PRACTICAL-1

Assignments on Java Generics

AIM:Write a Java Program to demonstrate a Generic Class.

CODE:

```
package anikettt;
import java.util.*;

class Bankaccount<T>
{
T a1;
public Bankaccount(T a1) // using constructor
{
this.a1=a1;
}

void getobject() // print the value of a1
{
System.out.println(a1);
}
}

public class genric {

public static void main(String[] args)
{
String s="Aniket";
Bankaccount<String> obj=new Bankaccount<String>(s);
obj.getobject();
Integer s1=5;
Bankaccount<Integer> obj2 =new Bankaccount<Integer>(s1);
obj2.getobject();
Double s3=5.23;
Bankaccount<Double> obj3 =new Bankaccount<Double>(s3);
obj3.getobject();
}

}
```

OUTPUT:

```
Aniket
5
5.23
```

AIM : Write a Java Program to demonstrate Generic Methods.

CODE:

```
package aniket;
class demo
{
public <T> void genericdemo(T num)
{
System.out.println("this is a generic method");
System.out.println(" the value = " + num);
}
}

public class genericmeth
{

public static void main(String[] args)
{

demo a1= new demo();
a1.<String>genericdemo("Aniket Raul");
a1.<Integer>genericdemo(35);

}

}
```

OUTPUT:

```
this is a generic method
 the value = Aniket Raul
this is a generic method
 the value = 35
```

generic method(Array)

generic method(Array)

```
package aniket;
class Bankaccount1<T>
{
T a1;
public <T> void deposit(T [ ] money)
{
for (T i: money)
{
System.out.println("money : "+ i);
}
}

}
public class genricddep {

public static void main(String[] args) {
Bankaccount1<Integer> obj5 =new Bankaccount1<>( );
Integer[] m1= {23,45,56};
Double[] m2= {23.45,45.45,56.26};
obj5.deposit(m1);
obj5.deposit(m2);

}

}
```

OUTPUT:

```
money : 23
money : 45
money : 56
money : 23.45
money : 45.45
money : 56.26
```

[Type here]

AIM : Write a Java Program to demonstrate Wildcards in Java Generics

CODE:

generic method(Upperbound)

```
package aniket;
import java.util.*;
public class upperbound<T>
{
private static Number summation(List<? extends Number>numbers)
{
Double sum=0.0;
for(Number n : numbers)
{
sum +=n.doubleValue();
}
return sum;
}
public static void main(String[] args)
{
List<Integer> list1=Arrays.asList(1,2,3,4,5);
System.out.println(" sum of the integr number : " +
summation(list1));

List<Double> list2=Arrays.asList(12.3,22.3,3.6,4.8,5.0);
System.out.println(" sum of the Double number : " +
summation(list2));
}

}
```

OUTPUT:

```
sum of the integr number : 15.0
sum of the Double number : 48.0
```

[Type here]

CODE:

generic method(Lowerbound)

```
package aniket;

import java.util.Arrays;
import java.util.List;

public class lowerbound {

public static void main(String[] args) {

List<Integer> list1=Arrays.asList(1,2,3,4,5);
System.out.println("integer list");
lowerbound(list1);

}
public static void lowerbound(List<? super Integer>list)
{
System.out.println(list);
}

}
```

OUTPUT:

```
integer list
[1, 2, 3, 4, 5]
```

PRACTICAL-2

Assignments on List Interface

AIM: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 anikettt;
import java.util.ArrayList;
import java.util.List;

public class genric {
public static void main(String[] args) {
// Create a List of Strings
List<String> stringList = new ArrayList<>();

// Add items to the list
stringList.add("Aniket");
stringList.add("vinay");
stringList.add("lalit");
stringList.add("Raj");

// Use a for-each loop to print each item
System.out.println("Items in the list:");
for (String item : stringList) {
System.out.println(item);
}
}
}
```

OUTPUT:

```
Servers  Terminal  Data Source Explorer  Properties
<terminated> genric [Java Application] C:\Users\ANIKET S RAUL\.
Items in the list:
Aniket
vinay
lalit
Raj
```

[Type here]

AIM:

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 anikettt;

import java.util.*;

public class genric {

public static void main(String[] args) {
LinkedList<String> list1= new LinkedList<String>();//creating an arraylist
list1.add("Tata");// add the element in the Linkedlist
list1.add("Mahindra");
list1.add("Suzuki");
list1.add("Kia");
list1.add( "Audi");
list1.addFirst("Nissan");//add element at first place of list
list1.addLast("Toyota");// Add element in the last position
System.out.println(list1);
System.out.println("Print the size of list " + list1.size());
System.out.println("Print the list in reverse order");
Iterator<String> itr= list1.descendingIterator();// reverse order printing in list
while(itr.hasNext())
{
System.out.println(itr.next());

}
}
}
```

```
<terminated> genric [Java Application] C:\Users\ANIKET S RAUL\.p2\pool\plugins\c
[Nissan, Tata, Mahindra, Suzuki, Kia, Audi, Toyota]
Print the size of list 7
Print the list in reverse order
Toyota
Audi
Kia
Suzuki
Mahindra
Tata
Nissan
```

PRACTICAL-3

Assignments on Set Interface

AIM: Write a Java program to create a Set containing list of items of type String and print the items in the list using Iterator interface. Also print the list in reverse/ backword direction.

CODE:

```
package anikettt;
import java.util.*;
public class genric {
public static void main(String[] args) {
Set<String> h1 = new LinkedHashSet<>();
h1.add("Aniket");
h1.add("vinay");
h1.add("sairaj");
Iterator i = h1.iterator();
while(i.hasNext())
System.out.println(i.next());
System.out.println("----Revrsed list----");
List<String> h2 = new ArrayList<>(h1);
Collections.reverse(h2);
Iterator i2 = h2.iterator();
while(i2.hasNext())
System.out.println(i2.next());
}
}
```

OUTPUT:

```
<terminated> genric [Java Application] C:\Users\ANIKET S RAUL\
Aniket
vinay
sairaj
----Revrsed list----
sairaj
vinay
Aniket
```

[Type here]

AIM : Write a Java program using Set interface containing list of items and perform the following operations:

a. Add items in the set.
b. Insert items of one set in to other set.
c. Search the specified item in the set
d. Remove item from set

CODE:

```
package anikettt;
import java.util.*;

public class genric {
public static void main(String[] args) {
Set<String> s1 = new HashSet<>();
// a. Add items to the set

s1.add("Aniket");
s1.add("Lalit");
s1.add("Sairaj");
System.out.println("Elements in the set are: " + s1);

// b. Insert items into another set
Set<String> s2 = new HashSet<>();
s2.addAll(s1);
System.out.println("Elements in the set 2 are: " + s2);

// c. Search for an item in the set
String searching = "Aniket";
if (s1.contains(searching)) {
System.out.println("Elements contain name: " + searching);
} else {
System.out.println("Does not contain name: " + searching);
}

// Add a new item to the set
String newItem = "John";
s1.add(newItem);
System.out.println("After adding '" + newItem + "', elements in
the set are: " + s1);
```

```
// d. Remove an item from the set
String itemToRemove = "Sairaj";
if (s1.remove(itemToRemove)) {
System.out.println("After removing '" + itemToRemove + "',
elements in the set are: " + s1);
} else {
System.out.println("Item '" + itemToRemove + "' not found in the
set.");
}
}
}
```

OUTPUT:

```
Servers  Terminal  Data Source Explorer  Properties  Console ×  Coverage
<terminated> genric [Java Application] C:\Users\ANIKET S RAUL\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.v
Elements in the set are: [Aniket, Sairaj, Lalit]
Elements in the set 2 are: [Aniket, Sairaj, Lalit]
Elements contain name: Aniket
After adding 'John', elements in the set are: [Aniket, John, Sairaj, Lalit]
After removing 'Sairaj', elements in the set are: [Aniket, John, Lalit]
```

PRACTICAL-4

Assignments on Map Interface

AIM: 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 anikettt;
import java.util.*;

public class genric {

public static void main(String[] args) {

// a. Add items to the map
Map<Integer, String> itemMap = new HashMap<>();
itemMap.put(1, "Aniket");
itemMap.put(2, "Sairaj");
itemMap.put(3, "Vinay");
itemMap.put(4, "RAJ");
itemMap.put(5, "LALIT");

// b. Print all keys and values of the map
System.out.println("Initial elements of the map:" + itemMap);


// c. Search for a specific key in the map
int keyToSearch = 2;
if (itemMap.containsKey(keyToSearch)) {
System.out.println("Key " + keyToSearch + " found in the map.");
} else {
System.out.println("Key " + keyToSearch + " not found in the
map.");
}
```

```java
// d. Get value of the specified key
int key = 1;
String value = itemMap.get(key);
System.out.println("Value for key " + key + ": " + value);

// e. Insert map elements of one map into another map
Map<Integer, String> m2 = new HashMap<>();
m2.putAll(itemMap);
System.out.println("All elements of itemMap in anotherMap: " + m2);

// f. Remove an item from the map
int keyToRemove = 3;
if (itemMap.containsKey(keyToRemove))
{
itemMap.remove(keyToRemove);
System.out.println("After removing key " + keyToRemove + ",
elements in the map:" + itemMap);
} else {
System.out.println("Key " + keyToRemove + " not found in the
map.");
}

// print all keys and values of a map
System.out.println("All keys in the map:");
for (Integer key1 : itemMap.keySet()) {
System.out.println(key1);
}

}
}
```

OUTPUT:

```
<terminated> genric [Java Application] C:\Users\ANIKET S RAUL\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17
Initial elements of the map:{1=Aniket, 2=Sairaj, 3=Vinay, 4=RAJ, 5=LALIT}
Key 2 found in the map.
Value for key 1: Aniket
All elements of itemMap in anotherMap: {1=Aniket, 2=Sairaj, 3=Vinay, 4=RAJ, 5=LALIT}
After removing key 3, elements in the map:{1=Aniket, 2=Sairaj, 4=RAJ, 5=LALIT}
All keys in the map:
1
2
4
5
```

PRACTICAL-5

Assignments on Lambda Expression

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

```
package anikettt;

public class map2 {

interface hello
{
public void display();
}
public static void main(String[] args)
{
hello r=()->System.out.println("hello world");
r.display();
}

}
```

Output:

```
Servers  Terminal  Data Source Explorer  Properties
<terminated> map2 [Java Application] C:\Users\ANIKET S RAUL\.
hello world
```

[Type here]

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

```
package anikettt;

import java.util.*;

public class map2 {
interface inter {
void myMethod(int x);
}

public static void main(String[] args) {
// Using lambda expression with a single parameter
inter a1 = (x) -> {
System.out.println("Value passed: " + x);
// Add your logic here based on the parameter
};

// Calling the lambda expression
a1.myMethod(10);
}
}
```

```
<terminated> map2 [Java Application] C:\Users\ANIKET S RAUL\.
Value passed: 10
```

[Type here]

AIM: Write a Java program using Lambda Expression with multiple parameters to add two numbers.

```
package anikettt;

public class map2 {
interface inter {
void myMethod(int x, int y);
}

public static void main(String[] args) {
// Using lambda expression with two parameters
inter a1 = (x, y) -> {
int sum = x + y;
System.out.println("Sum of " + x + " and " + y + " is: " + sum);
// Add your logic here based on the parameters
};

// Calling the lambda expression with two values
a1.myMethod(10, 5);
}
}
```

OUTPUT

```
<terminated> map2 [Java Application] C:\Users\ANIKET S RAUL\.p2\
Sum of 10 and 5 is: 15
```

[Type here]

AIM: Write a Java program using Lambda Expression to calculate the following:
a. Convert Fahrenheit to Celcius
b. Convert Kilometers to Miles.

```
package anikettt;
public class map2 {
// Functional interface for temperature conversion
interface TemperatureConverter {
double convert(double fahrenheit);
}

// Functional interface for distance conversion
interface DistanceConverter {
double convert(double kilometers);
}

public static void main(String[] args) {

// a. Convert Fahrenheit to Celsius
TemperatureConverter fc = (f) -> (f - 32) * 5 / 9;

double fahrenheitValue = 98.6; // Example Fahrenheit temperature
double celsiusValue = fc.convert(fahrenheitValue);
System.out.println(fahrenheitValue + " Fahrenheit is equal to " +
celsiusValue + " Celsius.");

// b. Convert Kilometers to Miles
DistanceConverter kms = (km) -> km * 0.621371;

double kilometersValue = 10; // Example distance in kilometers
double milesValue = kms.convert(kilometersValue);
System.out.println(kilometersValue + " Kilometers is equal to " + milesValue
+ " Miles.");
}
}
```

OUTPUT:

```
<terminated> map2 [Java Application] C:\Users\ANIKET S RAUL\.p2\pool\p
98.6 Fahrenheit is equal to 37.0 Celsius.
10.0 Kilometers is equal to 6.21371 Miles.
```

[Type here]

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

```
package anikettt;

public class map2 {
// Updated Functional interface with a generic method
interface Greeting<T> {
T greet(String name);
}

public static void main(String[] args) {
// Lambda expression without return keyword for greeting
Greeting<Void> gm = (name) -> {
System.out.println("Hello, " + name + "!");
return null; // Return null for void
};
gm.greet("Aniket");

// Lambda expression with return keyword for greeting
Greeting<String> gm2 = name -> {
String message = "Hello, " + name + "!";
System.out.println(message);
return message;
};

// Call the lambda expressions
gm2.greet("Sairaj");
}
}
```

```
<terminated> map2 [Java Application] C:\Users\ANIKET S RAUL\.p2\po
Hello, Aniket!
Hello, Sairaj!
```

[Type here]

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

```
package anikettt;
public class map2 {
interface concate {
String concatenate(String str1, String str2);
}

public static void main(String[] args) {
// Lambda expression for concatenating two strings
concate conca = (s1, s2) -> s1 + s2;

// Strings to concatenate
String firstString = "Hello, ";
String secondString = "Aniket!";

// Use the lambda expression to concatenate strings
String result = conca.concatenate(firstString, secondString);

// Print the result
System.out.println(result);
}
}
```

```
<terminated> map2 [Java Application] C:\Users\ANIKET S RAUL\.p2\
Hello, Aniket!
```

[Type here]

PRACTICAL-6

Assignments based on web application development using JSP

A) **AIM:** Write Programs to demonstrate different Implicit Objects

a. OUT
b. Request
c. Session

```
<%@ 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>
<h1>Out Object</h1>
<% out.println("Luffy : This is... a love ordeal");%>

<h1>Request Object</h1>
<%
String uri = request.getRequestURI();
out.println("Requested URI: " + uri);
%>

<h1>Session Object</h1>
<%
session.setAttribute("luffy", "I refuse your refusal");
String attribute = (String) session.getAttribute("luffy");
out.println("The value of the session attribute 'attribute' is: " + attribute);
%>
</body>
</html>
```

Output :

Out Object

Luffy : This is... a love ordeal

Request Object

Requested URI: /anikettt/aniket.jsp

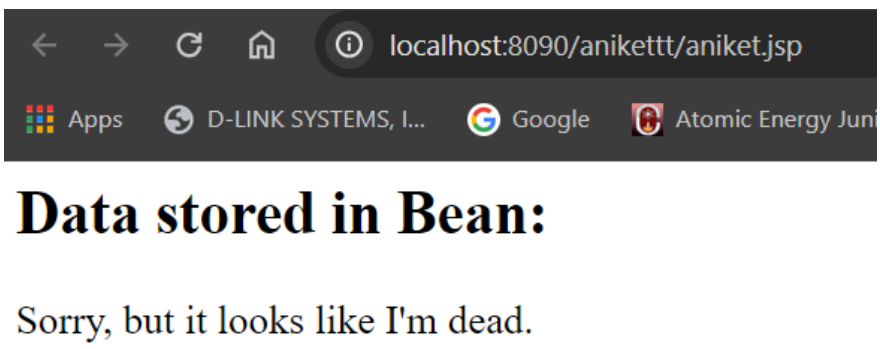
Session Object

The value of the session attribute 'attribute' is: I refuse your refusal

B) **AIM :**Write Programs to demonstrate temporary storage using Bean.

```
<%@ page import="java.util.ArrayList" %>
<jsp:useBean id="myBean" class="anikettt.MyBean" scope="request"/>
<%
// Set data in the bean
myBean.setData("Sorry, but it looks like I'm dead.");

// Retrieve data from the bean
String data = myBean.getData();
%>
<html>
<head><title>Temporary Storage Using Bean</title></head>
<body>
<h2>Data stored in Bean:</h2>
<p><%= data %></p>
</body>
</html>
```

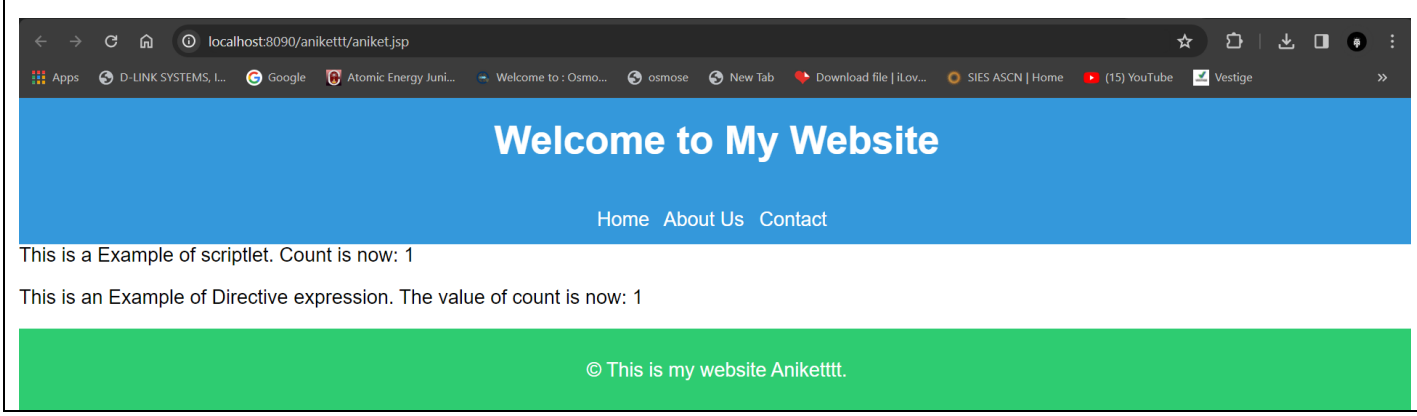
Output :



C) **AIM:** Write a program to demonstrate Standard 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>Practical 7</title>
</head>
<body>
<body>
<%@ include file="header.jsp" %> <!-- Directive to include header -->

<%-- JSP Declaration --%>
<%! int count = 0; %>
<%-- JSP Scriptlet --%>
<%
count++;
out.println("This is a Example of scriptlet. Count is now: " + count);
%>

<%-- JSP Expression --%>
<p>This is an Example of Directive expression. The value of count is now: <%= count %></p>
<%@ include file="footer.jsp" %> <!-- Directive to include footer -->
</body>
</body>
</html>
```

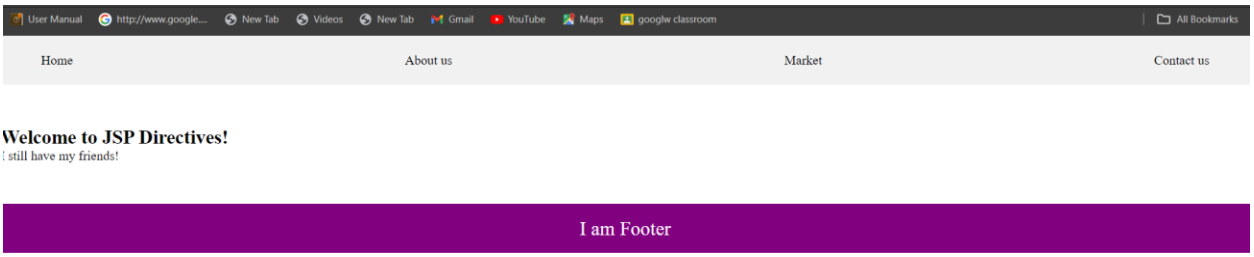
Output:



D) AIM : Write a program to demonstrate JSP Directives

```
<%@ 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="${'HEllo ANiket!'}"/>
    <%@ include file="footer.jsp" %>
</body>
</html>
```

Output :



E) AIM :Write a program to demonstrate Session Tracking using Cookies

```
<%@ 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:luffy");

   // Create a cookie for the username

   Cookie usernameCookie = new Cookie("username",
   "Cookie:Luffy"); 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 :

Session Tracking Using Cookies

Username stored in session: Session:luffy

Username stored in cookie: Cookie:Luffy

F) AIM : Write a program to demonstrate JSTL Tags

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

     <html>
     <head>
        <title>JSTL Demo</title>
     </head>
     <body>
        <h2>JSTL Core Tags Demo</h2>


        <c:set var="message" value="I love heroes, but I don't
        want to be one."
     />
        <p>Message: <c:out value="${message}" /></p>


        <c:if test="${5 > 3}">
           <p>The condition is true.</p>
        </c:if>

        <c:forEach var="i" begin="1" end="5">
           <p>Number: ${i}</p>
        </c:forEach>


     </body>
     </html>
```

Output :

JSTL Core Tags Demo

Message: I love heroes, but I don't want to be one.

The condition is true.

Number: 1

Number: 2

Number: 3

Number: 4

Number: 5

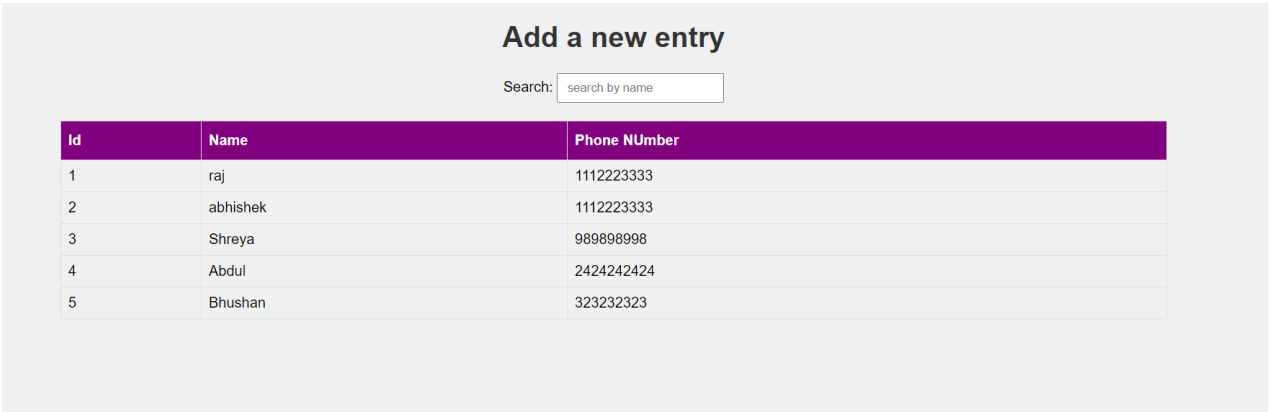
JSTL Formatting Tags Demo

G) AIM :Create a Telephone directory using JSP and store all the information within a database, so that later could be retrieved as per the requirement. Make your own assumptions.

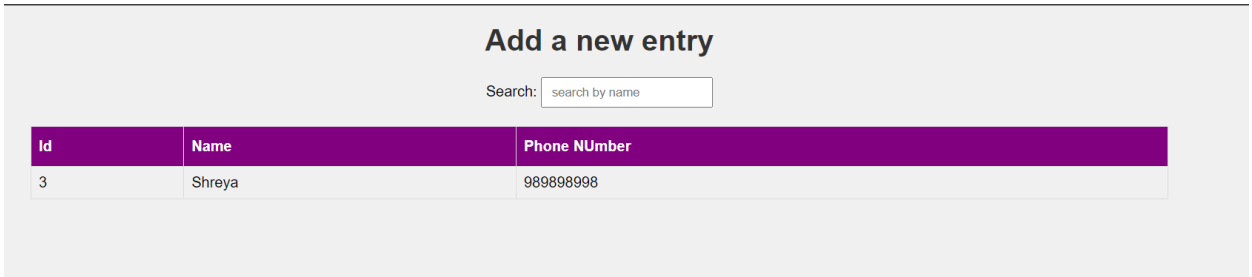
```
<%@ page import = "java.io.*,java.util.*,java.sql.*"%>
<%@ page import = "javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix =
"c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix =
"sql"%>


<%@ page language="java" contentType="text/html; charset=ISO-
8859-1"
   pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Practical 1</title>
</head>
<body>
     <h1>Add a new entry</h1>
   <form method="get">
      <label for="search">Search:</label>
      <input type="text" id="search" name="search"
      placeholder="search by
name">
   </form>
   <sql:setDataSource var = "snapshot" driver =
   "com.mysql.jdbc.Driver"
       url = "jdbc:mysql://localhost:3306/mcaraj"
       user = "root"  password = "root"/>


   <sql:query dataSource = "${snapshot}" var
       = "result"> SELECT * from telephone
       where name LIke ?;
       <sql:param value ="%${param.search}%"/>
</sql:query>
<table border = "1" width = "100%">
       <tr>
         <th>Id</th>
         <th>Name</th>
         <th>Phone NUmber </th>
       </tr>
```

```
        <c:forEach var = "row" items = "${result.rows}">
          <tr>
            <td><c:out value = "${row.id}"/></td>
            <td><c:out value = "${row.name}"/></td>
            <td><c:out value = "${row.phoneNumber}"/></td>
          </tr>
        </c:forEach>
      </table>

    </body>
  </body>
  </html>
```

Output



Id	Name	Phone NUmber
1	raj	1112223333
2	abhishek	1112223333
3	Shreya	989898998
4	Abdul	2424242424
5	Bhushan	323232323



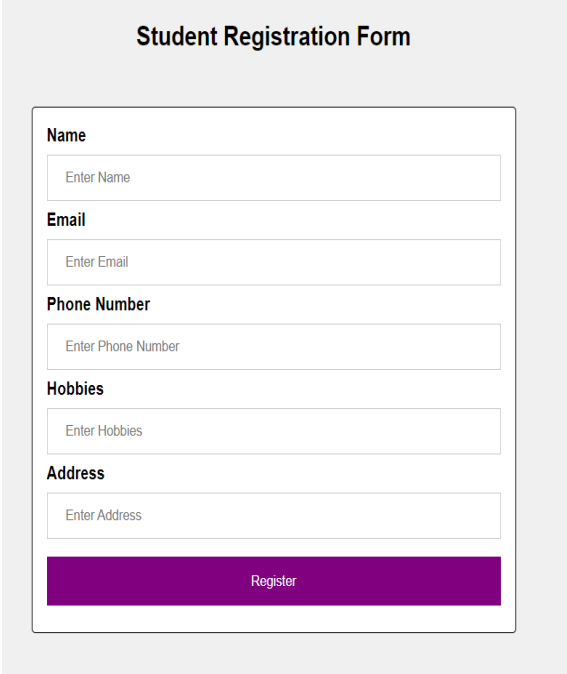
Id	Name	Phone NUmber
3	Shreya	989898998

H) AIM : Write a JSP page to display the Registration form (Make your own assumptions)

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Practical 2</title>
</head>
<body>
<h2>Student Registration Form</h2>

<div class="container">
        <label for="name"><b>Name</b></label>
        <input type="text" placeholder="Enter Name" name="name"
        required>

        <label for="email"><b>Email</b></label>
        <input type="text" placeholder="Enter Email" name="email"
        required>

        <label for="phone"><b>Phone Number</b></label>
        <input type="text" placeholder="Enter Phone
    Number" name="phone" required>

        <label for="hobbies"><b>Hobbies</b></label>
        <input type="text" placeholder="Enter Hobbies"
        name="hobbies" required>
```

```
<label for="address"><b>Address</b></label>
        <input type="text" placeholder="Enter Address"
        name="address" required>


        <button type="submit">Register</button>
    </div>
    </body>
    </html>
```

Output :

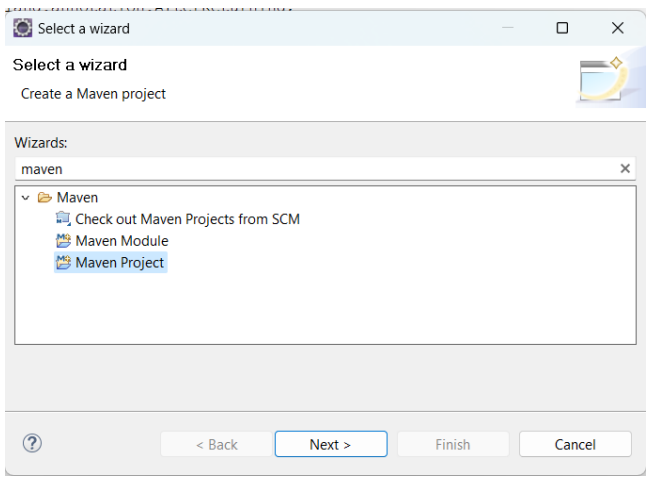


PRACTICAL 7: ASSIGNMENT BASED SPRING FRAMEWORK

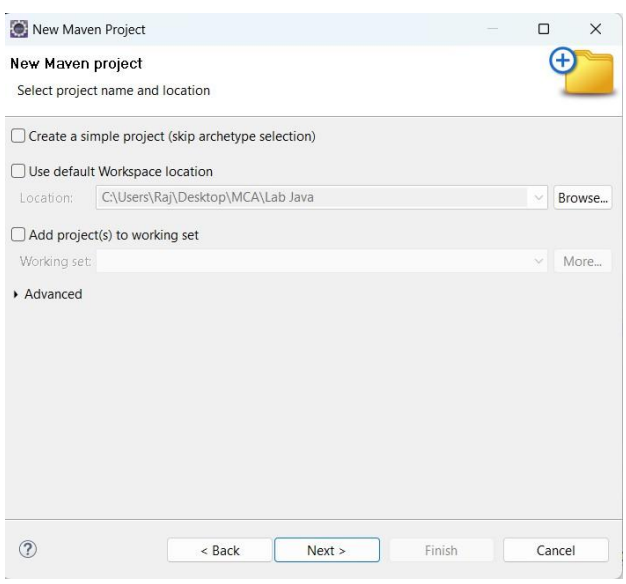
A) AIM : Write a program to print Singer Name and Age using spring framework.

Maven project

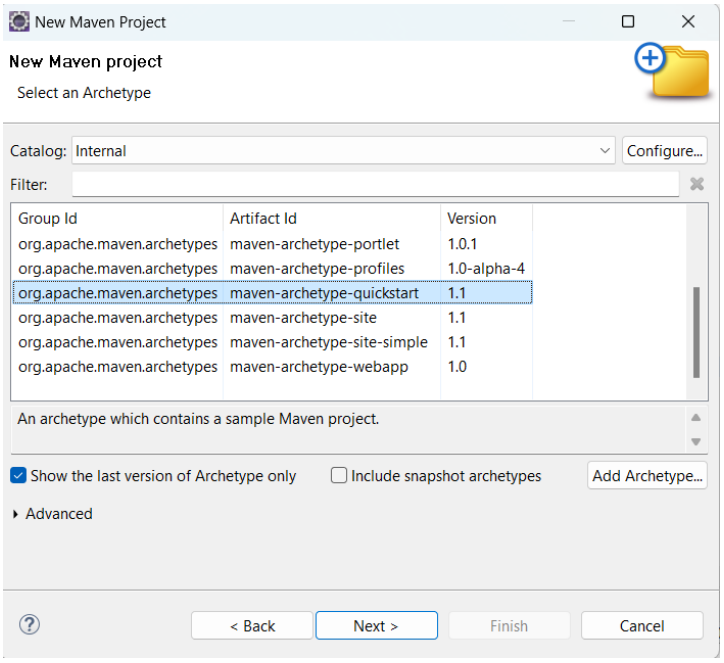
1) Open Eclipse IDE, Navigate to File, then New, then Others., Select Maven Project, Click on the "Next" button.



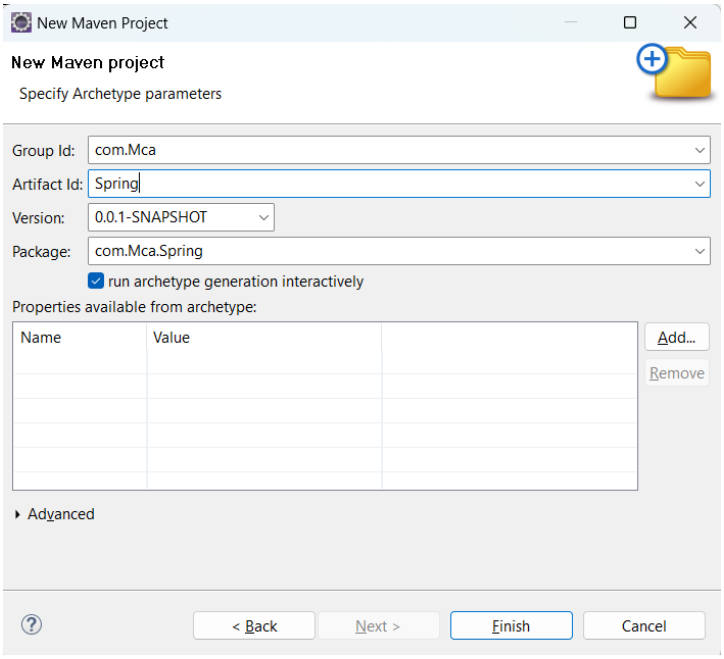
2) Check the option 'use default workspace location or choose your desired workspace location.



3) Select catalog Internal the archetype 'maven-archetype-quickstart'.Click on the "Next" button



4) Enter your project's Group Id. Enter your project's Artifact Id. Click on the "Finish" button.



[Type here]

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/be
ans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:context="http://www.springframework.org/s
chema/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/spr
ing-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/s
pring-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.ClassPathXmlApplicationCo
ntext;


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 :

```
Console ×
<terminated> test [Java Application] C:\Users\Raj\.p2\poc
Singer [name=Luffy, age=19]
```

B) AIM : 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;

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;
          }
public Zoro(String name, double height, int swords) {
               super();
               this.name =
               name;
               this.height = height;
               this.swords = swords;
          }
          public Zoro() {
               super();
          }
@Override
          public String toString() {
               return "name of Character = " + name + ", height of
               Character = "
     + height + ", No. of swords = " + swords ;
          }
     }
```

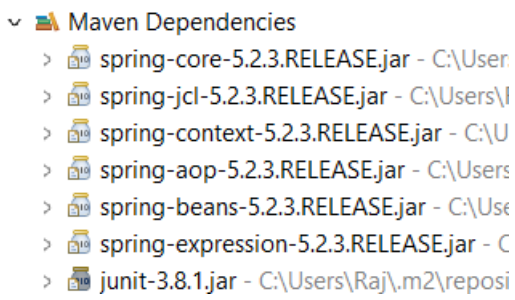
Configuration xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans
xmlns="http://www.springframework.org/schema/be
ans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:context="http://www.springframework.org/s
chema/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/spr
ing-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/s
pring-context.xsd">


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

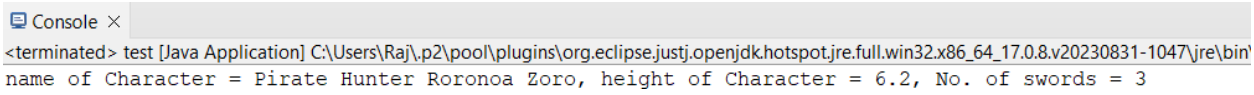
Main class

```
package MCA;
     import org.springframework.context.ApplicationContext;
     import
     org.springframework.context.support.ClassPathXmlApplicationCo
     ntext;

     public class test {
         public static void main(String[] args) {

             ApplicationContext context = new
     ClassPathXmlApplicationContext("MCA/mca
             Config.xml"); Zoro temp = (Zoro)
             context.getBean("zoro");
             System.out.println(temp);
         }
     }
```

Maven Dependencies



Output :




```
name of Character = Pirate Hunter Roronoa Zoro, height of Character = 6.2, No. of swords = 3
```

C) AIM : 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/be
ans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:context="http://www.springframework.org/s
chema/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/spr
ing-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/s
pring-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.ClassPathXmlApplicationCo
ntext;


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 :

```
Console ×
<terminated> test [Java Application] C:\Users\Raj\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_1
Charactername = Monkey D. Luffy, No. of gears = 5, height = 5.8]
```

D) AIM : 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();
          }
          @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;
   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;
          }


          public Zoro(String name, double height, int swords) {
               super();
               this.name =
               name;
               this.height = height;
               this.swords = swords;
          }
          public Zoro() {
               super();
          }
     @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/be
ans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:context="http://www.springframework.org/s
chema/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/spr
ing-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/s
pring-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.ClassPathXmlApplicationCo
ntext;


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 :

```
Console ×
<terminated> test [Java Application] C:\Users\Raj\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe
sanji [name=Vinsmoke Sanji, height=6.0,
obj=name of Character = Pirate Hunter Roronoa Zoro, height of Character = 6.2, No. of swords = 3]
```

E) AIM : 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/be
ans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:context="http://www.springframework.org/s
chema/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/spr
ing-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/s
pring-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.ClassPathXmlApplicationCo
    ntext;

    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 :

```
Console ×
<terminated> test [Java Application] C:\Users\Raj\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230
ussop [Name=Sogeking Ussop, height=5.11,
obj= Charactername = Monkey D. Luffy, No. of gears = 5, height = 5.8]]
```

F) AIM : 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/be
ans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:context="http://www.springframework.org/s
chema/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/spr
ing-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/s
pring-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.ClassPathXmlApplicationCo
     ntext;


     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 :

```
Console ×
<terminated> test [Java Application] C:\Users\Raj\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe  (21 Dec 2023, 18:14:29 – 18:14:29)
strawHat [name=The Straw Hat Pirates,
crewName=[Monkey D. Luffy, Roronoa Zoro, First son of sea Jimbei, Vinksmoke Sanji, Demon child Nico Robin],
bounty=[3,000,000,000, 1,200,000,000, 1,100,000,000, 1,032,000,000, 930,000,000],
ability={luffy=rubber body, zoro=swordsman, jimbei=Helmsman, sanji=cook, robin=archaeologist}]
```

G) AIM : 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;
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;
          }



          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/be
ans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:context="http://www.springframework.org/s
chema/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/spr
ing-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/s
pring-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.ClassPathXmlApplicationCo
     ntext;


     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 :

```
Console ×
<terminated> test [Java Application] C:\Users\Raj\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe (21 Dec 2023, 19:48:35 – 19:48:35) [pid: 6564]
chopper
[Zoro=name of Character = Pirate Hunter Roronoa Zoro, height of Character = 6.2, No. of swords = 3]
```

PRACTICAL 8:

ASSIGNMENT BASED ASPECT ORIENTED PROGRAMMING

A) AIM : 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.AfterRet
urning; import
org.aspectj.lang.annotation.AfterThr
owing; 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/schem
a/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:context="http://www.springframework.or
g/schema/context"
xmlns:aop="http://www.springframework.org/sc
hema/aop"

xsi:schemaLocation="http://www.springframewor
k.org/schema/beans
http://www.springframework.org/schema/beans/s
pring-beans.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spr
ing-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.ClassPathXmlApplicationCo
ntext;

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 :

```
Console ×
<terminated> App [Java Application] C:\Users\Raj\.p2\pool\plug
Yahoo Yahoo : I am before Aspect
Song Started
Song Ended
```

Write a program to demonstrate Spring AOP – after advice.

Aspect class

```
package aop;

import org.aspectj.lang.annotation.After;
import
org.aspectj.lang.annotation.AfterRet
urning; import
org.aspectj.lang.annotation.AfterThr
owing; 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 :

```
Console ×
<terminated> App [Java Application] C:\Users\Raj\.p2\poo
Song Started
Song Ended
Yahoo Yahoo : I am After Aspect
```

B) AIM :Write a program to demonstrate Spring AOP – around advice.

Aspect class

```
package aop;

import org.aspectj.lang.annotation.After;
import
org.aspectj.lang.annotation.AfterRet
urning; import
org.aspectj.lang.annotation.AfterThr
owing; 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 :

```
Console ×
<terminated> App [Java Application] C:\Users\Raj\.p2\p
Yahoo Yahoo : Around Aspect
```

C) AIM : 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.AfterRet
urning; import
org.aspectj.lang.annotation.AfterThr
owing; import
org.aspectj.lang.annotation.Around;
import
org.aspectj.lang.annotation.Aspe
ct; import
org.aspectj.lang.annotation.Befo
re; import
org.aspectj.lang.annotation.Poin
tcut;
@Aspect
public class mcaAspect {
@AfterReturning("execution(* brook.makeSong())")
    public void AfterReturnSong() {
        System.out.println("Yahoo Yahoo : After Returing
        Aspect");
    }


}
```

Output :

```
Console ×
<terminated> App [Java Application] C:\Users\Raj\.p2\pool\plugins\c
Song Started
Song Ended
Yahoo Yahoo : After Returing Aspect
```

D) AIM :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.AfterRet
urning; import
org.aspectj.lang.annotation.AfterThr
owing; import
org.aspectj.lang.annotation.Around;
import
org.aspectj.lang.annotation.Aspe
ct; import
org.aspectj.lang.annotation.Befo
re; import
org.aspectj.lang.annotation.Poin
tcut;

@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: ");
   }

}
```

[Type here]

Output :

```
<terminated> App [Java Application] C:\Users\Raj\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe  (25 Dec 2023, 14:12:23 – 14:
Song Started
Song Ended
Yahoo Yahoo : There has been an exception:
Exception in thread "main" java.lang.IllegalArgumentException: An error occurred while making the song.
        at aop.brook.makeSong(brook.java:10)
        at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:77)
        at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.base/java.lang.reflect.Method.invoke(Method.java:568)
        at org.springframework.aop.support.AopUtils.invokeJoinpointUsingReflection(AopUtils.java:344)
        at org.springframework.aop.framework.ReflectiveMethodInvocation.invokeJoinpoint(ReflectiveMethodInvocation.java:198)
        at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:163)
        at org.springframework.aop.aspectj.AspectJAfterThrowingAdvice.invoke(AspectJAfterThrowingAdvice.java:62)
        at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
        at org.springframework.aop.interceptor.ExposeInvocationInterceptor.invoke(ExposeInvocationInterceptor.java:95)
        at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
        at org.springframework.aop.framework.JdkDynamicAopProxy.invoke(JdkDynamicAopProxy.java:212)
        at jdk.proxy2/jdk.proxy2.$Proxy10.makeSong(Unknown Source)
```

E) AIM :Write a program to demonstrate Spring AOP – pointcuts.

Aspect class

```
package aop;

import org.aspectj.lang.annotation.After;
import
org.aspectj.lang.annotation.AfterRet
urning; import
org.aspectj.lang.annotation.AfterThr
owing; import
org.aspectj.lang.annotation.Around;
import
org.aspectj.lang.annotation.Aspe
ct; import
org.aspectj.lang.annotation.Befo
re; import
org.aspectj.lang.annotation.Poin
tcut;

@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 :

```
Console ×
<terminated> App [Java Application] C:\Users\Raj\.p2\pool
Song Started
Song Ended
Yahoo Yahoo : Used BY Pointcut
```

PRACTICAL 9 : ASSIGNMENT BASED SPRING JDBC

A) AIM :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/XMLS
chema-instance"
xmlns:context="http://www.springframew
ork.org/schema/context"
xmlns:p="http://www.springframework.or
g/schema/p"
xmlns:c="http://www.springframework.or
g/schema/c"
xsi:schemaLocation="http://www.springf
ramework.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.DriverManagerDataS
ource" 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.ClassPathXmlApplicationCo
      ntext;
      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:

Result Grid | Filter Rows: | Edit:

id	name	bounty
1	luffy	3 Billion
5	dummy	entry
NULL	NULL	NULL

Console

```
<terminated> App (1) [Java Application] C:\Users\Raj\.p2\pc
kaizokuo ni ore wa naru!
Loading class `com.mysql.jdbc.Driver'. This is dep
Number of records insetred 1
Number of records updated 1
Number of records Deleted 1
```

Result Grid | Filter Rows:

id	name	bounty
1	luffy	4 billion
2	zoro	1.2 Billion
NULL	NULL	NULL

B) AIM :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:

Result Grid | Filter Rows:

id	name	bounty
1	luffy	4 billion
2	zoro	1.2 Billion
NULL	NULL	NULL

Console

```
<terminated> App (1) [Java Application] C:\Users\Raj\.p2
kaizokuo ni ore wa naru!
Loading class `com.mysql.jdbc.Driver'. This is
Number of rows afftected 1
```

Result Grid | Filter Rows:

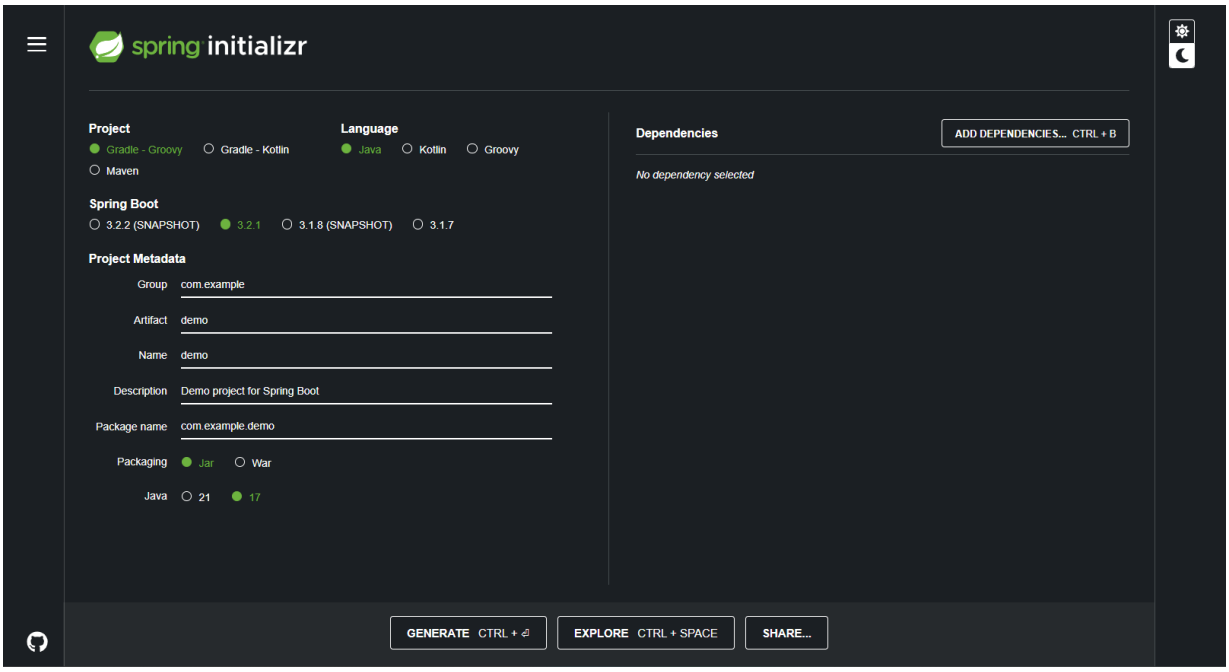
id	name	bounty
1	luffy	4 billion
2	zoro	1.2 Billion
3	zoro	1.1 Billion
NULL	NULL	NULL

PRACTICAL 10:

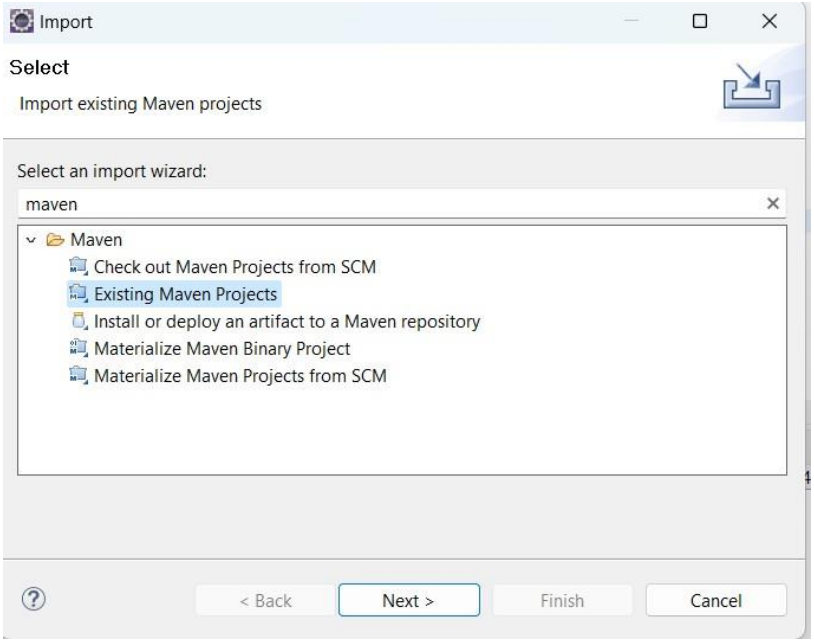
ASSIGNMENT BASED SPRING BOOT AND RESTFUL WEB SERVICES

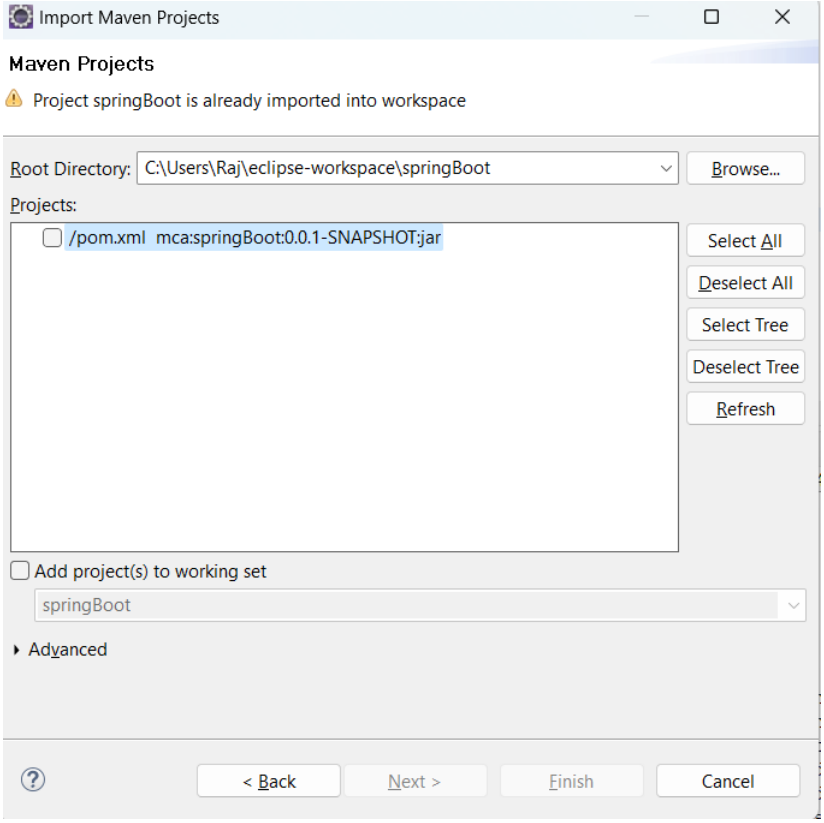
A)Write a program to create a simple Spring Boot application prints a message

1) Go to Spring Initializr. 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".



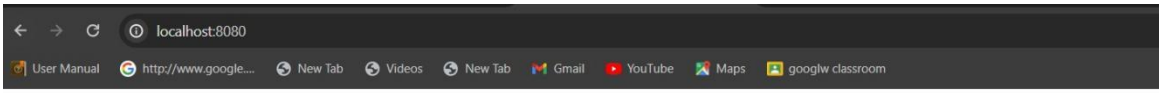


Main class

```
package com.mca.spring;

import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplicat
ion;
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
      { @GetMappin
      g("/")
      public
      String
      quote() {
return "I am Spring Boot . Let’s ROCK";
     }
      }
   }
```

Output :

```
Servers  Console ×
myApplication [Java Application] C:\Users\Raj\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe  (23 Dec 2023, 12:59:03) [pid: 23520]

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v3.2.1)

2023-12-23T12:59:03.913+05:30  INFO 23520 --- [           main] com.mca.spring.myApplication             : Starting myApplication using Java 17.0.8.1
2023-12-23T12:59:03.916+05:30  INFO 23520 --- [           main] com.mca.spring.myApplication             : No active profile set, falling back to 1 d
2023-12-23T12:59:04.620+05:30  INFO 23520 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port 8080 (http)
2023-12-23T12:59:04.630+05:30  INFO 23520 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2023-12-23T12:59:04.630+05:30  INFO 23520 --- [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10
2023-12-23T12:59:04.691+05:30  INFO 23520 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicatio
2023-12-23T12:59:04.692+05:30  INFO 23520 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization
2023-12-23T12:59:04.989+05:30  INFO 23520 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with co
2023-12-23T12:59:04.997+05:30  INFO 23520 --- [           main] com.mca.spring.myApplication             : Started myApplication in 1.366 seconds (pr
2023-12-23T12:59:21.995+05:30  INFO 23520 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dis
2023-12-23T12:59:21.996+05:30  INFO 23520 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
2023-12-23T12:59:21.996+05:30  INFO 23520 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 0 ms
```



I am Spring Boot. Let's Rock