1. Assignments on Java Generics

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

```
class GenericClass<T> {
    private T data;

    public GenericClass(T data) {
        this.data = data;
    }
    public T getData() {
        return data;
    }
    public void setData(T data) {
        this.data = data;
    }
    public void display() {
        System.out.println("Data: " + data);
    }
}
public class Ass1q1 {
    public static void main(String[] args) {
        GenericClass<Integer> intObject = new GenericClass<>(42);
        intObject.display();
        System.out.println("Retrieved data: " + intObject.getData());

        GenericClass<String> stringObject = new GenericClass<>("Hello, Generics!");
        stringObject.display();
        System.out.println("Retrieved data: " + stringObject.getData());

        GenericClass<Double> doubleObject = new GenericClass<>(3.14);
```

```
        doubleObject.display();
        System.out.println("Retrieved data: " + doubleObject.getData());
    }
}
```

Output:

```
PS C:\Users\Administrator\Desktop\Prem Java> javac Ass1q1.java
PS C:\Users\Administrator\Desktop\Prem Java> java Ass1q1
Data: 42
Retrieved data: 42
Data: Hello, Generics!
Retrieved data: Hello, Generics!
Data: 3.14
Retrieved data: 3.14
```

2. Write a Java Program to demonstrate Generic Methods.

```
class GenericMethodDemo {
    public static <T> void printArray(T[] array) {
        for (T element : array) {
            System.out.print(element + " ");
        }
        System.out.println();
    }
    public static <T extends Comparable<T>> T findMax(T first, T second) {
        return (first.compareTo(second) > 0) ? first : second;
    }
    public static void main(String[] args) {
        Integer[] intArray = {1, 2, 3, 4, 5};
        String[] stringArray = {"Java", "Generics", "Methods"};
        Double[] doubleArray = {1.1, 2.2, 3.3};
        System.out.println("Integer Array:");
        printArray(intArray);
        System.out.println("String Array:");
        printArray(stringArray);
        System.out.println("Double Array:");
        printArray(doubleArray);
        int maxInt = findMax(10, 20);
        System.out.println("Maximum Integer: " + maxInt);

        String maxString = findMax("Apple", "Orange");
        System.out.println("Maximum String: " + maxString);
    }
}
```

Output:

```
PS C:\Users\Administrator\Desktop\Prem Java> javac Ass1q2.java
PS C:\Users\Administrator\Desktop\Prem Java> java Ass1q2
Integer Array:
1 2 3 4 5
String Array:
Java Generics Methods
Double Array:
1.1 2.2 3.3
Maximum Integer: 20
Maximum String: Orange
```

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

```
import java.util.ArrayList;
import java.util.List;

class Ass1q3 {

    public static void printNumbers(List<? extends Number> numbers) {
        for (Number number : numbers) {
            System.out.println(number);
        }
    }

    public static void printList(List<?> list) {
        for (Object item : list) {
            System.out.println(item);
        }
    }

    public static void addElements(List<? super Integer> list) {
        for (int i = 1; i <= 5; i++) {
            list.add(i);
        }
    }

    public static void main(String[] args) {
        List<Integer> intList = new ArrayList<>();
        intList.add(10);
        intList.add(20);
        intList.add(30);
        System.out.println("Numbers (Upper Bounded Wildcard):");
```

```
        printNumbers(intList);

        List<Double> doubleList = new ArrayList<>();
        doubleList.add(1.1);
        doubleList.add(2.2);
        System.out.println("Doubles (Upper Bounded Wildcard):");
        printNumbers(doubleList);

        List<String> stringList = new ArrayList<>();
        stringList.add("Apple");
        stringList.add("Orange");
        System.out.println("Strings (Unbounded Wildcard):");
        printList(stringList);

        List<Number> numberList = new ArrayList<>();
        addElements(numberList);
        System.out.println("Numbers after adding elements (Lower Bounded Wildcard):");
        printList(numberList);
    }
}
```

Output:

```
PS C:\Users\Administrator\Desktop\Prem Java> java Ass1q3
Numbers (Upper Bounded Wildcard):
10
20
30
Doubles (Upper Bounded Wildcard):
1.1
2.2
Strings (Unbounded Wildcard):
Apple
Orange
Numbers after adding elements (Lower Bounded Wildcard):
1
2
3
4
5
```

2. 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**.

```
import java.util.ArrayList;

import java.util.List;


public class Ass2q1 {

    public static void main(String[] args) {

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


        stringList.add("Apple");

        stringList.add("Banana");

        stringList.add("Cherry");

        stringList.add("Date");

        stringList.add("Elderberry");


        System.out.println("Items in the list:");

        for (String item : stringList) {

            System.out.println(item);

        }

    }

}
```

Output:

```
PS C:\Users\Administrator\Desktop\Prem Java> javac Ass2q1.java
PS C:\Users\Administrator\Desktop\Prem Java> java Ass2q1
Items in the list:
Apple
Banana
Cherry
Date
Elderberry
```

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.**

```
import java.util.ArrayList;
import java.util.List;
import java.util.ListIterator;

public class Ass2q2 {
    public static void main(String[] args) {
        List<String> itemList = new ArrayList<>();

        itemList.add("Apple");
        itemList.add("Banana");
        itemList.add("Cherry");
        itemList.add("Date");
        itemList.add("Elderberry");

        ListIterator<String> iterator = itemList.listIterator();

        System.out.println("Items in forward direction:");
        while (iterator.hasNext()) {
            System.out.println(iterator.next());
        }

        System.out.println("\nItems in reverse direction:");
        while (iterator.hasPrevious()) {
            System.out.println(iterator.previous());
        }
    }
}
```

Output:

```
PS C:\Users\Administrator\Desktop\Prem Java> javac Ass2q2.java
PS C:\Users\Administrator\Desktop\Prem Java> java Ass2q2
Items in forward direction:
Apple
Banana
Cherry
Date
Elderberry

Items in reverse direction:
Elderberry
Date
Cherry
Banana
Apple
```

3. Assignments on Set Interface

1. **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**.

```
import java.util.HashSet;
import java.util.Set;
import java.util.Iterator;
import java.util.ArrayList;
import java.util.List;

public class Ass3q1 {
    public static void main(String[] args) {
        Set<String> itemSet = new HashSet<>();

        itemSet.add("Apple");
        itemSet.add("Banana");
        itemSet.add("Cherry");
        itemSet.add("Date");
        itemSet.add("Elderberry");

        System.out.println("Items in the Set:");
        Iterator<String> iterator = itemSet.iterator();
        while (iterator.hasNext()) {
            System.out.println(iterator.next());
        }

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

        System.out.println("\nItems in reverse order:");
        for (int i = itemList.size() - 1; i >= 0; i--) {
```

```
            System.out.println(itemList.get(i));
        }
    }
}
```

Output:

```
PS C:\Users\Administrator\Desktop\Prem Java> javac Ass3q1.java
PS C:\Users\Administrator\Desktop\Prem Java> java Ass3q1
Items in the Set:
Apple
Cherry
Date
Elderberry
Banana

Items in reverse order:
Banana
Elderberry
Date
Cherry
Apple
```

2. **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**

```
import java.util.HashSet;
import java.util.Set;

public class Ass3q2 {
    public static void main(String[] args) {
        Set<String> set1 = new HashSet<>();
        set1.add("Apple");
        set1.add("Banana");
        set1.add("Cherry");
        System.out.println("Set 1 after adding items: " + set1);

        Set<String> set2 = new HashSet<>();
        set2.add("Date");
        set2.add("Elderberry");
        System.out.println("Set 2: " + set2);

        set1.addAll(set2);
        System.out.println("Set 1 after inserting items from Set 2: " + set1);

        set1.remove("Banana");
        System.out.println("Set 1 after removing 'Banana': " + set1);

        String searchItem = "Cherry";
        if (set1.contains(searchItem)) {
            System.out.println("Set 1 contains the item: " + searchItem);
        } else {
            System.out.println("Set 1 does not contain the item: " + searchItem);
        }
```

```
    }
}
```

Output:

```
PS C:\Users\Administrator\Desktop\Prem Java> javac Ass3q2.java
PS C:\Users\Administrator\Desktop\Prem Java> java Ass3q2
Set 1 after adding items: [Apple, Cherry, Banana]
Set 2: [Date, Elderberry]
Set 1 after inserting items from Set 2: [Apple, Cherry, Date, Elderberry, Banana]
Set 1 after removing 'Banana': [Apple, Cherry, Date, Elderberry]
Set 1 contains the item: Cherry
```

4. Assignments on Map Interface

1. **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.**

```
import java.util.HashMap;

import java.util.Map;


public class Ass4q1 {

    public static void main(String[] args) {

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

        map1.put(1, "Apple");

        map1.put(2, "Banana");

        map1.put(3, "Cherry");

        System.out.println("Map 1 after adding items: " + map1);


        map1.remove(2);

        System.out.println("Map 1 after removing key 2: " + map1);


        int searchKey = 3;

        if (map1.containsKey(searchKey)) {

            System.out.println("Map 1 contains the key: " + searchKey);

        } else {

            System.out.println("Map 1 does not contain the key: " + searchKey);

        }


        int keyToGet = 1;

        String value = map1.get(keyToGet);

        if (value != null) {
```

```
            System.out.println("Value associated with key " + keyToGet + ": " + value);
        } else {
            System.out.println("Key " + keyToGet + " not found in the map.");
        }

        Map<Integer, String> map2 = new HashMap<>();
        map2.put(4, "Date");
        map2.put(5, "Elderberry");
        System.out.println("Map 2: " + map2);

        map1.putAll(map2);
        System.out.println("Map 1 after inserting elements from Map 2: " + map1);

        System.out.println("All keys and values in Map 1:");
        for (Map.Entry<Integer, String> entry : map1.entrySet()) {
            System.out.println("Key: " + entry.getKey() + ", Value: " + entry.getValue());
        }
    }
}
```

Output:

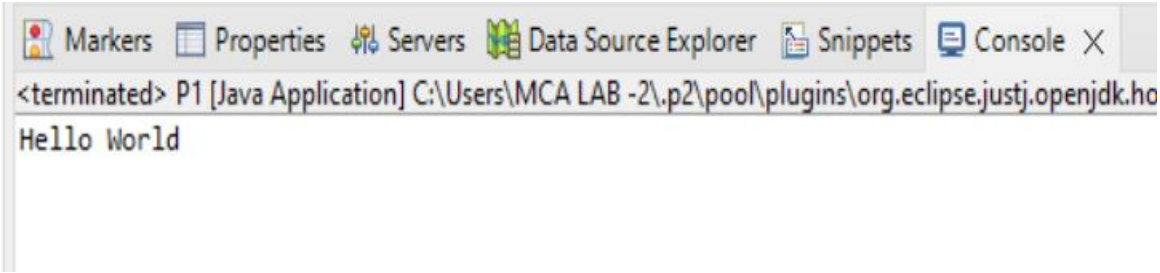
```
PS C:\Users\Administrator\Desktop\Prem Java> javac Ass4q1.java
PS C:\Users\Administrator\Desktop\Prem Java> java Ass4q1
Map 1 after adding items: {1=Apple, 2=Banana, 3=Cherry}
Map 1 after removing key 2: {1=Apple, 3=Cherry}
Map 1 contains the key: 3
Value associated with key 1: Apple
Map 2: {4=Date, 5=Elderberry}
Map 1 after inserting elements from Map 2: {1=Apple, 3=Cherry, 4=Date, 5=Elderberry}
All keys and values in Map 1:
Key: 1, Value: Apple
Key: 3, Value: Cherry
Key: 4, Value: Date
Key: 5, Value: Elderberry
```

5. Assignments on Lambda Expression

1. Write a Java program using Lambda Expression to print“HelloWorld”

```
public class P1 {
   interface Hello{
      void hello(String str);
   }
   public static void main(String[] args) {
      Hello refHello = (String str) -&gt; System.out.println(&quot;Hello &quot; + str);
      refHello.hello(&quot;World&quot;);
   }
}
```

Output:-

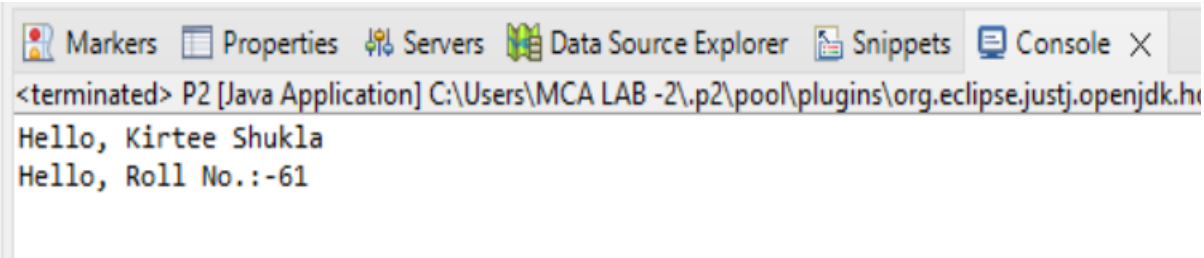


2. Write a Java program using Lambda Expression with single parameters.

```
interface Sayable{
   public String say(String name);
}
public class P2{
   public static void main(String[] args) {

      // Lambda expression with single parameter.
      Sayable s1=(name)-&gt;{
         return &quot;Hello, &quot;+name;
```

```
        };
        System.out.println(s1.say(&quot;Kirtee Shukla&quot;));

        // You can omit function parentheses
        Sayable s2= name -&gt;{
            return &quot;Hello, &quot;+name;
        };
        System.out.println(s2.say(&quot;Roll No.:-61&quot;));
    }
}
```

Output:



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

```
interface Addable{
    int add(int a,int b);
}

public class P3{
    public static void main(String[] args) {

        // Multiple parameters in lambda expression
        Addable ad1=(a,b)-&gt;(a+b);
        System.out.println(ad1.add(40,60));
```

```
        // Multiple parameters with data type in lambda expression
        Addable ad2=(int a,int b)-&gt;(a+b);
        System.out.println(ad2.add(800,200));
    }
}
```

Output:

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console
<terminated> P3 [Java Application] C:\Users\MCA LAB -2\.p2\pool\plugins\org.eclipse.justj.openjdk.hot
100
1000
```

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

```
import java.util.function.Function;
public class P4 {
  public static void main(String[] args) {
    Function&lt;Integer,Double&gt; centToFahrenheitInt = x -&gt; new
Double((x*9/5)+32);
    double fahrenheit = centToFahrenheitInt.apply(100);
    System.out.println(&quot;Centigrade to Fahrenheit: &quot;+fahrenheit);
  }
}
```

Output:

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console
<terminated> P4 [Java Application] C:\Users\MCA LAB -2\.p2\pool\plugins\org.eclipse.justj.openjdk.hotsp
Centigrade to Fahrenheit: 212.0
```

b. Convert Kilometers to Miles

```
import java.util.Scanner;
public class P4 {
```

```
    public static void main(String[] args) {
        // Declaring the variables
        double kiloMeters, miles;
        // 1 mile = 1.609344 kilometers =&gt; 1 kilometer = 1/1.609344 miles.
        double conversionFactor = 1.609344;
        // Getting user input using Scanner class

        System.out.println(&quot;Enter distance value in Kilometers : &quot;);
        Scanner input = new Scanner(System.in);
        kiloMeters = input.nextDouble();
        // To convert kilometers to miles, dividing the kilometers by 1.609344
        miles = kiloMeters / conversionFactor;
        //printing the output
        System.out.println(&quot;The distance in Miles : &quot; + miles);
    }
}
```

Output:

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console
<terminated> P4 [Java Application] C:\Users\MCA LAB -2\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.j
Enter distance value in Kilometers :
500
The distance in Miles : 310.685596118667
```

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

```
interface Addable{
    int add(int a,int b);
}

public class P5 {
    public static void main(String[] args) {
```

```
        // Lambda expression without return keyword.
        Addable ad1=(a,b)-&gt;(a+b);
        System.out.println(ad1.add(80,20));

        // Lambda expression with return keyword.
        Addable ad2=(int a,int b)-&gt;{
                        return (a+b);
                        };
        System.out.println(ad2.add(800,200));

    }
}
```

Output:

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console ×
<terminated> P5 [Java Application] C:\Users\MCA LAB -2\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspo
100
1000
```

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

```
import java.util.*;
import java.util.stream.*;
public class P6 {
    public static void main(String[] args) {
        List&lt;String&gt; list = new ArrayList&lt;&gt;();
        list.add(&quot;Kir&quot;);
        list.add(&quot;tee&quot;);
        list.add(&quot;shu&quot;);
        String result = list
                .stream()
                .map(s -&gt; s.substring(0, 2 ))
```

```
            .collect(Collectors.joining());
        System.out.println(result); //Prem
    }
}
```

Output:

```
Markers  Properties  Servers  Data Source Explorer  Snippets  Console
<terminated> P6 [Java Application] C:\Users\MCA LAB -2\.p2\pool\plugins\org.eclipse.justj.openjdk.hotsp
Prem
```

6. Assignments based on web application development using JSP

1. **Write Programs to demonstrate different Implicit Objects**
   a. OUT
   b. Request
   c. Session

```
<%@pagelanguage="java"contentType="text/html;charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<!DOCTYPEhtml>
<html>
<head>
<metacharset="ISO-8859-1">
<title>Inserttitlehere</title>
</head>
<body>
<h1>OutObject</h1>
<%out.println("Luffy:Thisis...aloveordeal");%>
<h1>ReuqestObject</h1>
<%
Stringuri=request.getRequestURI();
out.println("RequestedURI:"+uri);
%>
<h1>SessionObject</h1>
<%
session.setAttribute("luffy", "I refuse your refusal"); Stringattribute=(String)session.getAttribute("luffy");
out.println("Thevalueofthesessionattribute'attribute'is:"+ attribute);
%>
</body>
</html>
```

Output:


Out Object
Luffy : This is... a love ordeal
Reuqest Object
Requested URI: /MCA-78/CoreTag.jsp
Session Object
The value of the session attribute 'attribute' is: I refuse your refusal

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

```
<%@pageimport="java.util.ArrayList"%>
<jsp:useBeanid="myBean"class="jspExample.MyBean"scope="request"/>
<%
//Setdatainthebean
myBean.setData("Sorry,butitlookslikeI'mdead.");
// Retrieve data from the bean
Stringdata=myBean.getData();
%>
<html>
<head><title>TemporaryStorageUsingBean</title></head>
<body>
<h2>DatastoredinBean:</h2>
<p><%=data%></p>
</body>
</html>
```

Output:

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

```
<%@pagelanguage="java"contentType="text/html;charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<!DOCTYPEhtml>
<html>
<head>
<metacharset="ISO-8859-1">
<title>Practical7</title>
</head>
<body>
<body>
<%@includefile="header.jsp"%><!--Directivetoincludeheader-->
<%--JSPDeclaration--%> <%!intcount=0;%>
<%--JSPScriptlet--%> <%
count++;
out.println("ThisisaExampleofscriptlet.Countisnow:"+count);
%>
<%--JSPExpression--%>
<p>ThisisanExampleofDirectiveexpression.Thevalueofcountisnow: <%= count %></p>
<%@includefile="footer.jsp"%><!--Directivetoincludefooter-->
</body>
</body>
</html>
```

Output:

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

```
<%@pagelanguage="java"contentType="text/html;charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<%@includefile="header.jsp"%>

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

<!DOCTYPEhtml>

<html>

<head>

<metacharset="ISO-8859-1">

<title>JSPDirectives</title>

</head>

<body>

<h2>WelcometoJSPDirectives!</h2>

<c:outvalue="${'Istillhavemyfriends!'}"/>

<%@includefile="footer.jsp"%>

</body>

</html>
```

Output:

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

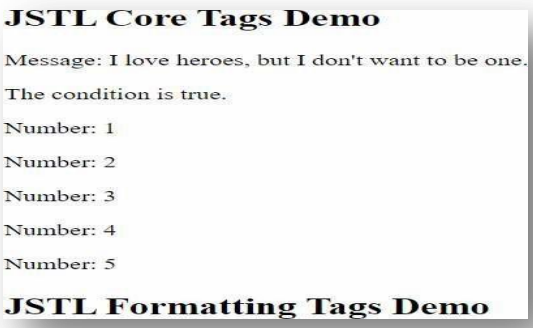
```
<%@pageimport="java.io.PrintWriter"%>
<%
// Get the current session or create a new one
HttpSessionsession1=request.getSession(true);
// Set session attribute
session1.setAttribute("username", "Session:luffy");
//Createacookiefortheusername
CookieusernameCookie=newCookie("username","Cookie:Luffy");
response.addCookie(usernameCookie);
%>
<html>
<head><title>SessionTrackingUsingCookies</title></head>
<body>
<h2>SessionTrackingUsingCookies</h2>
<p>Usernamestoredinsession:<%=session1.getAttribute("username")
%></p>
<p>Usernamestoredincookie:<%=usernameCookie.getValue()%></p>
</body>
</html>
```

Output:

6. Write a program to demonstrate JSTL Tags

```
<%@taglib uri="http://java.sun.com/jsp/jstl/core"prefix="c"%>
<%@tagliburi="http://java.sun.com/jsp/jstl/fmt"prefix="fmt"%>
<html>
<head>
<title>JSTLDemo</title>
</head>
<body>
<h2>JSTLCoreTagsDemo</h2>
<c:setvar="message"value="Iloveheroes,butIdon'twanttobeone."
/>
<p>Message:<c:outvalue="${message}"/></p>
<c:iftest="${5>3}">
<p>Theconditionistrue.</p>
</c:if>
<c:forEach var="i" begin="1" end="5">
<p>Number:${i}</p>
</c:forEach>
</body>
</html>
```

Output:

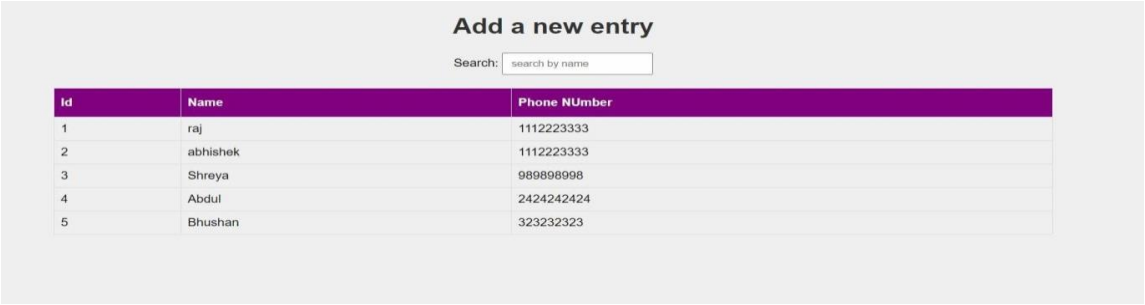


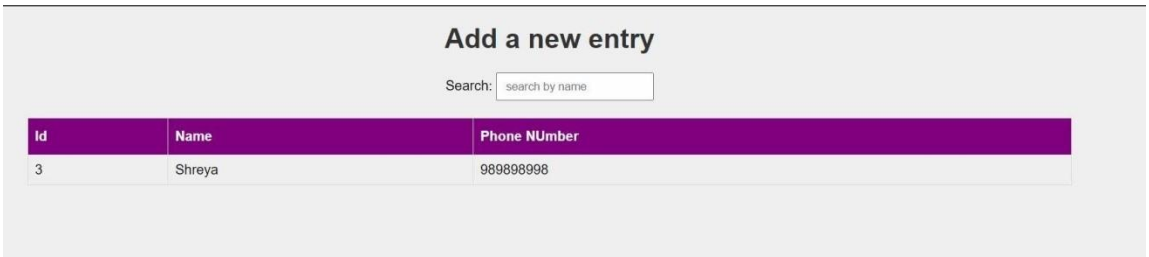
7. Create a Telephone directory using JSP and store all the information within a database, so that later could be retrieved as per the requirement. Make your own assumptions.

```
<%@pageimport="java.io.*,java.util.*,java.sql.*"%>

<%@pageimport="javax.servlet.http.*,javax.servlet.*"%>

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

<%@tagliburi="http://java.sun.com/jsp/jstl/sql"prefix="sql"%>

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

<!DOCTYPEhtml>

<html>

<head>

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

<title>Practical1</title>

<style> body{ font-family:Arial,sans-serif; background-color: #f0f0f0; margin: 0; padding:0;

} h1{ color:#333; text-align:center; margin-top: 20px;

}

form{ text-align:center; margin-top: 20px;

} table{ margin: 0 auto; margin-top:20px; border-collapse:collapse; width: 80%;

} table,th,td{ border:1pxsolid#ddd; padding: 8px; } padding-top: 12px; th{ padding-bottom: 12px;
text-align: left; background-color:purple; color: white;

}

input{ height : 20px; padding:5px10px;

}

</style>

</head>

<body>

<h1>Addanewentry</h1>

<formmethod="get">

<labelfor="search">Search:</label>

<inputtype="text"id="search"name="search"placeholder="searchby name">
```

```
</form>
<sql:setDataSourcevar="snapshot"driver="com.mysql.jdbc.Driver"
url="jdbc:mysql://localhost:3306/mcaraj" user="root"password="root"/>
<sql:querydataSource = "${snapshot}"var =
"result">SELECT*fromtelephonewherename LIke ?;
<sql:paramvalue="%${param.search}%"/>
</sql:query>
<tableborder="1"width="100%">
<tr>
<th>Id</th>
<th>Name</th>
<th>PhoneNUmber</th>
</tr>
<c:forEachvar="row"items="${result.rows}">
<tr>
<td><c:outvalue="${row.id}"/></td>
<td><c:outvalue="${row.name}"/></td>
<td><c:outvalue="${row.phoneNumber}"/></td>
</tr>
</c:forEach>
</table>
</body>
</body>
</html>
```

Output:



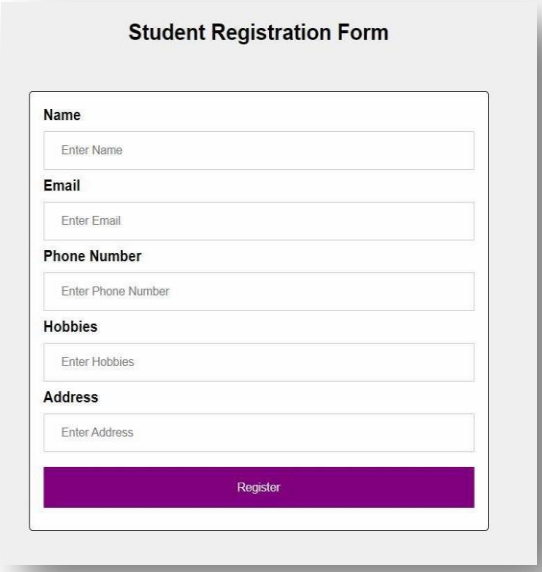

Add a new entry
Search:
search by name
Id
Name
Phone NUmber
3
Shreya
989898998

8. Write a JSP page to display the Registration form (Make your own assumptions)

```
<%@pagelanguage="java"contentType="text/html;charset=ISO-8859-1" pageEncoding="ISO-8859-
1"%>
<!DOCTYPEhtml>
<html>
<head>
<metacharset="ISO-8859-1">
<title>Practical2</title>
<style>
body{
font-family:Arial,sansserif; background-color:
#f0f0f0;
}
.container{
width: 500px; padding:16px;
background-color:white; margin: 0 auto;
margin-top:50px;border: 1pxsolidblack;border- radius:4px;
}
input[type=text], input[type=password] { width:
100%; padding: 12px 20px; margin:8px0;display: inline-block;
border:1pxsolid#ccc; box-sizing:borderbox;
}
button{ background-color:purple;color: white;
padding:14px20px; margin: 8px 0; border: none; cursor: pointer; width: 100%;
}
button:hover{ opacity:0.8;
}
h2{
text-align:center;
}
```

```
</style>
</head>
<body>
<h2>StudentRegistrationForm</h2>
<divclass="container">
<labelfor="name"><b>Name</b></label>
<inputtype="text"placeholder="EnterName"name="name"required>
<labelfor="email"><b>Email</b></label>
<inputtype="text"placeholder="EnterEmail"name="email"required>
<labelfor="phone"><b>PhoneNumber</b></label>
<inputtype="text"placeholder="EnterPhoneNumber"name="phone"required>
<labelfor="hobbies"><b>Hobbies</b></label>
</div>
</body>
</html”
```

Output:



7. Assignment based Spring Framework

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

Singer.java

```
packagecom.example.SpringTest;

publicclassSinger{

String name; int age; publicStringgetName(){ returnname; }

publicvoidsetName(Stringname){ this.name=name;

}

publicintgetAge(){ returnage; }

publicvoidsetAge(intage){ this.age=age;

} voiddisplayInfo()

{

System.out.println("Name:"+name+"Age:"+age); }

}
```

ApplicationContext.xml

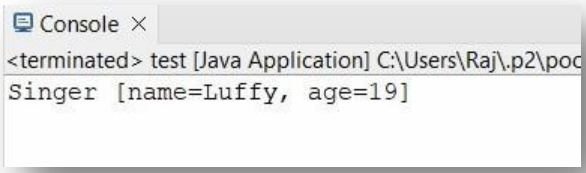
```
<?xmlversion="1.0"encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"xmlns:xsi="http://www.w3

.org/2001/XMLSchema-

instance"xmlns:context="http://www.springframework.org/schema/context"
xmlns:p="http://www.springframework.org/schema/p"xmlns:c="http://www.s
pringframework.org/schema/c"xsi:schemaLocation="http://www.springframe
work.org/schema/beanshttp://www.springframework.org/schema/beans/sprin g-

beans.xsdhttp://www.springframework.org/schema/contexthttp://www.sprin
gframework.org/schema/context/spring-context.xsd">

<beanid="Singer"class="com.example.Spring">

<propertyname="name"value="Luffy"></property>

<propertyname="age"value="19"></property> </bean> </beans>
```

SingerTest.java

```
packagecom.example.SpringTest; importorg.springframework.context.ApplicationContext;
```

```
importorg.springframework.context.support.ClassPathXmlApplicationContext;

publicclassSingerTest{ privatestaticApplicationContextctx;

public static void main(String[] args) {

ApplicationContext context = new

ClassPathXmlApplicationContext("Appctx.xml"); Singertemp=(Singer)ctx.getBean("Singer");
s1.displayInfo();

}

}
```

Output:



2. Write a program to demonstratedependency injection via setter method. (Primitive)

```
packageMCA;
publicclassZoro{ private String name; privatedoubleheight; private int swords;
        //      setterandgettermethods
publicStringgetName(){ returnname;
}
publicvoidsetName(Stringname){ this.name=name;
}
publicdoublegetHeight(){ returnheight;
}
publicvoidsetHeight(doubleheight){ this.height=height;
}
publicintgetSwords(){ returnswords;
}
publicvoidsetSwords(intswords){ this.swords=swords;
}
        //      Constructor
publicZoro(Stringname,doubleheight,intswords){ super(); this.name = name; this.height=height;
this.swords=swords;
}
publicZoro(){ super();
}
        //      tostring
method
@Override
publicStringtoString(){
return"nameofCharacter="+name+",heightofCharacter="
+height+",No.ofswords="+swords;
}
}
```

ApplicationContext.xml

```
<?xmlversion="1.0"encoding="UTF-8"?>

<beansxmlns="http://www.springframework.org/schema/beans"xmlns:xsi="http://www.w3.org/2001/XM
LSchemainstance"xmlns:context="http://www.springframework.org/schema/co
ntext"xmlns:p="http://www.springframework.org/schema/p"xmlns:c="
http://www.springframework.org/schema/c"xsi:schemaLocation="http
://www.springframework.org/schema/beanshttp://www.springframew
ork.org/schema/beans/springbeans.xsdhttp://www.springframework.org/schema/contexthttp://ww
w.springframework.org/schema/context/spring-context.xsd">

<beanclass="MCA.Zoro"name="zoro"p:name="PirateHunterRoronoaZoro"
p:height="6.2"p:swords="3"/> </beans>
```

MainClass

```
packageMCA;

importorg.springframework.context.ApplicationContext;

importorg.springframework.context.support.ClassPathXmlApplicationContext;

publicclasstest{

publicstaticvoidmain(String[]args){

ApplicationContextcontext=new

ClassPathXmlApplicationContext("MCA/mcaConfig.xml"

); Zoro temp = (Zoro) context.getBean("zoro"); System.out.println(temp);

}

}
```

Output:

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

PojoClass

```
packageMCA;

publicclassluffy{ private String name; private int gears; privatedoubleheight;
publicluffy(Stringname,intgears,doubleheight){ super(); this.name = name; this.gears=gears;

this.height=height;

}

@Override

publicStringtoString(){

return"Charactername="+name+",No.ofgears="+ gears + ", height

= "+ height + "]";

}

}
```

ApplicationContext.xml

```
<?xmlversion="1.0"encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"xmlns:xsi="
http://www.w3.org/2001/XMLSchemainstance"xmlns:context="http://www.springframework.org/schema
/co ntext"xmlns:p="http://www.springframework.org/schema/p"xmlns:c="
http://www.springframework.org/schema/c"xsi:schemaLocation="http
://www.springframework.org/schema/beanshttp://www.springframew
ork.org/schema/beans/springbeans.xsdhttp://www.springframework.org/schema/contexthttp://ww
w.springframework.org/schema/context/spring-context.xsd">

<beanclass="MCA.luffy"name="luffy"c:name="MonkeyD.Luffy"c:height="5.8"c:gears="5"/>

</beans>
```

MainClass

```
packageMCA;

importorg.springframework.context.ApplicationContext;

importorg.springframework.context.support.ClassPathXmlApplicationContext;

publicclasstest{ publicstaticvoidmain(String[]args){

ApplicationContextcontext=new

ClassPathXmlApplicationContext("MCA/mcaConfig.xml"); luffytemp=(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]
```

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

PojoClass

```
packageMCA;
public class sanji{
private String name; privatedoubleheight; private Zoro obj;
publicStringgetName(){ returnname;
}
publicvoidsetName(Stringname){ this.name=name;
}
publicdoublegetHeight(){ returnheight;
}
publicvoidsetHeight(doubleheight){ this.height=height;
}
publicZorogetObj(){ returnobj;
}
publicvoidsetObj(Zoroobj){ this.obj=obj;
}
publicsanji(Stringname,doubleheight,Zoroobj){ super(); this.name = name; this.height=height;
this.obj=obj;
}
publicsanji(){ super();
//TODOAuto-generatedconstructorstub
}
@Override
publicStringtoString(){
return"sanji[name="+name+",height="+height+",\nobj=" +obj+"]";
}
}
```

ReferenceClass

```
packageMCA;
```

```
publicclassZoro{ private String name; privatedoubleheight; private int swords;
        //      setterandgettermethods
publicStringgetName(){ returnname;
}
publicvoidsetName(Stringname){ this.name=name;
}
publicdoublegetHeight(){ returnheight;
}
publicvoidsetHeight(doubleheight){ this.height=height;
}
publicintgetSwords(){ returnswords;
}
publicvoidsetSwords(intswords){ this.swords=swords;
}
        //      Constructor
publicZoro(Stringname,doubleheight,intswords){ super(); this.name = name; this.height=height;
this.swords=swords;
}
publicZoro(){ super();
}
        //      tostring
method
@Override
publicStringtoString(){
return"nameofCharacter="+name+",heightofCharacter="
+height+",No.ofswords="+swords;
} }
```

ApplicationContext.xml

```
<?xmlversion="1.0"encoding="UTF-8"?>
```

```
<beans xmlns="http://www.springframework.org/schema/beans"xmlns:xsi="
http://www.w3.org/2001/XMLSchema-

instance"xmlns:context="http://www.springframework.org/schema/co
ntext"xmlns:p="http://www.springframework.org/schema/p"xmlns:c="
http://www.springframework.org/schema/c"xsi:schemaLocation="http
://www.springframework.org/schema/beanshttp://www.springframew
ork.org/schema/beans/springbeans.xsdhttp://www.springframework.org/schema/contexthttp://ww
w.springframework.org/schema/context/spring-context.xsd">

<beanclass="MCA.Zoro"name="zoro"p:name="PirateHunterRoronoaZoro"
p:height="6.2"p:swords="3"/>

<beanclass="MCA.sanji"name="sanji"p:name="VinsmokeSanji"p:height="6.0"p:obj-ref="zoro"/>

</beans>
```

MainClass

```
packageMCA;

importorg.springframework.context.ApplicationContext;

importorg.springframework.context.support.ClassPathXmlApplicationContext;

publicclasstest{ publicstaticvoidmain(String[]args){

ApplicationContextcontext=new

ClassPathXmlApplicationContext("MCA/mcaConfig.xml"); sanjitemp=(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]
```

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

```
packageMCA;
publicclassussop{
private String Name; privatedoubleheight; private luffy obj;
@Override
publicStringtoString(){
return"ussop[Name="+Name+",height="+height+",\nobj=" +obj+"]";
}
publicussop(Stringname,doubleheight,luffyobj){ super();Name= name; this.height=height; this.obj=obj;
}
}
```

ReferenceClass

```
packageMCA;
publicclassluffy{
private String name; private int gears; privatedoubleheight;
publicluffy(Stringname,intgears,doubleheight){
super(); this.name = name; this.gears=gears;
this.height=height;
}
@Override
publicStringtoString(){
return"Charactername="+name+",No.ofgears="+ gears + ", height
= "+ height + "]";
}
}
```

ApplicationContext.xml

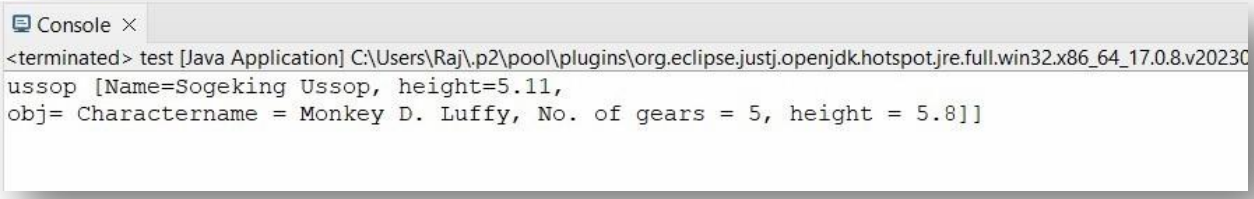
```
<?xmlversion="1.0"encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"xmlns:xsi="
http://www.w3.org/2001/XMLSchemainstance"xmlns:context="http://www.springframework.org/schema
```

```
/co ntext"xmlns:p="http://www.springframework.org/schema/p"xmlns:c="
http://www.springframework.org/schema/c"xsi:schemaLocation="http
://www.springframework.org/schema/beanshttp://www.springframew
ork.org/schema/beans/springbeans.xsdhttp://www.springframework.org/schema/contexthttp://ww
w.springframework.org/schema/context/spring-context.xsd">

<beanclass="MCA.luffy"name="luffy"c:name="MonkeyD.Luffy"c:height="5.8"c:gears="5"/>

<beanclass="MCA.ussop"name="ussop"c:name="SogekingUssop"c:height="5.11"c:obj-ref="luffy"/>

</beans>
```

MainClass

```
packageMCA;

importorg.springframework.context.ApplicationContext;

importorg.springframework.context.support.ClassPathXmlApplicationContext;

publicclasstest{ publicstaticvoidmain(String[]args){

ApplicationContextcontext=new

ClassPathXmlApplicationContext("MCA/mcaConfig.xml");
ussoptemp=(ussop)context.getBean("ussop");

System.out.println(temp);

}

}
```

Output:



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

PojoClass

```
packageMCA;

importjava.util.*;

publicclassstrawHat{ privateStringname; private List<String>crewName; private Set<String>bounty;
private Map<String, String>ability;

publicstrawHat(Stringname,List<String>crewName,Set<String>bounty,

Map<String, String>ability) {

super(); this.name = name;

this.crewName=crewName; this.bounty = bounty; this.ability = ability;

}

@Override

publicStringtoString(){

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

}

}
```

ApplicationContext.xml

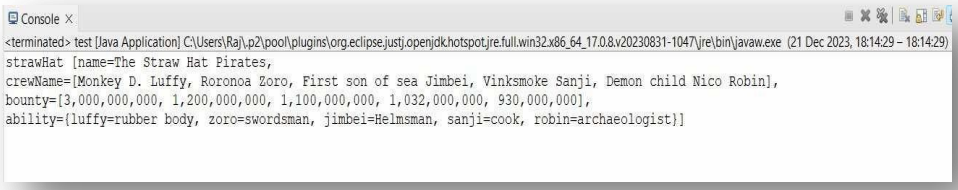
```
<?xmlversion="1.0"encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"xmlns:xsi="
ork.org/schema/beans/springbeans.xsdhttp://www.springframework.org/schema/contexthttp://ww
w.springframework.org/schema/context/spring-context.xsd">

<beanclass="MCA.strawHat"name="strawHat">

<constructor-argname="name"value="TheStrawHatPirates"/>

<constructor-argname="crewName">

<list>

<value>MonkeyD.Luffy</value>

<value>RoronoaZoro</value>

<value>FirstsonofseaJimbei</value> <value>VinksmokeSanji</value>

<value>DemonchildNicoRobin</value>

</list>
```

```
</constructor-arg>
<constructor-argname="bounty">
<set>
</constructor-arg>
<constructor-argname="ability">
<map>
<entrykey="luffy"value="rubberbody"/>
<entrykey="zoro"value="swordsman"/>
<entrykey="jimbei"value="Helmsman"/>
<entrykey="sanji"value="cook"/>
<entrykey="robin"value="archaeologist"/>
</map>
</constructor-arg>
</bean> </beans>
```

MainClass

```
packageMCA;
importorg.springframework.context.ApplicationContext;
importorg.springframework.context.support.ClassPathXmlApplicationContext;
publicclasstest{
publicstaticvoidmain(String[]args){
ApplicationContextcontext=new
ClassPathXmlApplicationContext("MCA/mcaConfig.xml");
strawHattemp=(strawHat)context.getBean("strawHat"); System.out.println(temp);
}
```

Output:



8. Assignment based Aspect Oriented Programming

1. Write a program to demonstrate Spring AOP –before advice.

Pom.xml

```
<projectxmlns="http://maven.apache.org/POM/4.0.0"xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"xsi:schemaLocation="http://maven.apache.org/POM/4.0.0http://maven.apache.org/xsd/maven4.0.0.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 packageaop; publicinterfaceGuitar{ publicvoidmakeSong();
}
TargetObject
packageaop; publicclassbrookimplementsGuitar{
publicvoidmakeSong(){
```

```
System.out.println("Song

Started");

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

}

}
```

AspectClass

```
packageaop;

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

org.aspectj.lang.annotation.AfterReturning;impo rt org.aspectj.lang.annotation.AfterThrowing; import
org.aspectj.lang.annotation.Around; import org.aspectj.lang.annotation.Aspect; import
org.aspectj.lang.annotation.Before;

@Aspect publicclassmcaAspect{

@Before("execution(*brook.makeSong())") publicvoidbeforeSong(){

System.out.println("YahooYahoo:IambeforeAspect");

}

}
```

ConfigurationClass

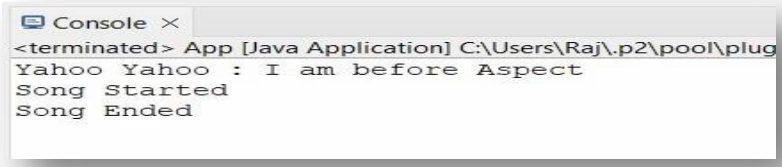
```
<?xmlversion="1.0"encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"xmlns:
xsi="http://www.w3.org/2001/XMLSchemainstance"xmlns:context="http://www.springframework.org/sc
hem a/context"xmlns:aop="http://www.springframework.org/schema/a op"
xsi:schemaLocation="http://www.springframework.org/schema/beanshttp://www.springframework.org/sc
hema/beans/springbeans.xsdhttp://www.springframework.org/schema/aophttp://www.s
pringframework.org/schema/aop/spring-aop.xsd">

<aop:aspectj-autoproxy/>

<beanname="brook"class="aop.brook"/>

<beanname="mcaaspect"class="aop.mcaAspect"/>

</beans>
```

MainClass

```
packageaop;

importorg.springframework.context.ApplicationContext;
```

```
importorg.springframework.context.support.ClassPathXmlApplicationContext; publicclassApp{

public static void main(String[] args) { ApplicationContext context= new

ClassPathXmlApplicationContext("aop/aopConfig.xml"); Guitar temp = (Guitar)
context.getBean("brook"); temp.makeSong();

}

}
```

Output:

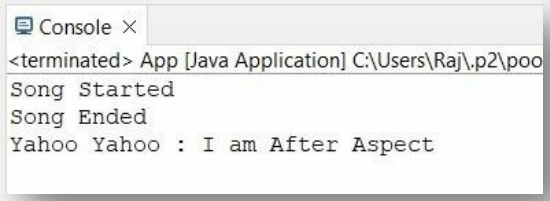


2. **Write a program to demonstrate Spring AOP–after advice.**

AspectClass

```
packageaop;

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

org.aspectj.lang.annotation.AfterReturning;impo rt org.aspectj.lang.annotation.AfterThrowing; import
org.aspectj.lang.annotation.Around; import org.aspectj.lang.annotation.Aspect; import
org.aspectj.lang.annotation.Before;

@Aspect publicclassmcaAspect{

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

publicvoidafterSong(){

System.out.println("YahooYahoo:IamAfterAspect");}
```

Output:



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

AspectClass

```
packageaop;

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

org.aspectj.lang.annotation.AfterReturning;impo rt org.aspectj.lang.annotation.AfterThrowing; import
org.aspectj.lang.annotation.Around; import org.aspectj.lang.annotation.Aspect; import
org.aspectj.lang.annotation.Before;

@Aspect publicclassmcaAspect{

@Around("execution(*brook.makeSong())") publicvoidaroundSong(){

System.out.println("YahooYahoo:AroundAspect");

}

}
```

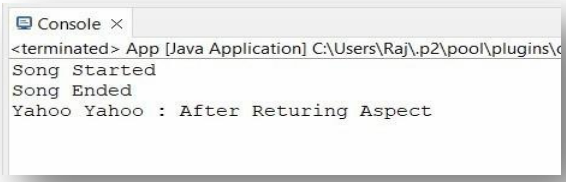
Output:

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

AspectClass

```
packageaop;
import org.aspectj.lang.annotation.After;
 import org.aspectj.lang.annotation.AfterReturning;
import org.aspectj.lang.annotation.AfterThrowing;
 import org.aspectj.lang.annotation.Around; import
org.aspectj.lang.annotation.Aspect;
 import org.aspectj.lang.annotation.Before;
 import org.aspectj.lang.annotation.Pointcut;
@Aspect publicclassmcaAspect{
@AfterReturning("execution(*brook.makeSong())") publicvoidAfterReturnSong(){
System.out.println("YahooYahoo:AfterReturingAspect");
}
}
```

Output:

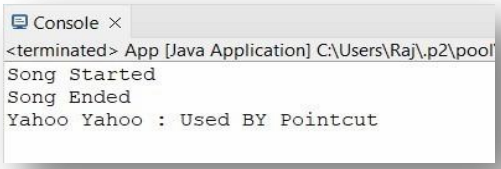


5. **Write a program to demonstrate SpringAOP – point cuts.**

AspectClass

```
packageaop;
import org.aspectj.lang.annotation.After;
import org.aspectj.lang.annotation.Around; import
org.aspectj.lang.annotation.Aspect;
import org.aspectj.lang.annotation.Before;
import org.aspectj.lang.annotation.Pointcut;
@Aspect publicclassmcaAspect{
@Pointcut("execution(*brook.makeSong())") publicvoidsongPointCut(){
System.out.println("YahooYahoo:Iampointcut");
}
@AfterReturning("songPointCut()") publicvoidafterSong(){
System.out.println("YahooYahoo:UsedBYPointcut");
}
}
```

Output:



9. Assignment based Spring JDBC

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

Pom.xml

```
<projectxmlns="http://maven.apache.org/POM/4.0.0"xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0http://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

```
<?xmlversion="1.0"encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"xmlns:
xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"xmlns:p="http://www.springframework.org/schema/p"xmlns:c="http:
ramework.org/schema/c"xsi:schemaLocation="http://www.springfra
mework.org/schema/beanshttp://www.springframework.org/schema
/beans/springbeans.xsdhttp://www.springframework.org/schema/contexthttp://ww
w.springframework.org/schema/context/spring- context.xsd">
<beanclass="org.springframework.jdbc.datasource.DriverManagerDataSource"name="ds">
```

```
<propertyname="driverClassName"value="com.mysql.jdbc.Driver"/>

<propertyname="url"value="jdbc:mysql://localhost:3306/springjdbc"/>

<propertyname="username"value="root"/>

<propertyname="password"value="root"/>

</bean>

<beanclass="org.springframework.jdbc.core.JdbcTemplate"name="jdbcTemplate"p:dataSource-
ref="ds"/> </beans>
```

MainClass

```
packagecom.mca;

importorg.springframework.context.ApplicationContext;

importorg.springframework.context.support.ClassPathXmlApplicationContext;
importorg.springframework.jdbc.core.JdbcTemplate;

publicclassApp

{

publicstaticvoidmain(String[]args)

{

System.out.println("kaizokuoniorewanaru!");

ApplicationContext context= new ClassPathXmlApplicationContext("com/mca/config.xml");
JdbcTemplate temp =context.getBean("jdbcTemplate",JdbcTemplate.class);

        //      insertQuery

Stringquery1="insertintostrawHatvalues(?,?,?)";Stringquery2="updatestrawHatsetbounty=?whereid=?";
String query3 = "delete from strawHatwhereid=?";

        //      firequery

intresult1=temp.update(query1,2,"zoro","1.2Billion");System.out.println("Numberofrecords insetred "+
result1);

intresult2=temp.update(query2,"4billion",1);System.out.println("Numberofrecordsupdated

"+ result2);

intresult3=temp.update(query3,5);System.out.println("NumberofrecordsDeleted "+ 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

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

MainClass

```
packagecom.mca;

importjava.sql.Connection;import java.sql.PreparedStatement; import java.sql.SQLException;

importorg.springframework.context.ApplicationContext;

importorg.springframework.context.support.ClassPathXmlApplicationContext;
importorg.springframework.jdbc.core.JdbcTemplate;
importorg.springframework.jdbc.core.PreparedStatementCreator;

publicclassApp

{ publicstaticvoidmain(String[]args)

{

System.out.println("kaizokuoniorewanaru!");

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

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

Stringquery1="insertintostrawHat(id,name,bounty)values(?,?,?)";
intresult=temp.update(newPreparedStatementCreator(){

@Override publicPreparedStatementcreatePreparedStatement(Connectioncon)throws

SQLException{ PreparedStatementps=con.prepareStatement(query1); ps.setInt(1,

3);

ps.setString(2, "zoro"); ps.setString(3,"1.1Billion");

returnps;

} });
```

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:

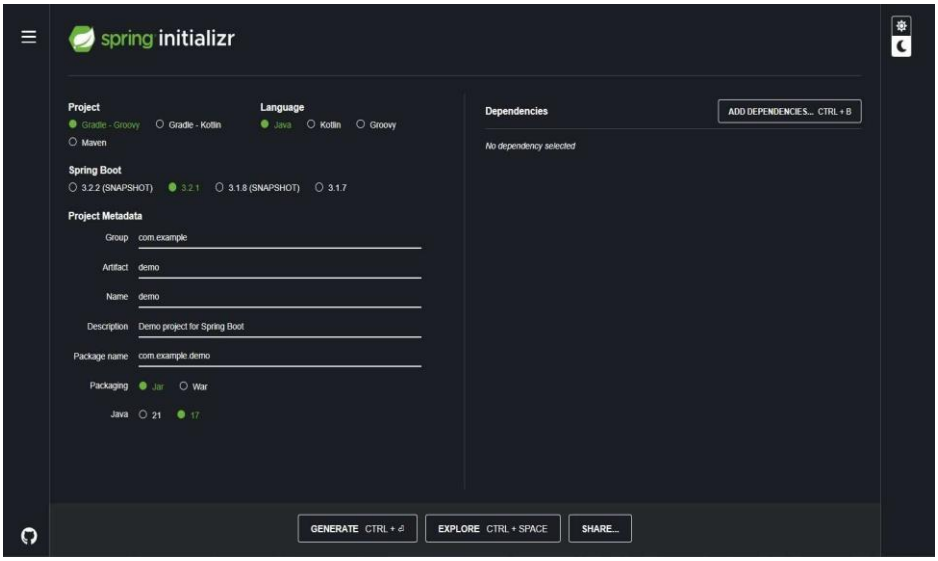
id	name	bounty
1	luffy	4 billion
2	zoro	1.2 Billion
3	zoro	1.1 Billion
NULL	NULL	NULL

10.Assignment based Spring Boot and RESTful Web Services

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

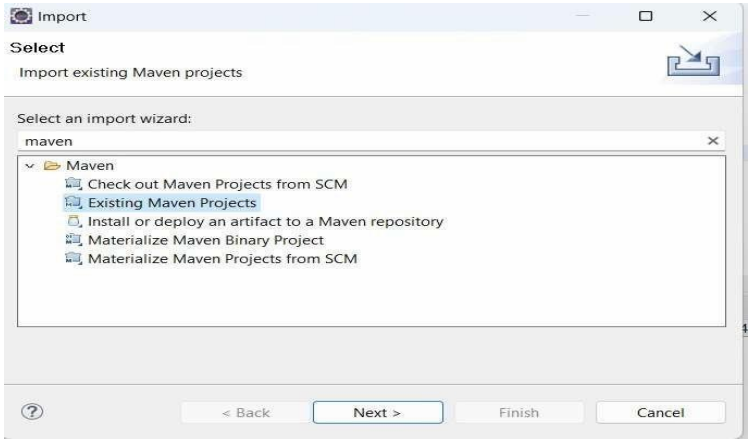
Step1:

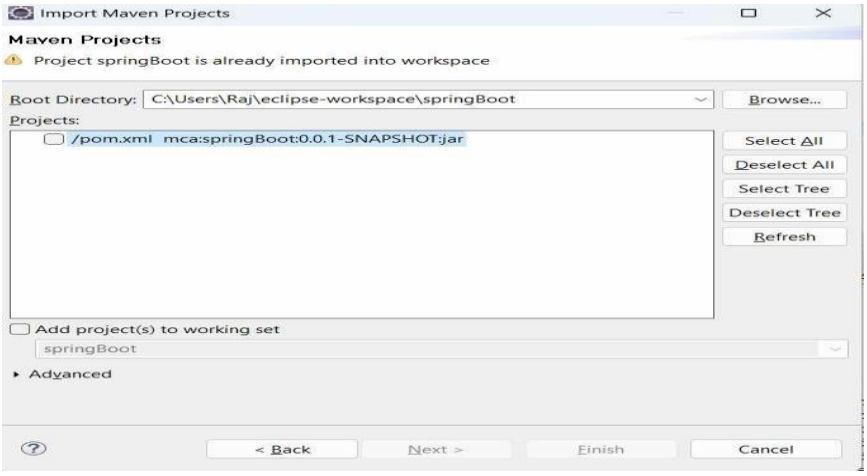
Go to Spring Initializr. Select the type of project (Maven). Choose the language (Java). Select the Spring Boot version. Fill in the project metadata. Add the necessary dependencies (at least spring- boot- starter-web). Click on "Generate" to download the project**.**



Step 2:

OpenEclipseIDE.NavigatetoFile>Import.Select"ExistingMavenProjects".Clickon"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".



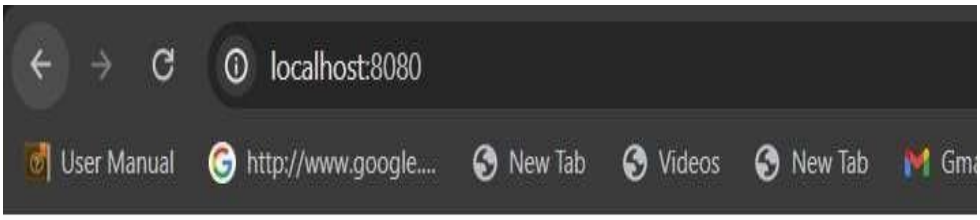


MainClass

```
packagecom.mca.spring;

importorg.springframework.boot.SpringApplication;
importorg.springframework.boot.autoconfigure.SpringBootApplication;
importorg.springframework.web.bind.annotation.GetMapping;
importorg.springframework.web.bind.annotation.RestController;

@SpringBootApplication publicclassmyApplication{

publicstaticvoidmain(String[]args){SpringApplication.run(myApplication.class,args);

}

@RestController

public class controller { @GetMapping("/") publicStringquote(){

return"Hero?No!We’repirates!Iloveheroes,butIdon’twanna

beone!";

}

}

}
```

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
```



Hero? No! We're pirates! I love heroes, but I don't wanna be one!