

username.setMaxAge(60\*60\*10);

email.setMaxAge(60\*60\*10);

response.addCookie(username);

response.addCookie(email);

%>

<html>

<head>

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

<title>Standard Action Tags</title>

</head>

<body>

<b>username:</b>

<%= request.getParameter("username")%>

<br><br>

<b>Email:</b>

<%= request.getParameter("email") %>

</body>

</html>

**Action\_cookie.jsp**

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

pageEncoding=*"UTF-8"*%>

<!DOCTYPE html>

<html>

<head>

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

<title></title>

</head>

<body>

<form action=*"action\_cookie\_main.jsp"* method=*"get"*>

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

<br><br>

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

<br><br>

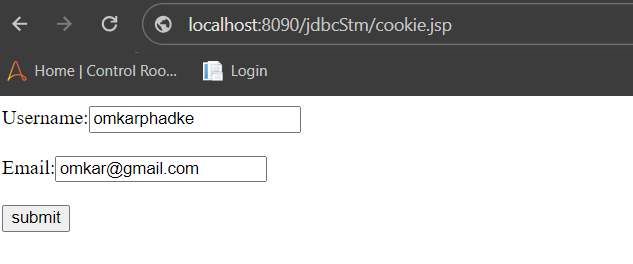
<input type=*"submit"* value=*"submit"*>

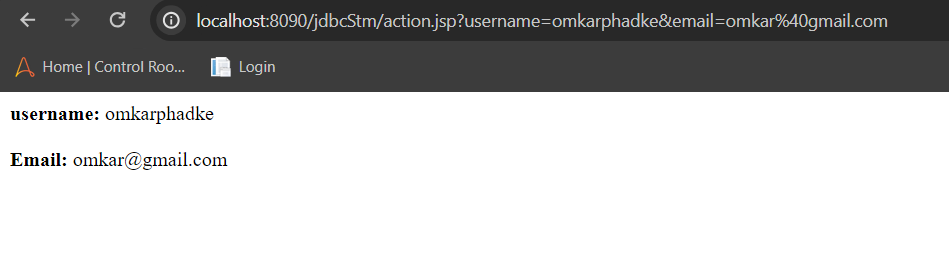
</form>

</body>

</html>

**Output:**

****



**4. Write a program to demonstrate JSP Directives**

**Code:**

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

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

<%@ taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"* %>

<!DOCTYPE html>

<html>

<head>

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

<title>JSP Directives</title>

</head>

<body>

<h2>Welcome to JSP Directives!</h2>

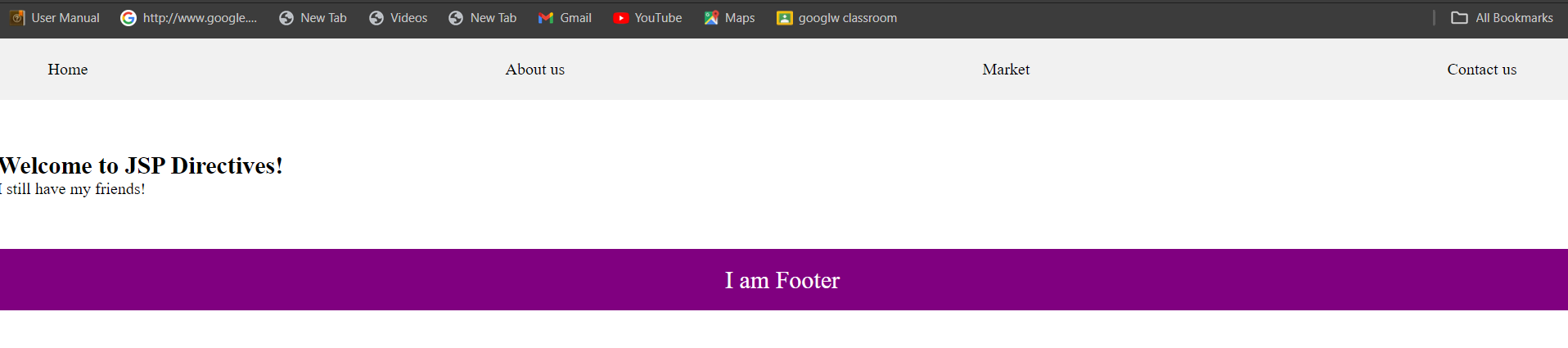
<c:out value=*"*${'I still have my friends!'}*"*/>

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

</body>

</html>

**Output:**

**5.Write a program to demonstrate Session Tracking using Cookies**

**Code:**

<%@ page import=*"java.io.PrintWriter"* %>

<%

// Get the current session or create a new one

HttpSession session1 = request.getSession(**true**);

// Set session attribute

session1.setAttribute("username", "Session:ok");

// Create a cookie for the username

Cookie usernameCookie = **new** Cookie("username", "Cookie:om");

response.addCookie(usernameCookie);

%>

<html>

<head><title>Session Tracking Using Cookies</title></head>

<body>

<h2>Session Tracking Using Cookies</h2>

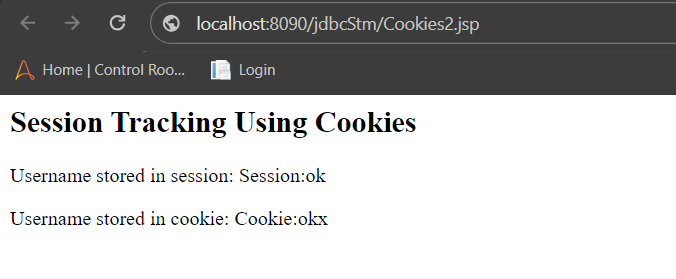
<p>Username stored in session: <%= session1.getAttribute("username") %></p>

<p>Username stored in cookie: <%= usernameCookie.getValue() %></p>

</body>

</html>

**Output:**



**6. Write a program to demonstrate JSTL Tags**

**a ction\_tags.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>action tag</title>

</head>

<body>

<h1>JSP Action Tags Demonstration</h1>

<!-- jsp:include action tag -->

<h3>Use of include action tags</h3>

<jsp:include page="header.jsp" />

<!-- jsp:useBean action tag -->

<h3>Use of useBean action tags</h3>

<jsp:useBean id="date" class="java.util.Date" /> Current Date: <%= date %><br><br>

</body>

</html>

# **h eader.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>document</title>

</head>

<body>

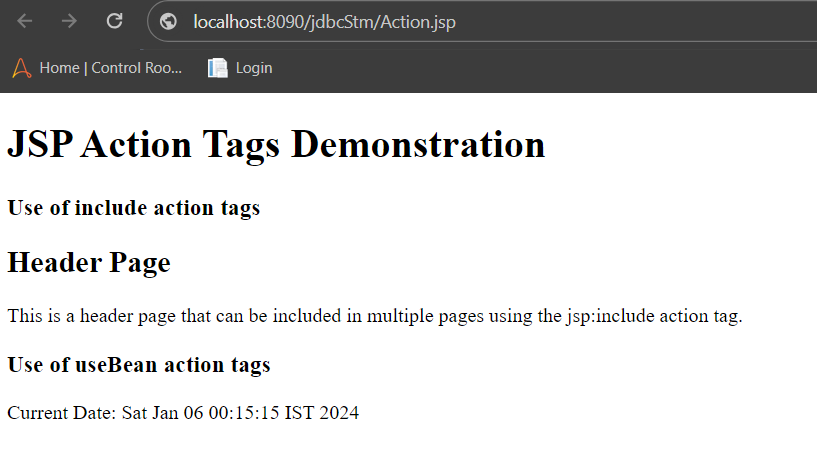
<h2>Header Page</h2>

<p>This is a header page that can be included in multiple pages using the jsp:include action tag.</p>

</body>

</html>

# **O utput:**

****

7)**Action Tags:-**

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

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

<!DOCTYPE html>

<html>

<head>

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

<title>forward tag</title>

</head>

<body>

<h1>JSP Action Tags Demonstration</h1>

<!-- jsp:forward action tag -->

<h3>Use of forward action tags</h3>

<jsp:forward page=*"forwardPage.jsp"* />

</body>

</html>

# **f orwardPage.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>

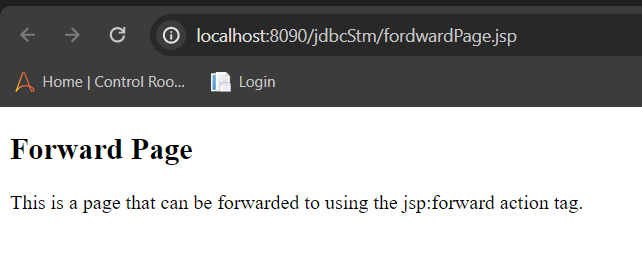
<h2>Forward Page</h2>

<p>This is a page that can be forwarded to using the jsp:forward action tag.</p>

</body>

</html>

# **O utput:**



**7. 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:**

**Index.html**

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

<fieldset>

<label for="fname">Name:</label>

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

<label for="mobileno">Mobile Number:</label>

<input type="number" name="mobileno"><br>

<label for="city">city</label>

<input type="text" name="city"><br>

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

</fieldset>

</form>

**Indexaction.jsp**

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

<html>

<head>

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

<title>Retrieve Data</title>

</head>

<body>

<h1>Retrieve Data </h1>

<table border=*"1"*>

<tr>

<th>Name</th>

<th>Mobile Number</th>

<th>City</th>

</tr>

<%

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

String moblieno=request.getParameter("mobileno");

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

**try**{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/ omkar ","root","student");

Statement st=con.createStatement();

st.executeUpdate("insert into TELEPHONE values('"+name+"','"+moblieno+"','"+city+"')");

ResultSet rs = st.executeQuery("SELECT \* FROM TELEPHONE");

**while** (rs.next()) {

%>

<tr>

<td><%= rs.getString(1) %></td>

<td><%= rs.getString(2) %></td>

<td><%= rs.getString(3) %></td>

</tr>

<%

}

con.close();

//response.sendRedirect("save.html");

}

**catch**(Exception e){

e.printStackTrace();

}

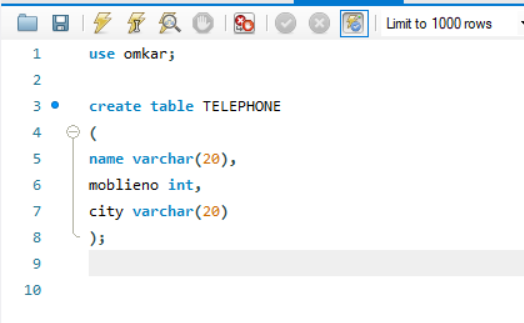
%>

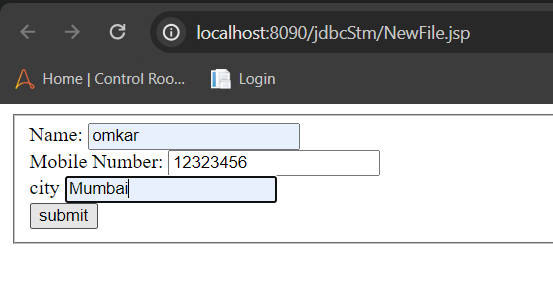
</table>

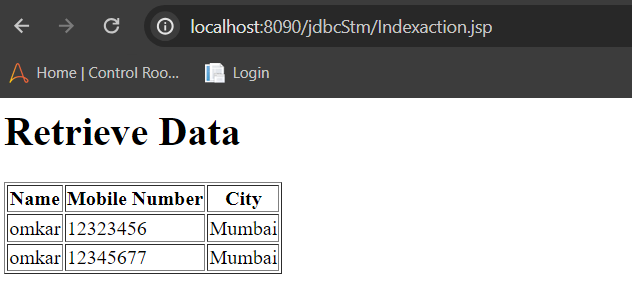
</body>

</html>

**Output:**



****

****

**8. 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:**

CREATE DATABASE omkar;

use omkar;

create table Reg(first\_name varchar(25),last\_name varchar(25),username varchar(30), password varchar(15), address varchar(40), contact int(10));

desc Reg;

**Registrationform.html**

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

<fieldset>

<h1> Registration Form</h1>

<table>

<tr>

<td>First Name</td>

<td><input type=*"text"* name=*"first\_name"* /></td>

</tr>

<tr>

<td>Last Name</td>

<td><input type=*"text"* name=*"last\_name"* /></td>

</tr>

<tr>

<td>UserName</td>

<td><input type=*"text"* name=*"username"* /></td>

</tr>

<tr>

<td>Password</td>

<td><input type=*"password"* name=*"password"* /></td>

</tr>

<tr>

<td>Address</td>

<td><input type=*"text"* name=*"address"* /></td>

</tr>

<tr>

<td>Contact No</td>

<td><input type=*"text"* name=*"contact"* /></td>

</tr></table>

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

</fieldset>

</form>

**regaction.jsp**

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

<html>

<head>

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

<title>Retrieve Data</title>

</head>

<body>

<h1> User Info </h1>

<table border=*"1"*>

<tr>

<th>First Name</th>

<th>Last Name</th>

<th>Username</th>

<th>Address</th>

<th>Contact no</th>

</tr>

<%

String first\_name = request.getParameter("first\_name");

String last\_name = request.getParameter("last\_name");

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

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

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

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

**try**{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/ omkar","root","student");

Statement st=con.createStatement();

st.executeUpdate("insert into Reg values('"+first\_name+"','"+last\_name+"','"+username+"','"+password+"','"+address+"','"+contact+"')");

ResultSet rs = st.executeQuery("SELECT \* FROM Reg");

**while** (rs.next()) {

%>

<tr>

<td><%= rs.getString(1) %></td>

<td><%= rs.getString(2) %></td>

<td><%= rs.getString(3) %></td>

<td><%= rs.getString(5) %></td>

<td><%= rs.getString(6) %></td>

</tr>

<%

}

con.close();

//response.sendRedirect("save.html");

}

**catch**(Exception e){

e.printStackTrace();

}

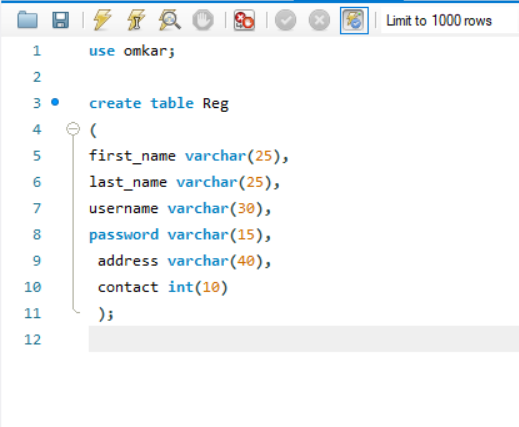
%>

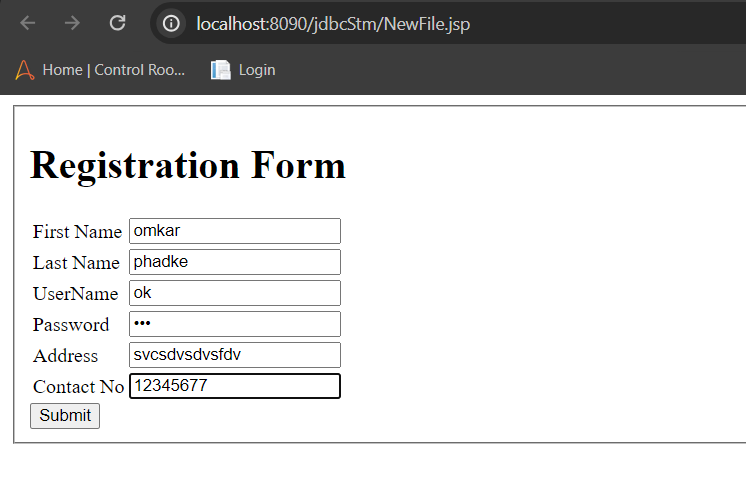
</table>

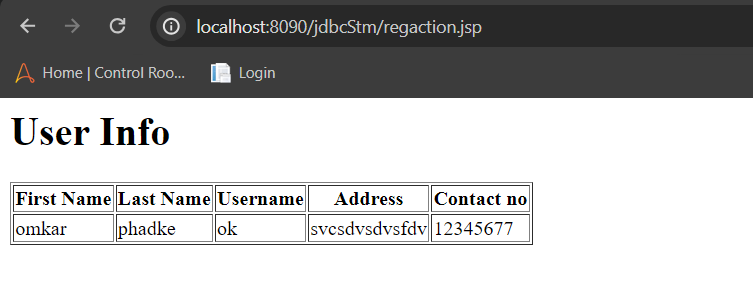
</body>

</html>

**Output:**

****

****

****

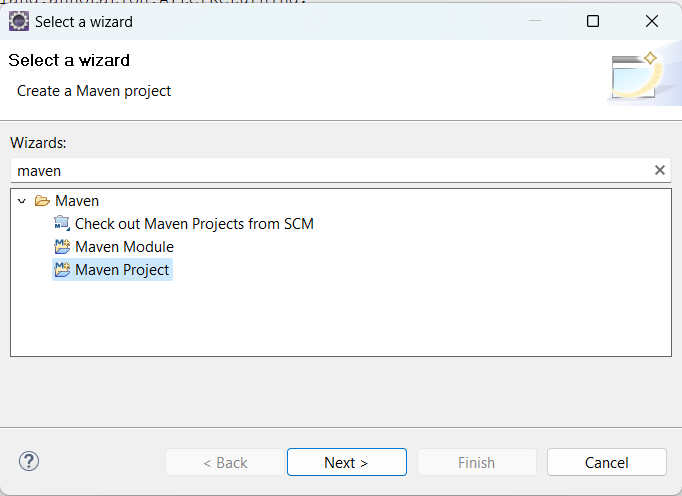
# **PRACTICALNO-7**

**Aim: Assignment Based Spring Framework**

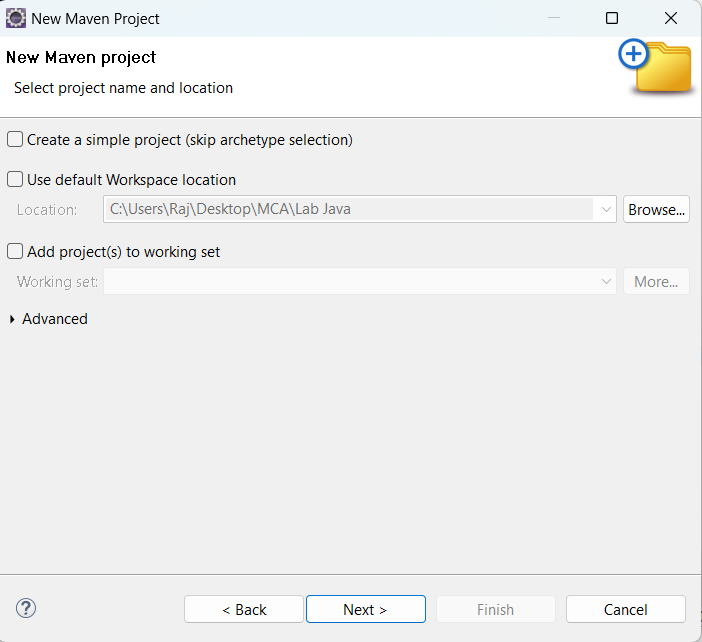
# **A) Write a program to print Singer Name and Age using spring framework.**

**Maven project**

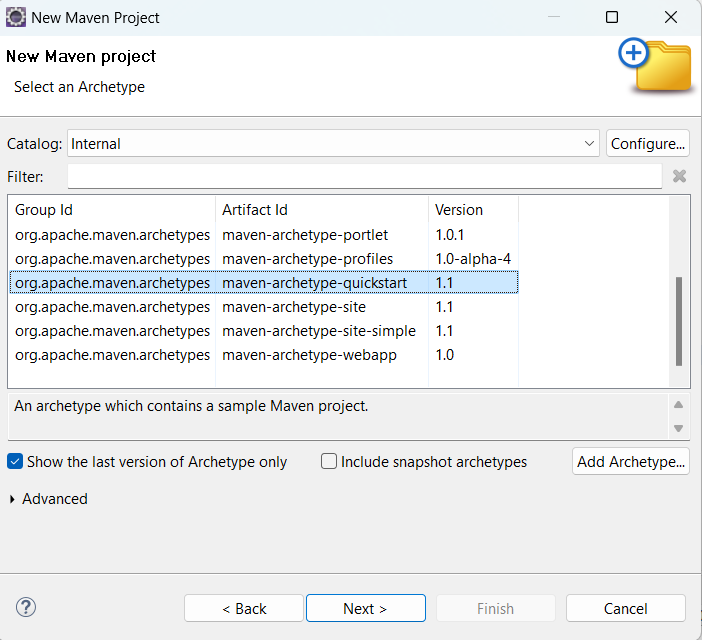
1. **Open Eclipse IDE, Navigate to File, then New, then Others., Select Maven Project, Click on the “Next” button.**



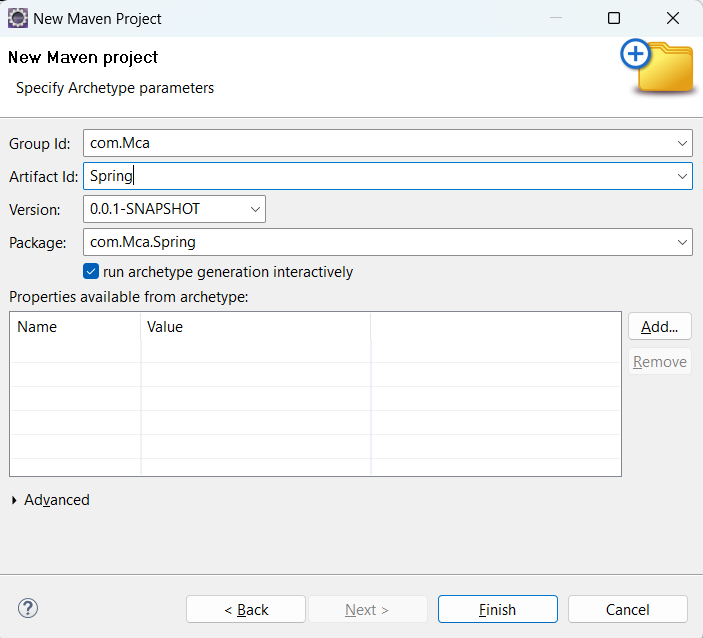
1. **Check the option ‘use default workspace location or choose your desired workspace location.**



1. **Select catalog Internal the archetype ‘maven-archetype-quickstart’.Click on the “Next” button**



1. **Enter your project’s Group Id. Enter your project’s Artifact Id. Click on the “Finish” button.**



**Pom.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.springMca</groupId>

<artifactId>springMca</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>springMca</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-core -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>5.2.3.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.2.3.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-aop -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.2.3.RELEASE</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**POJO class**

package MCA;

public class Singer {

private String name;

private Integer age;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Integer getAge() {

return age;

}

public void setAge(Integer age) {

this.age = age;

}

public Singer(String name, Integer age) {

super();

this.name = name;

this.age = age;

}

public Singer() {

super();

// TODO Auto-generated constructor stub

}

@Override

public String toString() {

return "Singer [name=" + name + ", age=" + age + "]";

}

}

**Configuration xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:c=*"http://www.springframework.org/schema/c"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<bean class=*"MCA.Singer"* name=*"singer"* p:Name=*"Luffy"* p:Age=*"19"*/>

</beans>

**Main class**

package MCA;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class test {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("MCA/mcaConfig.xml");

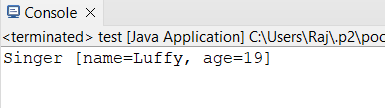
Singer temp = (Singer) context.getBean("singer");

System.out.println(temp);

}

}

**Output:**

****

# **Write a program to demonstrate dependency injection via setter method. (Primitive)**

**POJO Class**

package MCA;

public class Zoro {

private String name;

private double height;

private int swords;

// setter and getter methods

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public double getHeight() {

return height;

}

public void setHeight(double height) {

this.height = height;

}

public int getSwords() {

return swords;

}

public void setSwords(int swords) {

this.swords = swords;

}

// Constructor

public Zoro(String name, double height, int swords) {

super();

this.name = name;

this.height = height;

this.swords = swords;

}

public Zoro() {

super();

}

// to string method

@Override

public String toString() {

return "name of Character = " + name + ", height of Character = " + height + ", No. of swords = " + swords ;

}

}

**Configuration xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:c=*"http://www.springframework.org/schema/c"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<bean class=*"MCA.Zoro"* name=*"zoro"* p:name=*"Pirate Hunter Roronoa Zoro"* p:height=*"6.2"* p:swords=*"3"*/>

</beans>

**Main class**

package MCA;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class test {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("MCA/mcaConfig.xml");

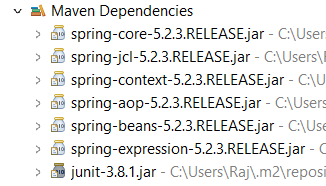
Zoro temp = (Zoro) context.getBean("zoro");

System.out.println(temp);

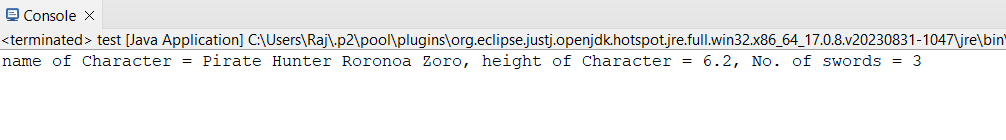
}

}

**Maven Dependencies**



Output:



# **Write a program to demonstrate dependency injection via Constructor. (Primitive)**

**POJO class**

package MCA;

public class luffy {

private String name;

private int gears;

private double height;

public luffy(String name, int gears, double height) {

super();

this.name = name;

this.gears = gears;

this.height = height;

}

@Override

public String toString() {

return " Charactername = " + name + ", No. of gears = " + gears + ", height = " + height + "]";

}

}

**Configuration xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:c=*"http://www.springframework.org/schema/c"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<bean class=*"MCA.luffy"* name=*"luffy"* c:name=*"Monkey D. Luffy"* c:height=*"5.8"* c:gears=*"5"*/>

</beans>

**Main class**

package MCA;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class test {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("MCA/mcaConfig.xml");

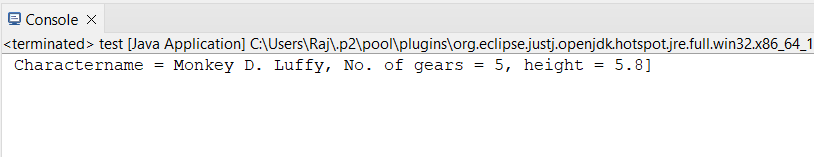
luffy temp = (luffy) context.getBean("luffy");

System.out.println(temp);

}

}

**Output:**



# **Write a program to demonstrate dependency injection via setter method.(Non Primitive)**

**POJO class**

package MCA;

public class sanji {

private String name;

private double height;

private Zoro obj;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public double getHeight() {

return height;

}

public void setHeight(double height) {

this.height = height;

}

public Zoro getObj() {

return obj;

}

public void setObj(Zoro obj) {

this.obj = obj;

}

public sanji(String name, double height, Zoro obj) {

super();

this.name = name;

this.height = height;

this.obj = obj;

}

public sanji() {

super();

// TODO Auto-generated constructor stub

}

@Override

public String toString() {

return "sanji [name=" + name + ", height=" + height + ", \nobj=" + obj + "]";

}

}

**Reference class**

package MCA;

public class Zoro {

private String name;

private double height;

private int swords;

// setter and getter methods

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public double getHeight() {

return height;

}

public void setHeight(double height) {

this.height = height;

}

public int getSwords() {

return swords;

}

public void setSwords(int swords) {

this.swords = swords;

}

// Constructor

public Zoro(String name, double height, int swords) {

super();

this.name = name;

this.height = height;

this.swords = swords;

}

public Zoro() {

super();

}

// to string method

@Override

public String toString() {

return "name of Character = " + name + ", height of Character = " + height + ", No. of swords = " + swords ;

}

}

**Configuration xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:c=*"http://www.springframework.org/schema/c"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<bean class=*"MCA.Zoro"* name=*"zoro"* p:name=*"Pirate Hunter Roronoa Zoro"* p:height=*"6.2"* p:swords=*"3"*/>

<bean class=*"MCA.sanji"* name=*"sanji"* p:name=*"Vinsmoke Sanji"* p:height=*"6.0"* p:obj-ref=*"zoro"*/>

</beans>

**Main class**

package MCA;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class test {

public static void main(String[] args) {

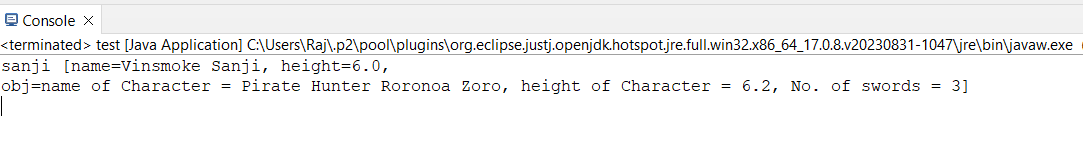
ApplicationContext context = new ClassPathXmlApplicationContext("MCA/mcaConfig.xml");

sanji temp = (sanji) context.getBean("sanji");

System.out.println(temp);

}

}

**Output:**

# **5.Write a program to demonstrate dependency injection via Constructor.(Non Primitive)By Ref**

**POJO class**

package MCA;

public class ussop {

private String Name;

private double height;

private luffy obj;

@Override

public String toString() {

return "ussop [Name=" + Name + ", height=" + height + ", \nobj=" + obj + "]";

}

public ussop(String name, double height, luffy obj) {

super();

Name = name;

this.height = height;

this.obj = obj;

}

}

**Reference Class**

**package** MCA;

**public** **class** luffy {

**private** String name;

**private** **int** gears;

**private** **double** height;

**public** luffy(String name, **int** gears, **double** height) {

**super**();

**this**.name = name;

**this**.gears = gears;

**this**.height = height;

}

@Override

**public** String toString() {

**return** " Charactername = " + name + ", No. of gears = " + gears + ", height = " + height + "]";

}

}

Configuration xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:c=*"http://www.springframework.org/schema/c"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<bean class=*"MCA.luffy"* name=*"luffy"* c:name=*"Monkey D. Luffy"* c:height=*"5.8"* c:gears=*"5"*/>

<bean class=*"MCA.ussop"* name=*"ussop"* c:name=*"Sogeking Ussop"* c:height=*"5.11"* c:obj-ref=*"luffy"*/>

</beans>

**Main Class**

package MCA;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class test {

public static void main(String[] args) {

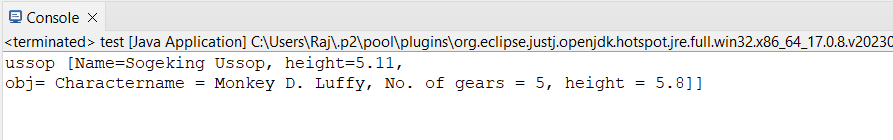
ApplicationContext context = new ClassPathXmlApplicationContext("MCA/mcaConfig.xml");

ussop temp = (ussop) context.getBean("ussop");

System.out.println(temp);

}

}

**Output:**

# **Write a program to demonstrate dependency injection via Constructor.(Collection )**

**POJO Class**

package MCA;

import java.util.\*;

public class strawHat {

private String name;

private List<String> crewName;

private Set<String> bounty;

private Map<String, String> ability;

public strawHat(String name, List<String> crewName, Set<String> bounty, Map<String, String> ability) {

super();

this.name = name;

this.crewName = crewName;

this.bounty = bounty;

this.ability = ability;

}

@Override

public String toString() {

return "strawHat [name=" + name + ", \ncrewName=" + crewName + ", \nbounty=" + bounty + ", \nability=" + ability + "]";

}

}

**Configuration xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:c=*"http://www.springframework.org/schema/c"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<bean class=*"MCA.strawHat"* name=*"strawHat"*>

<constructor-arg name=*"name"* value=*"The Straw Hat Pirates"*/>

<constructor-arg name=*"crewName"*>

<list>

<value>Monkey D. Luffy</value>

<value>Roronoa Zoro</value>

<value>First son of sea Jimbei</value>

<value>Vinksmoke Sanji</value>

<value>Demon child Nico Robin</value>

</list>

</constructor-arg>

<constructor-arg name=*"bounty"*>

<set>

<value>3,000,000,000</value>

<value>1,200,000,000</value>

<value>1,100,000,000</value>

<value>1,032,000,000</value>

<value>930,000,000</value>

</set>

</constructor-arg>

<constructor-arg name=*"ability"*>

<map>

<entry key=*"luffy"* value=*"rubber body"*/>

<entry key=*"zoro"* value=*"swordsman"*/>

<entry key=*"jimbei"* value=*"Helmsman"*/>

<entry key=*"sanji"* value=*"cook"*/>

<entry key=*"robin"* value=*"archaeologist"*/>

</map>

</constructor-arg>

</bean>

</beans>

**Main class**

package MCA;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class test {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("MCA/mcaConfig.xml");

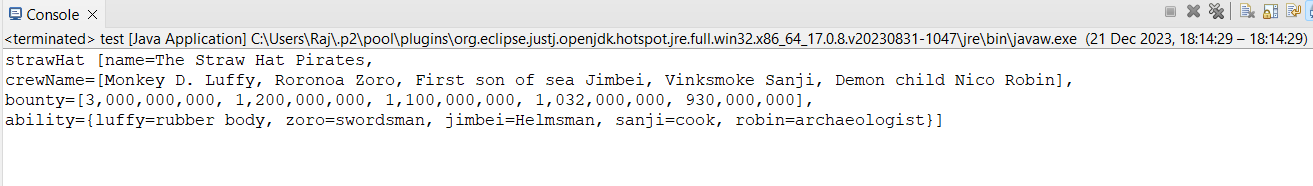
strawHat temp = (strawHat) context.getBean("strawHat");

System.out.println(temp);

}

}

**Output:**



# **Write a program to demonstrate Autowiring**

**POJO class**

package MCA;

public class chopper {

private Zoro Zoro;

public Zoro getZoro() {

return Zoro;

}

public void setZoro(Zoro zoro) {

Zoro = zoro;

}

public chopper(MCA.Zoro zoro) {

super();

Zoro = zoro;

}

public chopper() {

super();

}

@Override

public String toString() {

return "chopper [Zoro=" + Zoro + "]";

}

}

**Reference Class**

package MCA;

public class Zoro {

private String name;

private double height;

private int swords;

// setter and getter methods

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public double getHeight() {

return height;

}

public void setHeight(double height) {

this.height = height;

}

public int getSwords() {

return swords;

}

public void setSwords(int swords) {

this.swords = swords;

}

// Constructor

public Zoro(String name, double height, int swords) {

super();

this.name = name;

this.height = height;

this.swords = swords;

}

public Zoro() {

super();

}

// to string method

@Override

public String toString() {

return "name of Character = " + name + ", height of Character = " + height + ", No. of swords = " + swords ;

}

}

**Configuration xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:c=*"http://www.springframework.org/schema/c"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<bean class=*"MCA.Zoro"* name=*"Zoro"* p:name=*"Pirate Hunter Roronoa Zoro"* p:height=*"6.2"* p:swords=*"3"*/>

<bean class=*"MCA.chopper"* name=*"chopper"* autowire=*"byType"* />

</beans>

Main class

package MCA;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class test {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("MCA/mcaConfig.xml");

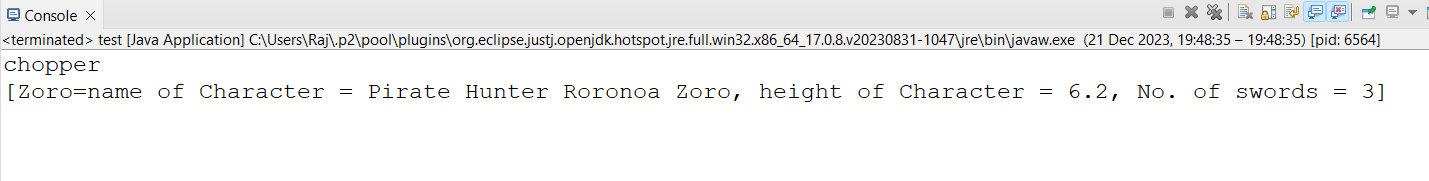
chopper temp = (chopper) context.getBean("chopper");

System.out.println(temp);

}

}

**Output**:



# **PRACTICAL NO-8**

**Aim: Assignment Based Aspect Oriented Programming**

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

**Pom.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.springMca</groupId>

<artifactId>springMca</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>springMca</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-core -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>5.2.3.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.2.3.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-aop -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.2.3.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.aspectj/aspectjrt -->

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjrt</artifactId>

<version>1.9.7</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.aspectj/aspectjweaver -->

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjweaver</artifactId>

<version>1.9.6</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**Interface**

package aop;

public interface Guitar {

public void makeSong();

}

Target Object

package aop;

public class brook implements Guitar {

public void makeSong() {

System.*out*.println("Song Started");

System.*out*.println("Song Ended");

}

}

**Aspect class**

package aop;

import org.aspectj.lang.annotation.After;

import org.aspectj.lang.annotation.AfterReturning;

import org.aspectj.lang.annotation.AfterThrowing;

import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect;

import org.aspectj.lang.annotation.Before;

@Aspect

public class mcaAspect {

@Before("execution(\* brook.makeSong())")

public void beforeSong() {

System.*out*.println("Yahoo Yahoo : I am before Aspect");

}

}

**Configuration class**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:aop=*"http://www.springframework.org/schema/aop"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/aop*

*http://www.springframework.org/schema/aop/spring-aop.xsd "*>

<aop:aspectj-autoproxy/>

<bean name=*"brook"* class=*"aop.brook"*/>

<bean name=*"mcaaspect"* class=*"aop.mcaAspect"*/>

</beans>

**Main class**

package aop;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class App {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("aop/aopConfig.xml");

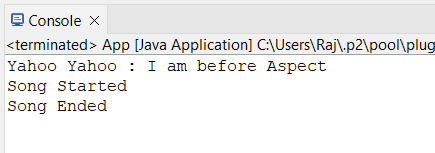
Guitar temp = (Guitar) context.getBean("brook");

temp.makeSong();

}

}

**Output:**



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

**Aspect class**

package aop;

import org.aspectj.lang.annotation.After;

import org.aspectj.lang.annotation.AfterReturning;

import org.aspectj.lang.annotation.AfterThrowing;

import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect;

import org.aspectj.lang.annotation.Before;

@Aspect

public class mcaAspect {

@After("execution(\* brook.makeSong())")

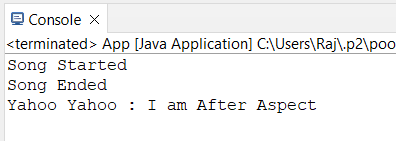
public void afterSong() {

System.*out*.println("Yahoo Yahoo : I am After Aspect");

}

}

**Output:**



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

**Aspect class**

package aop;

import org.aspectj.lang.annotation.After;

import org.aspectj.lang.annotation.AfterReturning;

import org.aspectj.lang.annotation.AfterThrowing;

import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect;

import org.aspectj.lang.annotation.Before;

@Aspect

public class mcaAspect {

@Around("execution(\* brook.makeSong())")

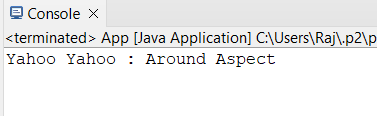
public void aroundSong() {

System.*out*.println("Yahoo Yahoo : Around Aspect");

}

}

**Output:**



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

**Aspect class**

package aop;

import org.aspectj.lang.annotation.After;

import org.aspectj.lang.annotation.AfterReturning;

import org.aspectj.lang.annotation.AfterThrowing;

import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect;

import org.aspectj.lang.annotation.Before;

import org.aspectj.lang.annotation.Pointcut;

@Aspect

public class mcaAspect {

@AfterReturning("execution(\* brook.makeSong())")

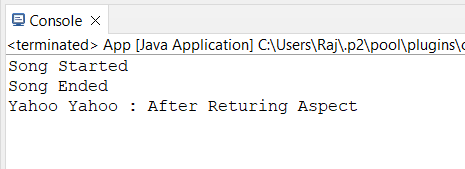
public void AfterReturnSong() {

System.*out*.println("Yahoo Yahoo : After Returing Aspect");

}

}

**Output:**



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

**Target Class:**

package aop;

public class brook implements Guitar {

public void makeSong() {

System.*out*.println("Song Started");

System.*out*.println("Song Ended");

throw new IllegalArgumentException("An error occurred while making the song.");

}

}

**Aspect class**

package aop;

import org.aspectj.lang.annotation.After;

import org.aspectj.lang.annotation.AfterReturning;

import org.aspectj.lang.annotation.AfterThrowing;

import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect;

import org.aspectj.lang.annotation.Before;

import org.aspectj.lang.annotation.Pointcut;

@Aspect

public class mcaAspect {

@Pointcut("execution(\* brook.makeSong(..))")

private void selectAll(){}

@AfterThrowing(pointcut = "selectAll()", throwing= "error")

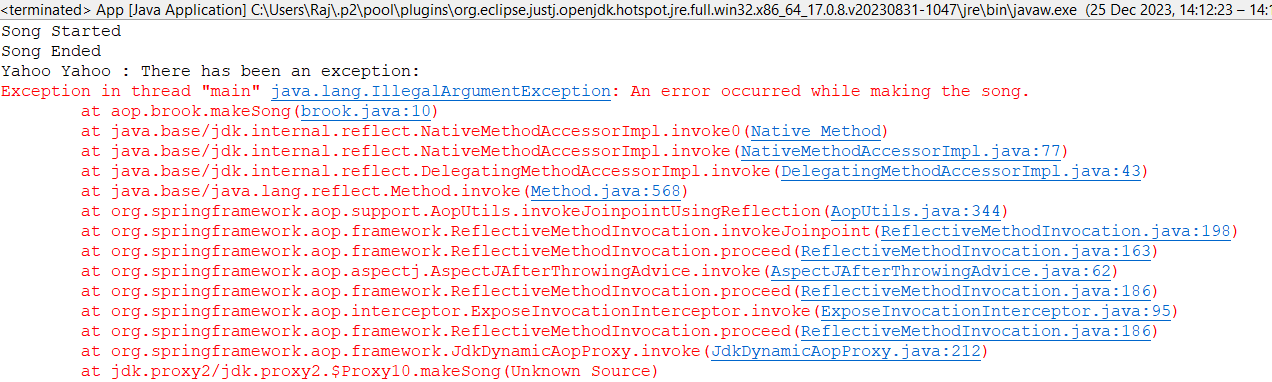
public void afterThrowingAdvice(IllegalArgumentException error){

System.*out*.println("Yahoo Yahoo : There has been an exception: ");

}

}

**Output:**



# **Write a program to demonstrate Spring AOP – pointcuts.**

**Aspect class**

package aop;

import org.aspectj.lang.annotation.After;

import org.aspectj.lang.annotation.AfterReturning;

import org.aspectj.lang.annotation.AfterThrowing;

import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect;

import org.aspectj.lang.annotation.Before;

import org.aspectj.lang.annotation.Pointcut;

@Aspect

public class mcaAspect {

@Pointcut("execution(\* brook.makeSong())")

public void songPointCut() {

System.*out*.println("Yahoo Yahoo : I am pointcut ");

}

@AfterReturning("songPointCut()")

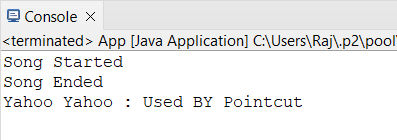
public void afterSong() {

System.*out*.println("Yahoo Yahoo : Used BY Pointcut");

}

}

**Output:**



# **PRACTICAL NO-9**

# **Aim: Assignment Based Spring Jdbc**

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

**Pom.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.mca</groupId>

<artifactId>springJDBC</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>springJDBC</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-core -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>5.2.3.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.2.3.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>5.2.3.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.20</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

Config.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:c=*"http://www.springframework.org/schema/c"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<bean class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"* name=*"ds"*>

<property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"*/>

<property name=*"url"* value=*"jdbc:mysql://localhost:3306/springjdbc"* />

<property name=*"username"* value=*"root"*/>

<property name=*"password"* value=*"root"*/>

</bean>

<bean class=*"org.springframework.jdbc.core.JdbcTemplate"* name=*"jdbcTemplate"* p:dataSource-ref=*"ds"*/>

</beans>

**Main class**

package com.mca;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.jdbc.core.JdbcTemplate;

public class App

{

public static void main( String[] args )

{

System.*out*.println( "kaizokuo ni ore wa naru!" );

ApplicationContext context = new ClassPathXmlApplicationContext("com/mca/config.xml");

JdbcTemplate temp = context.getBean("jdbcTemplate", JdbcTemplate.class);

// insert Query

String query1 = "insert into strawHat values(?,?,?)";

String query2 = "update strawHat set bounty=? where id=?";

String query3 = "delete from strawHat where id=?";

// fire query

int result1 = temp.update(query1,2,"zoro","1.2 Billion");

System.*out*.println("Number of records insetred " + result1);

int result2 = temp.update(query2,"4 billion",1);

System.*out*.println("Number of records updated " + result2);

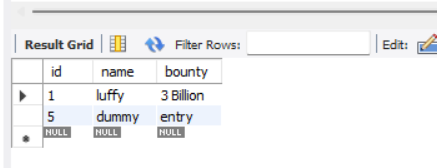
int result3 = temp.update(query3,5);

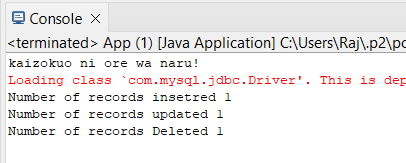
System.*out*.println("Number of records Deleted " + result3);

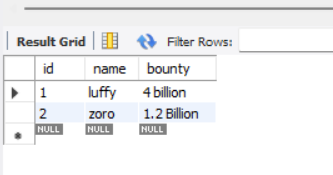
}

}

**Output:**







# **Write a program to demonstrate PreparedStatement in Spring JdbcTemplate**

**Main class**

**package** com.mca;

**import** java.sql.Connection;

**import** java.sql.PreparedStatement;

**import** java.sql.SQLException;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

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

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

**public** **class** App

{

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

{

System.***out***.println( "kaizokuo ni ore wa naru!" );

ApplicationContext context = **new** ClassPathXmlApplicationContext("com/mca/config.xml");

JdbcTemplate temp = context.getBean("jdbcTemplate", JdbcTemplate.**class**);

String query1 = "insert into strawHat(id,name,bounty) values(?,?,?)";

**int** result = temp.update(**new** PreparedStatementCreator() {

@Override

**public** PreparedStatement createPreparedStatement(Connection con) **throws** SQLException {

PreparedStatement ps = con.prepareStatement(query1);

ps.setInt(1, 3);

ps.setString(2, "zoro");

ps.setString(3, "1.1 Billion");

**return** ps;

}

});

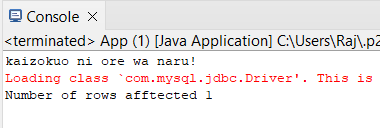
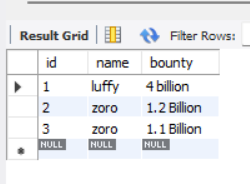
System.***out***.println("Number of rows afftected " + result);

}

}

**Output:**

## 



# **PRACTICAL NO-10**

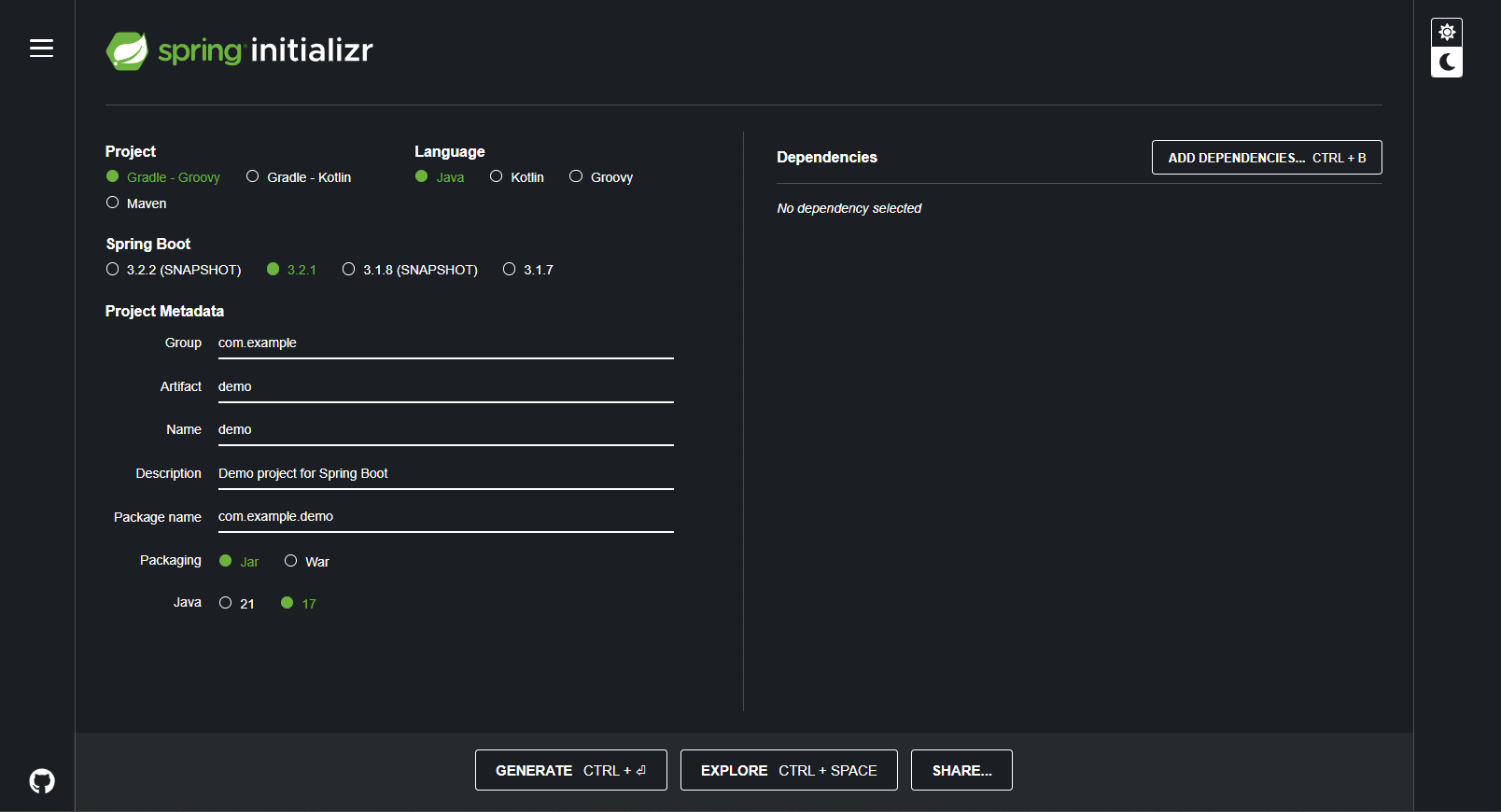
# **Aim: ASSIGNMENT BASED SPRING BOOT AND RESTFUL WEB** SERVICES

# Write a program to create a simple Spring Boot application prints a message

1. **Go to**[**Spring Initializr**](https://start.spring.io/)**. Select the type of project (Maven).**

**Choose the language (Java). Select the Spring Boot version.**

**Fill in the project metadata. Add the necessary dependencies (at least spring-boot-starter-web). Click on “Generate” to download the project.**

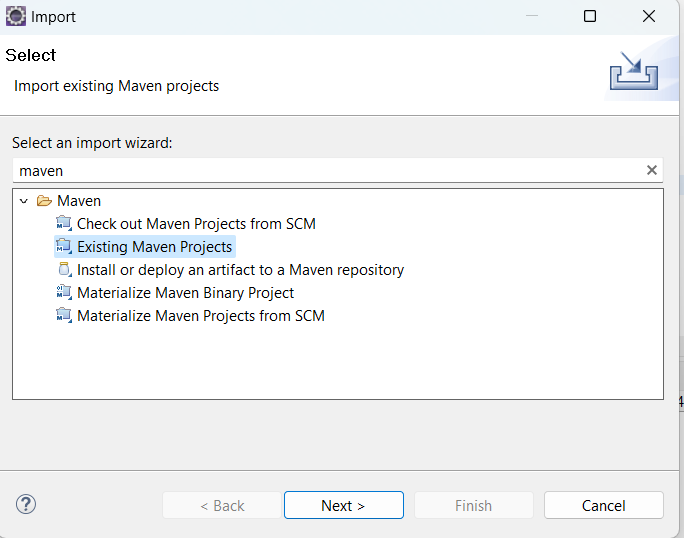


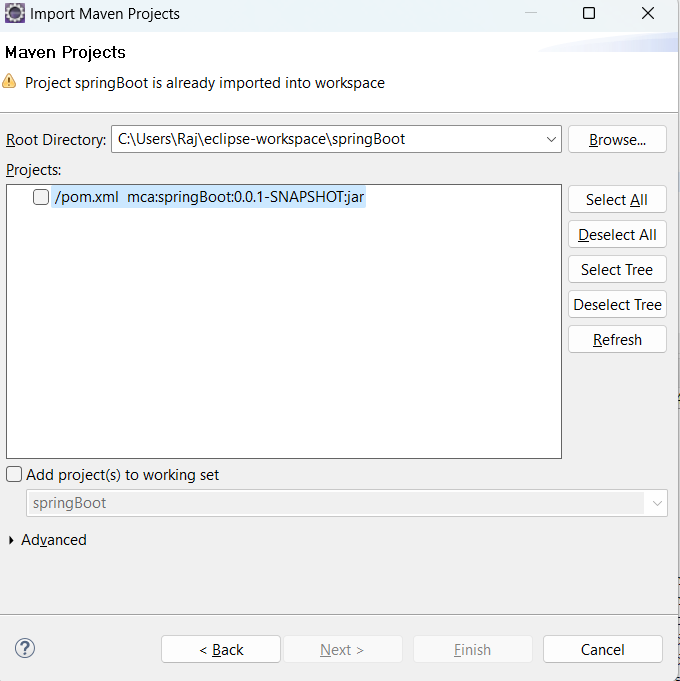
**2) Open Eclipse IDE. Navigate to File > Import.**

**Select “Existing Maven Projects”. Click on “Next”.**

**Click on “Browse” and navigate to the location where you downloaded the project.**

**Make sure the pom.xml file is checked. Click on “Finish”.**





**Main class**

**package** com.mca.spring;

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

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

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

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

@SpringBootApplication

**public** **class** myApplication {

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

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

}

@RestController

**public** **class** controller {

@GetMapping("/")

**public** String quote() {

**return** "Hero? No! We’re pirates! I love heroes, but I don’t wanna be one!";

}

}

}

**Output:**



