**Practical No: 1**

**Aim: Assignments on Java Generics.**

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

**Code:**

package collection;

class main1<T> {

T obj;

void add(T obj) {

this.obj=obj;

}

T get() {

return obj;

}

}

public class my\_generic{

public static void main(String[] args) {

main1<Integer> nn =new main1<Integer>();

nn.add(11);

main1<String> ss =new main1<String>();

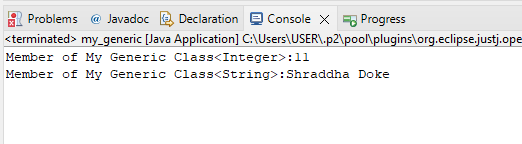
ss.add("Shraddha Doke");

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 collection;

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};

Character[] char\_Array = {'S','H','R','D','H','A'};

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

printGenericArray(int\_Array);

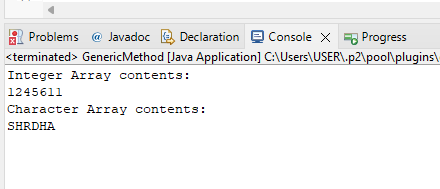
System.out.println("Character Array contents: ");

printGenericArray(char\_Array );

}

}

**Output:**



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

**1) UpperBounded:**

package WildCard;

import java.util.Arrays;

import java.util.List;

public class Unbounded {

public static void main(String[] args) {

List<Integer> list1 = Arrays.asList(1, 2, 3);

List<Double> list2 = Arrays.asList(1.1, 2.2, 3.3);

printlist(list1);

printlist(list2);

}

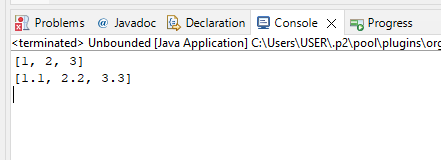
private static void printlist(List<?> list) {

System.out.println(list);

}

}

**Output:**



**2) Lowerbound**

package WildCard;

import java.util.Arrays;

import java.util.List;

public class lowerbound {

public static void main(String[] args) {

List<Integer>list1=Arrays.asList(5,8,6,7,11);

superClass(list1);

List<Number>list2=Arrays.asList(4,8,55,7,41);

superClass(list2);

}

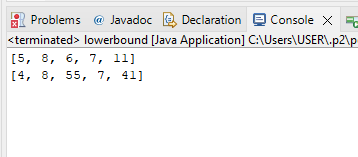
public static void superClass(List<? super Integer>list) {

System.out.println(list);

}

}

**Output:**



**3) UpperBounded**

package WildCard;

import java.util.\*;

public class upperbound<T> {

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

double a=0.0;

for(Number n:numbers)

a+=n.doubleValue();

return a;}

public static void main(String[] args) {

List<Integer>list1=Arrays.asList(5,8,6,7,11);

System.out.println(sum(list1));

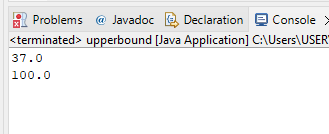
List<Double>list2=Arrays.asList(4.2,8.5,5.5,77.7,4.1);

System.out.println(sum(list2));

}

}

**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 collection;

import java.util.\*;

public class ArrayListDemo {

public static void main(String args[]) {

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

list.add("hello");

list.add("dear");

list.add("root");

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

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

for(String s:list){

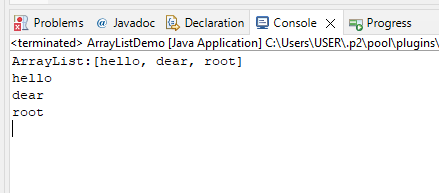
System.out.println(s);

}

}

}

**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 collection;

import java.util.ArrayList;

import java.util.Iterator;

public class ReverseArrayList {

public static void main(String[] args) {

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

arr.add("hello");

arr.add("dear");

arr.add("root");

System.out.println(arr);

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

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

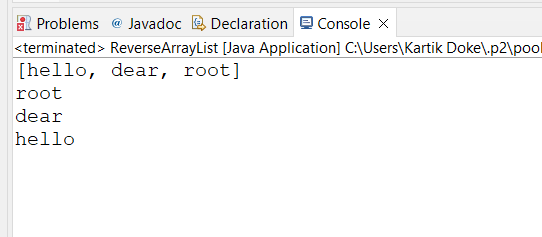
System.out.println(arr.get(i));

}

}

}

**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 Collection;

import java.util.\*;

public class Hash\_Set {

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("Papaya");

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="Papaya";

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);

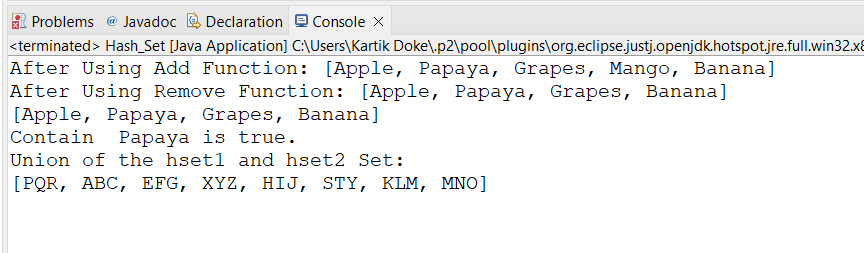
System.out.println("Union of the hset1 and hset2 Set: ");

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** Collection;

**import** java.util.\*;

**public** **class** MapInterface {

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

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

map1.put(**new** ~~Integer~~(1), "Shraddha");

map1.put(**new** ~~Integer~~(2), "Kartik");

map1.put(**new** ~~Integer~~(3), "Vighnesh");

map1.put(**new** ~~Integer~~(4), "Ovi");

map1.put(**new** ~~Integer~~(5), "Sonam");

map1.put(**new** ~~Integer~~(6), "Siddhi");

System.***out***.println(map1);

map1.remove(**new** ~~Integer~~(3));

System.***out***.println(map1);

System.***out***.println("Key 3 is present "+map1.containsKey(3));

String val=(String)map1.get(4);

System.***out***.println("Value of specified key: "+val);

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

map2.put(6, "Ritika");

map2.put(7, "Snehal");

map2.put(8, "Radhika");

map2.put(9, "Nikita");

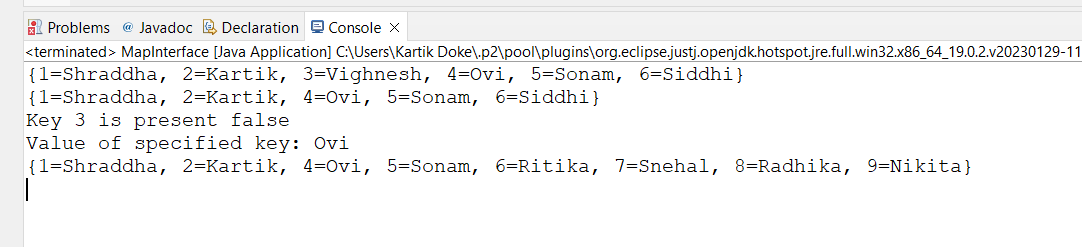
map1.putAll(map2);

System.***out***.println(map1);

}

}

**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** P1 {

**interface** Hello {

**void** hello();

}

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

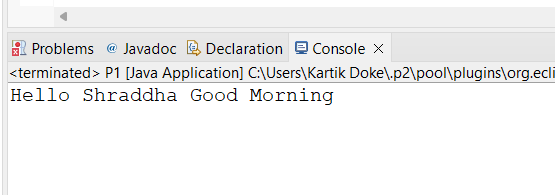
Hello obj = () -> System.***out***.println("Hello Shraddha Good Morning");

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** P2 {

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

Sayable S1 = (name) -> {

**return** "Hello," + name;

};

System.***out***.println(S1.say("Shraddha"));

Sayable S2 = (name) -> {

**return** "Shraddha your " + name;

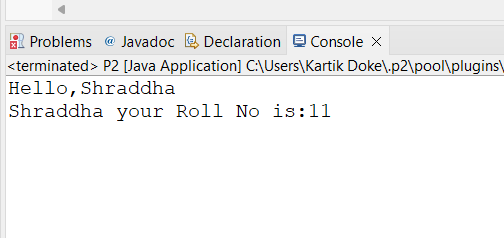
};

System.***out***.println(S2.say("Roll No is:11"));

}

}

**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** P3 {

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

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

System.***out***.println("Addition is: "+ad1.add(50, 11));

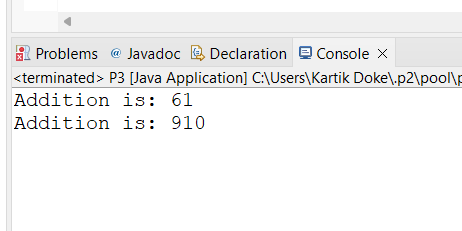
Addition ad2=(**int** x,**int** y)->(x+y);

System.***out***.println("Addition is: "+ad2.add(510,400));

}

}

**Output:**

****

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

**a. Convert Fahrenheit to Celcius**

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( );

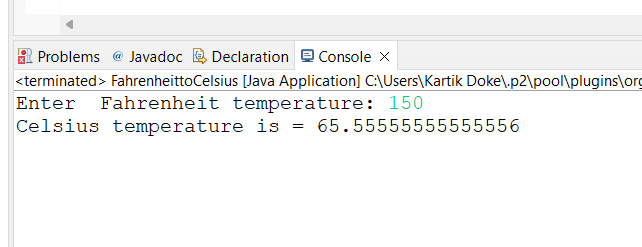
double result=fah.celsius(a);

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

}

}

**Output:**

****

**b. Convert Kilometers to Miles.**

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();

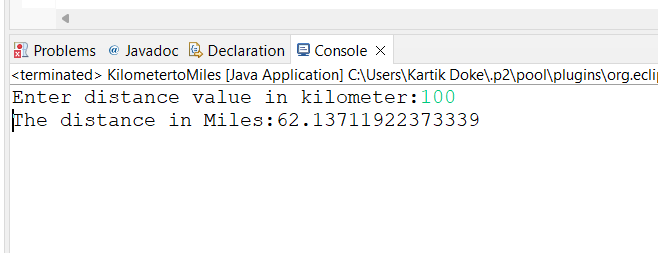
miles = kilometers / conversionFactor;

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;

**import** java.util.stream.LongStream;

**public** **class** P7 {

**public** **static** **long** factorialStreams(**long** n) {

**return** LongStream.*rangeClosed*(2, n).reduce(1, (**long** x, **long** y) -> x \* y);

}

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

**long** n = 5;

**long** fact;

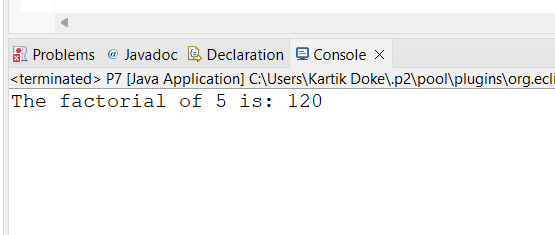
fact = *factorialStreams*(n);

System.***out***.println("The factorial of " + n + " is: " + fact);

}

}

**Output:**

****

**Practical No: 6**

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

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

**a. OUT**

<%@ 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=10; int num2=20;

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

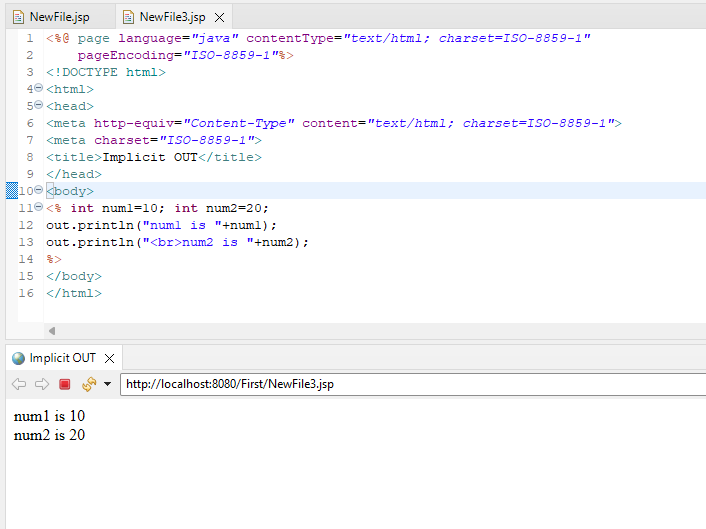
out.println("<br>num2 is "+num2);

%>

</body>

</html>

**Output:**



**b. Request**

**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>

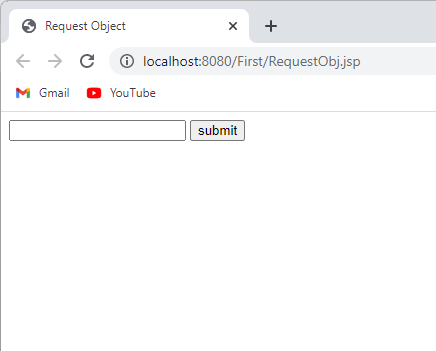
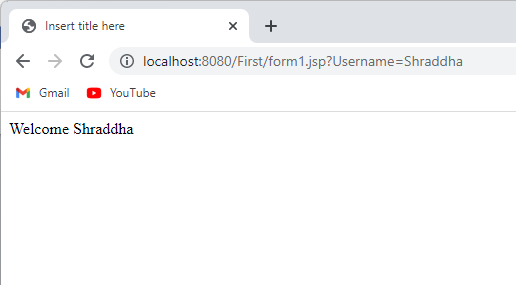
<% String Username = request.getParameter("Username");

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>Insert title here</title>

</head>

<body>

<% session.setAttribute("user", "Shraddha"); %>

<a href="Implicit\_jsp8.jsp">Click here to here User name</a>

</body>

</html>

**Implicit\_jsp8.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");

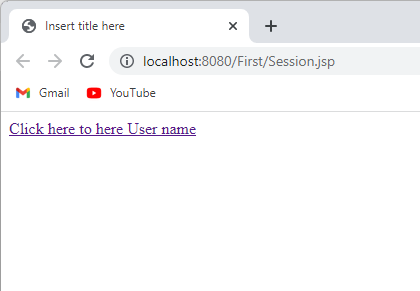
out.println("User Name is: "+name);

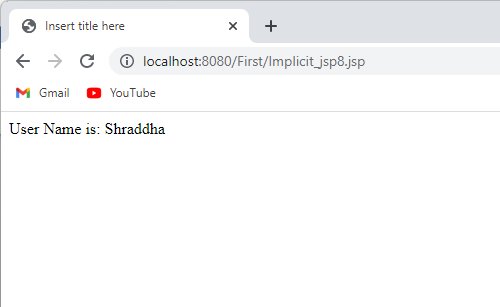
%>

</body>

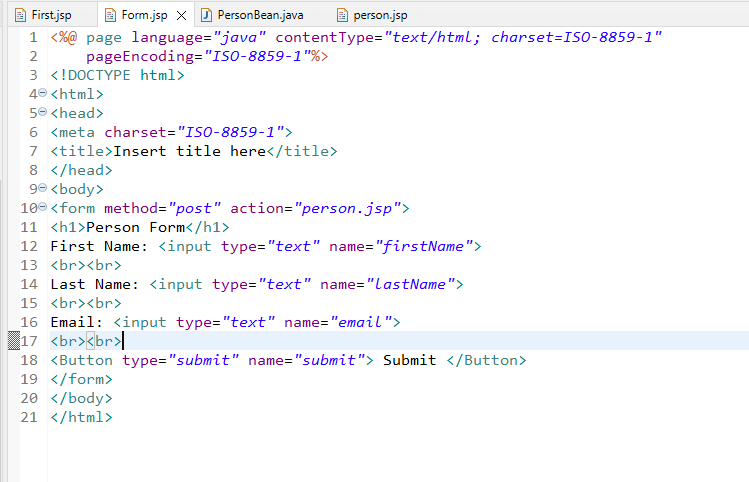
</html>

**Output:**





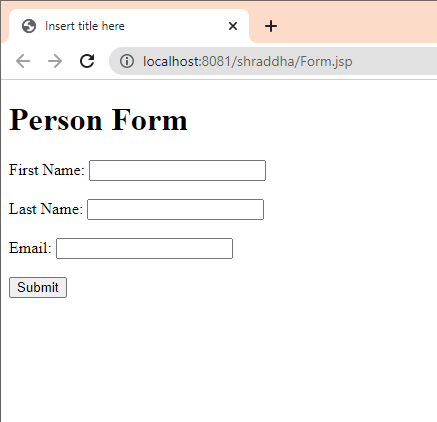
**2. Write Programs to demonstrate temporary storage using Bean.**

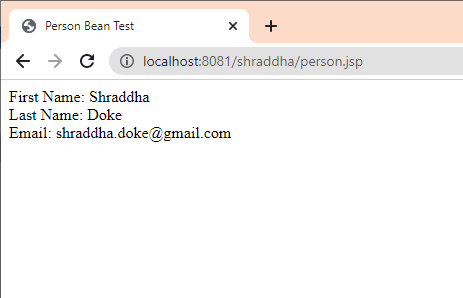






**Output:**





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

**action\_cookie.java**

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

pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Insert title here</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>

**action\_cookie\_main.java**

<%@ 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>Insert title here</title>

</head>

<body>

<b>username:</b>

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

<br><br>

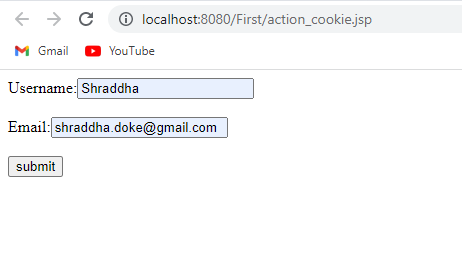
<b>Email:</b>

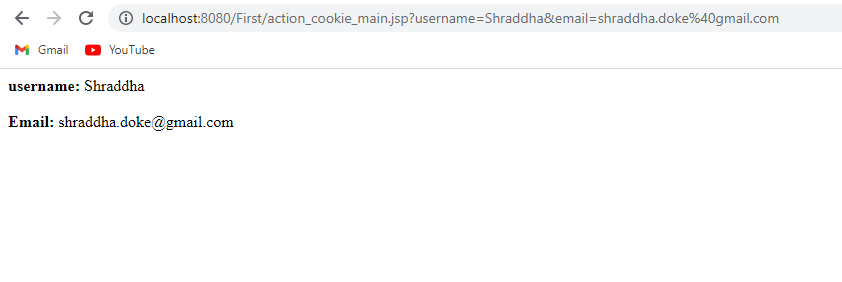
<%= request.getParameter("email") %>

</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:



**F orward Action Tag**

<%@ 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>

**Indexacton.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/ DIRECTORY ","root","12345");

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();

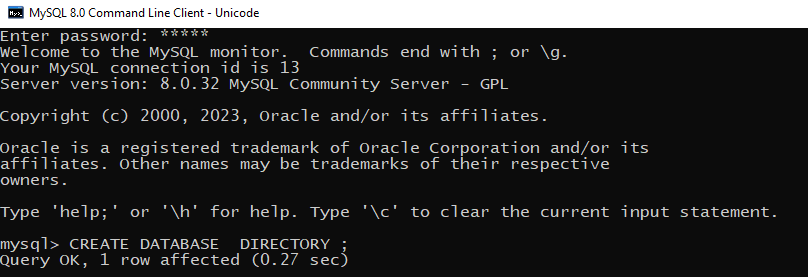
}

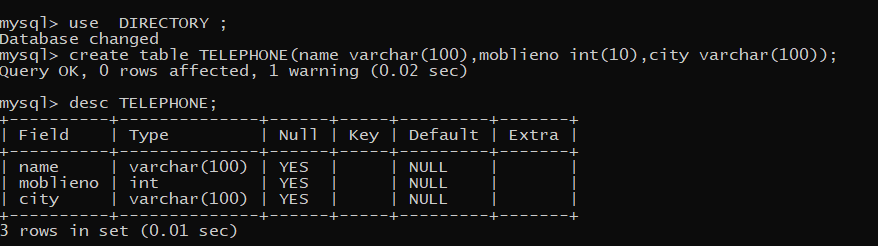
%>

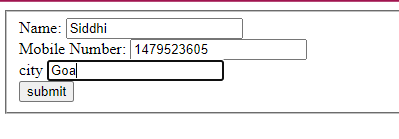
</table>

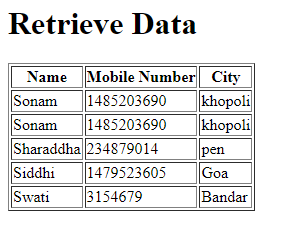
</body>

</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 Formdb;

use Formdb;

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**

<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/ Formdb ","root","12345");

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();

}

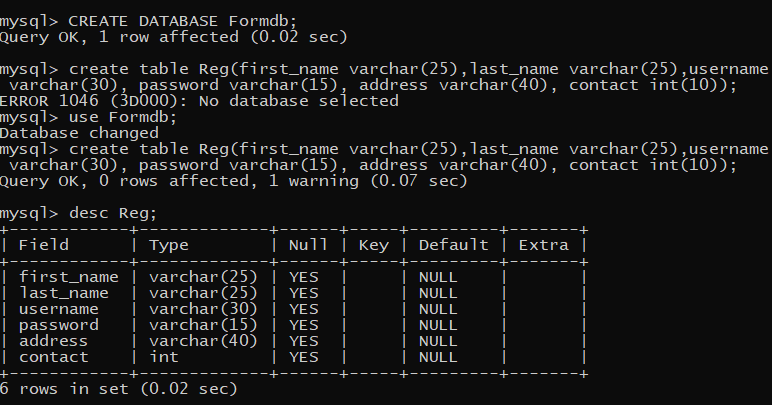
%>

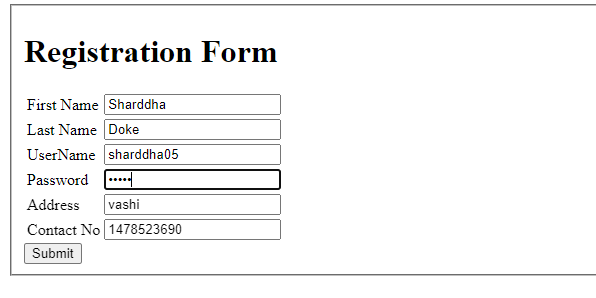
</table>

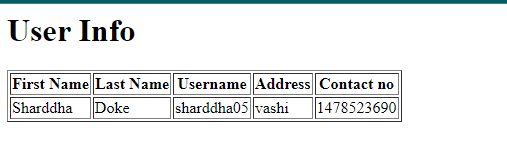
</body>

</html>

**Output:**







**Practical No: 7**

**Aim: Assignment based Spring Framework.**

**1. Write a program to print Singer Name and Age using spring framework.**

**Code:**

**saap.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"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="Singer" class="com.demo.Singer">

<property name="name" value="Radhika Panchal"></property>

<property name="age" value="21"></property>

</bean>

</beans>

**TestSinger.java**

package com.demo;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class TestSinger {

private static ApplicationContext ctx;

public static void main(String[] args) {

ctx=new ClassPathXmlApplicationContext("saap.xml");

Singer s1=(Singer)ctx.getBean("Singer");

s1.displayInfo();

}

}

**Singer.java**

package com.demo;

public class Singer {

String name;

int age;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

void displayInfo()

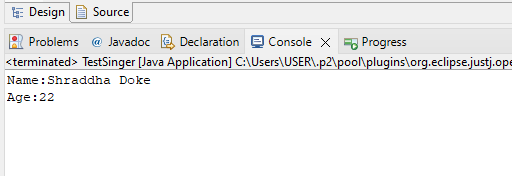
{

System.out.println("Name:"+name+ " " +"\nAge:"+age);

}

}

**Output:**



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

**Appctx.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"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="Account" class="com.demo.Account">

<constructor-arg index="0" value="1">

</constructor-arg>

<constructor-arg type="String" value="Shraddha Doke">

</constructor-arg>

<constructor-arg value="1100"></constructor-arg>

</bean>

</beans>

**Account.java**

package com.demo;

public class Account {

int acNo;

String acName;

double acbalance;

public int getAcNo() {

return acNo;

}

public void setAcNo(int acNo) {

this.acNo = acNo;

}

public String getAcName() {

return acName;

}

public void setAcName(String acName) {

this.acName = acName;

}

public double getAcbalance() {

return acbalance;

}

public void setAcbalance(double acbalance) {

this.acbalance = acbalance;

}

public Account(int acNo, String acName, double acbalance) {

super();

this.acNo = acNo;

this.acName = acName;

this.acbalance = acbalance;

}

public Account() {

super();

}

}

**AccountTest.java**

package com.demo;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.context.ApplicationContext;

public class AccountTest {

public static void main(String[] args) {

ApplicationContext ctx=new ClassPathXmlApplicationContext("appctx.xml");

Account a1=(Account) ctx.getBean("Account");

System.out.println("Ac NO:"+a1.getAcNo());

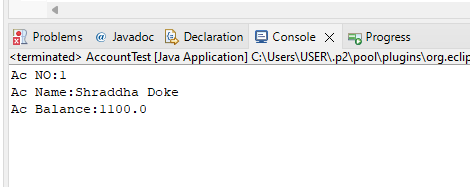
System.out.println("Ac Name:"+a1.getAcName());

System.out.println("Ac Balance:"+a1.getAcbalance());

}

}

**Output:**



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

**Shape.java**

**package** spring3;

**public** **interface** Shape {

**void** show();

}

**Circle.java**

**package** spring3;

**public** **class** Circle **implements** Shape{

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

{

System.***out***.println("I am Circle");

}

}

**ShapeManager.java**

**package** spring3;

**public** **class** ShapeManager {

Shape myShape;

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

{

**this**.myShape.show();

}

**public** Shape getMyShape() {

**return** myShape;

}

**public** **void** setMyShape(Shape myShape) {

**this**.myShape = myShape;

}

}

**ShapeTest.java**

package ShapeTest;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import spring3.ShapeManager;

public class ShapeTest {

private static ApplicationContext appCon;

public static void main(String[] args)

{

appCon = new ClassPathXmlApplicationContext("Appctx3.xml");

ShapeManager factory=(ShapeManager)appCon.getBean("ShapeMan");

factory.show();

}

}

**Appctx3.xml**

**Output:**

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

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

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

xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd"*>

<bean id=*"Circle"* class=*"spring3.Circle"*></bean>

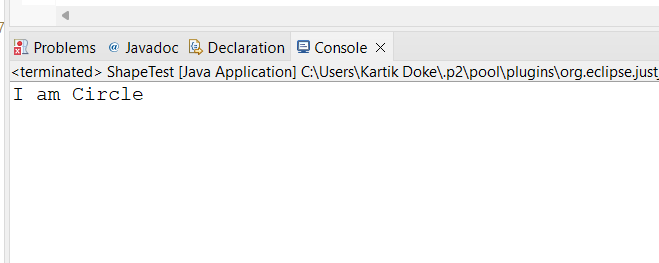
<bean id=*"ShapeMan"* class=*"spring3.ShapeManager"*>

<property name=*"myShape"* ref=*"Circle"*></property>

</bean>

</beans>

**Output:**

****

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

**Employee.java**

**package** spring2;

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

**int** id;

FullName name;

**public** Employee(**int** id, FullName name) {

**super**();

**this**.id = id;

**this**.name = name;

}

**void** show() {

System.***out***.println("Id:" + id);

System.***out***.println("Full Name:" + name);

}

}

**FullName.java**

**package** spring2;

**public** **class** FullName {

**private** String Fname;

**private** String Mname;

**private** String Lname;

**public** FullName(String fname, String mname, String lname) {

**super**();

Fname = fname;

Mname = mname;

Lname = lname;

}

**public** String toString(){

**return** Fname+" "+Mname+" "+Lname;

}

}

**Apptcx1:**

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

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

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

xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd"*>

<bean id=*"FullName"* class=*"spring2.FullName"*>

<constructor-arg value=*"Shraddha"*></constructor-arg>

<constructor-arg value=*"Rohidas"*></constructor-arg>

<constructor-arg value=*"Doke"*></constructor-arg>

</bean>

<bean id=*"Employee"* class=*"spring2.Employee"*>

<constructor-arg value=*"1"*></constructor-arg>

<constructor-arg>

<ref bean=*"FullName"*/>

</constructor-arg>

</bean>

</beans>

**EmpTest.java**

**package** spring2;

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

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

**public** **class** EmpTest {

**private** **static** ApplicationContext *appCon*;

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

{

*appCon* = **new** ClassPathXmlApplicationContext("appctx1.xml");

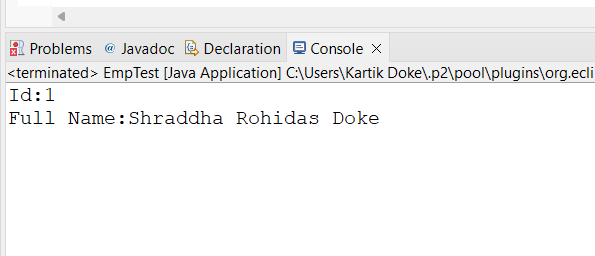
Employee factory=(Employee)*appCon*.getBean("Employee");

factory.show();

}

}

**Output:**

****

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

**Movie.java**

package spring4;

import java.util.\*;

public class Movie {

private List<String> movies;

public List<String> getMovies() {

return movies;

}

public void setMovies(List<String> movies) {

this.movies = movies;

}

public void printMovies(){

System.out.println(this.movies);

}

}

**Appctx4.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"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="Movie" class="spring4.Movie">

<property name="movies">

<list>

<value>A Beautiful Mind</value>

<value>Inception</value>

<value>Cast Away</value>

</list>

</property>

</bean>

</beans>

**MovieTest:**

package spring4;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MovieTest {

private static ApplicationContext appCon;

public static void main(String[] args) {

appCon = new ClassPathXmlApplicationContext("Appctx4.xml");

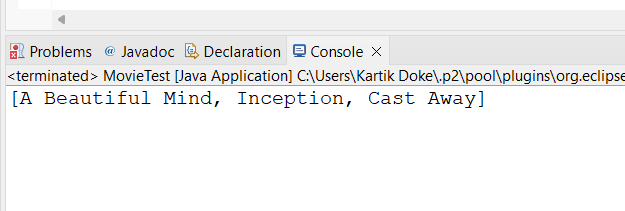
Movie factory = (Movie) appCon.getBean("Movie");

factory.printMovies();

}

}

**Output:**

****

**Practical No: 9**

**Aim: Assignment based Spring JDBC**

**Q.1 Write a program to insert, update and delete records from the given table.**

**Student.java**

package experiment8;

public class Student {

private int id;

private String name;

private int age;

// parameterized constructor

public Student(int id, String name, int age)

{

this.id = id;

this.name = name;

this.age = age;

}

// getters and setters

public int getId()

{

return id;

}

public void setId(int id)

{

this.id = id;

}

public String getName()

{

return name;

}

public void setName(String name)

{

this.name = name;

}

public int getAge()

{

return age;

}

public void setAge(int age)

{

this.age = age;

}

// a method to display data

public void show()

{

System.out.println("Id = " + id + ", Name = " + name + ", Age = " + age ); }

}

**StudentDao.java**

package experiment8;

import experiment8.Student;

public interface StudentDao {

public int save(Student s) throws Exception;

public int update(Student s) throws Exception;

public int delete(int id) throws Exception;

}

Studentdaoimpl.java

package experiment8;

import java.sql.Connection;

import java.sql.PreparedStatement;

import javax.sql.DataSource;

import experiment8.StudentDao;

import experiment8.Student;

public class StudentDaoImpl implements StudentDao {

private DataSource ds;

private Connection con;

public void setDs(DataSource ds)

{

this.ds = ds;

}

@Override

public int save(Student s) throws Exception

{

con = ds.getConnection();

String sql = "insert into student values(?,?,?)";

PreparedStatement ps = con.prepareStatement(sql); ps.setInt(1, s.getId());

ps.setString(2, s.getName());

ps.setInt(3, s.getAge());

return ps.executeUpdate();

}

@Override

public int update(Student s) throws Exception

{

con = ds.getConnection();

String sql = "update student set name=?, age=? where id=?";

PreparedStatement ps = con.prepareStatement(sql);

ps.setString(1, s.getName());

ps.setInt(2, s.getAge());

ps.setInt(3, s.getId());

return ps.executeUpdate();

}

@Override

public int delete(int id) throws Exception

{

con = ds.getConnection();

String sql = "delete from student where id=?";

PreparedStatement ps = con.prepareStatement(sql);

ps.setInt(1, id);

return ps.executeUpdate();

}

}

**Appconfig.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"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="bds" class="org.springframework.jdbc.datasource.DriverManagerDataSource">

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

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

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

<property name="password" value="12345" />

</bean>

<bean id="dao" class="experiment8.StudentDaoImpl"> <property name="ds" ref="bds" />

</bean>

</beans>

**Test.java**

package experiment8;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import experiment8.StudentDao;

import experiment8.StudentDaoImpl;

import experiment8.Student;

public class test {

public static void main(String[] args) throws Exception {

// Loading bean configuration file

ApplicationContext ac = new

ClassPathXmlApplicationContext("appConfig.xml"); // using required bean

StudentDao dao = (StudentDaoImpl)ac.getBean("dao");

// Creating Student object

Student s1 = new Student(18, "Ajit", 21);

Student s2 = new Student(45, "Hrishikesh", 22);

Student s3 = new Student(15, "Bhushan", 23);

// Saving Student obj in db

int check1 = dao.save(s1);

int check2 = dao.save(s2);

int check3 = dao.save(s3);

System.out.println("\nChecking if inserted?");

System.out.println((check1>0) ? "1st Record Inserted" : "1st Record Not Inserted"); System.out.println((check2>0) ? "2nd Record Inserted" : "2nd Record Not Inserted"); System.out.println((check3>0) ? "3rd Record Inserted" : "3rd Record Not Inserted"); s1.setName("Ajit Gupta");

s2.setName("Hrishikesh Rane");

s3.setName("Vedant Naidu");

int update1 = dao.update(s1);

int update2 = dao.update(s2);

int update3 = dao.update(s3);

System.out.println("\nChecking if updated?");

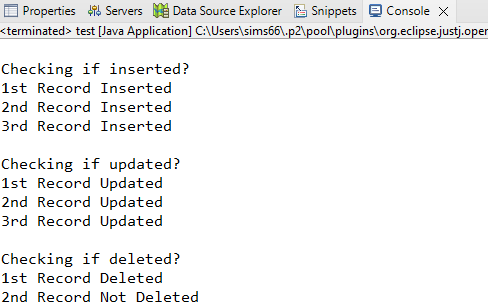
System.out.println((update1>0) ? "1st Record Updated" : "1st Record Not Updated"); System.out.println((update2>0) ? "2nd Record Updated" : "2nd Record Not Updated"); System.out.println((update3>0) ? "3rd Record Updated" : "3rd Record Not Updated"); int delete1 = dao.delete(45);

int delete2 = dao.delete(45);

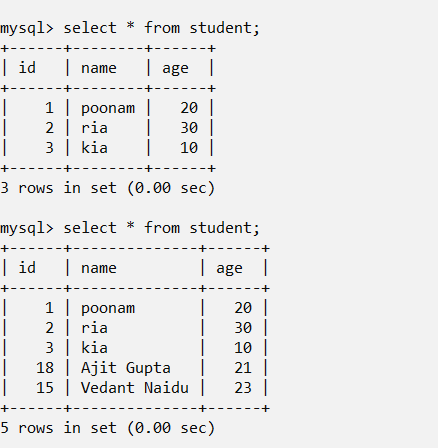
System.out.println("\nChecking if deleted?");

System.out.println((delete1>0) ? "1st Record Deleted" : "1st Record Not Deleted"); System.out.println((delete2>0) ? "2nd Record Deleted" : "2nd Record Not Deleted"); } }

**OUTPUT:-**



**OUTPUT: Before and after insertion of data**



**2. Write a program to demonstrate Prepared Statement in Spring JdbcTemplate.**

**Employee.java**

**package** spring;

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

**private** **int** id;

**private** String name;

**private** **float** salary;

**public** Employee(**int** id, String name, **float** salary) {

**this**.id = id;

**this**.name = name;

**this**.salary = salary;

}

**public** **int** getId() {

**return** id;

}

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

**this**.id = id;

}

**public** String getName() {

**return** name;

}

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

**this**.name = name;

}

**public** **float** getSalary() {

**return** salary;

}

**public** **void** setSalary(**float** salary) {

**this**.salary = salary;

}

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

System.***out***.println("Employee [id=" + id + ", name=" + name + ", salary=" + salary + "]");

}

}

**EmployeeDao.java**

**package** spring;

**import** spring.Employee;

**public** **interface** EmployeeDao {

**public** **int** saveEmployee(Employee e);

**public** **int** updateEmployee(Employee e);

}

**EmployeeDaoImpl.java**

**package** spring;

**import** java.sql.PreparedStatement;

**import** java.sql.SQLException;

**import** java.util.List;

**import** org.springframework.dao.DataAccessException;

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

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

**import** spring.EmployeeDao;

**import** spring.Employee;

**public** **class** EmployeeDaoImpl **implements** EmployeeDao {

**private** JdbcTemplate jdbcTempl;

**public** **void** setJdbcTempl(JdbcTemplate jdbcTempl) {

**this**.jdbcTempl = jdbcTempl;

}

@Override

**public** **int** saveEmployee(Employee e) {

String query="insert into employee values(?,?,?)";

**return** jdbcTempl.execute(query, **new** PreparedStatementCallback<Integer>() {

@Override

**public** Integer doInPreparedStatement(PreparedStatement ps)

**throws** SQLException, DataAccessException {

ps.setInt(1, e.getId());

ps.setString(2, e.getName());

ps.setFloat(3, e.getSalary());

**return** ps.executeUpdate();

}

});

}

@Override

**public** **int** updateEmployee(Employee e) {

String query="update employee set name=?,salary=? where id=?";

**return** jdbcTempl.execute(query, **new** PreparedStatementCallback<Integer>() {

@Override

**public** Integer doInPreparedStatement(PreparedStatement ps)

**throws** SQLException, DataAccessException {

ps.setString(1, e.getName());

ps.setFloat(2, e.getSalary());

ps.setInt(3, e.getId());

**return** ps.executeUpdate();

}

});

}

}

**applicationContext.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"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd"*>

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

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

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

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

<property name=*"password"* value=*"12345"* />

</bean>

<bean id=*"jdbcTempl"* class=*"org.springframework.jdbc.core.JdbcTemplate"*>

<property name=*"dataSource"* ref=*"ds"*></property>

</bean>

<bean id=*"edao"* class=*"spring.EmployeeDaoImpl"*>

<property name=*"jdbcTempl"* ref=*"jdbcTempl"*/>

</bean>

</beans>

**Employee\_Test.java**

**package** spring;

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

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

**import** spring.EmployeeDao;

**import** spring.EmployeeDaoImpl;

**import** spring.Employee;

**public** **class** Employee\_Test {

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

// Loading bean configuration file

ApplicationContext ctx=**new** ClassPathXmlApplicationContext("applicationContext.xml");

EmployeeDao dao=(EmployeeDao)ctx.getBean("edao");

// Inserting into bean

**int** insert1 = dao.saveEmployee(**new** Employee(101,"Deepak",45000));

**int** insert2 = dao.saveEmployee(**new** Employee(102,"Rahul",35000));

System.***out***.println("\nChecking if record inserted?");

System.***out***.println((insert1>0) ? "1st Record Inserted" : "1st Record Not Inserted");

System.***out***.println((insert2>0) ? "2nd Record Inserted" : "2nd Record Not Inserted");

**int** update1 = dao.updateEmployee(**new** Employee(101,"Divya Mhatre",45000));

**int** update2 = dao.updateEmployee(**new** Employee(102,"Poonam Naik",35000));

System.***out***.println("\nChecking if record updated?");

System.***out***.println((update1>0) ? "1st Record Updated" : "1st Record Not Updated");

System.***out***.println((update2>0) ? "2nd Record Updated" : "2nd Record Not Updated");

}

}

**OUTPUT:-**

