**Practical 1**

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

|  |
| --- |
| package Brijesh116;  public class Generic<T> {  T obj;  Generic(T obj) {  this.obj = obj;  }  T getObject() {  return obj;  }  }  package Brijesh116;  class G {  public static void main(String args[]) {  Integer i = 20;  Generic<Integer> obj = new Generic<Integer>(i);  System.out.println("Stored value: " + obj.getObject());    Double D = 20.25;  Generic<Double> obj2 = new Generic<Double>(D);  System.out.println("Stored value: " + obj2.getObject());    String S = "Brijesh Prasad";  Generic<String> obj3 = new Generic<String>(S);  System.out.println("Stored value: " + obj3.getObject());  }  } |

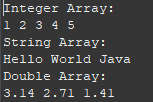
Output:



2. Write a Java Program to demonstrate Generic Methods.

|  |
| --- |
| package Brijesh116;  public class GenericMethodDemo { // Generic method that takes an argument of any type and prints it  public static <T> void printArray(T[] array) {  for (T element : array) {  System.out.print(element + " ");  }  System.out.println(); // New line after printing the array  }  // Main method to test the generic method  public static void main(String[] args) {  // Integer array  Integer[] intArray = {1, 2, 3, 4, 5};  System.out.println("Integer Array:");  printArray(intArray); // Calling the generic method with Integer type  // String array  String[] strArray = {"Hello", "World", "Java"};  System.out.println("String Array:");  printArray(strArray); // Calling the generic method with String type  // Double array  Double[] doubleArray = {3.14, 2.71, 1.41};  System.out.println("Double Array:");  printArray(doubleArray); // Calling the generic method with Double type  }  } |

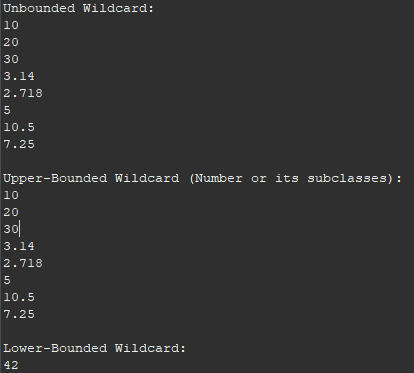
Output:



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

|  |
| --- |
| package Brijesh116;  import java.util.ArrayList;  import java.util.List;  class WildcardExample {  // Method using an unbounded wildcard  public static void printList(List<?> list) {  for (Object item : list) {  System.out.println(item);  }  }  // Method using an upper-bounded wildcard (can accept List of any subclass of Number)  public static void printNumberList(List<? extends Number> list) {  for (Number num : list) {  System.out.println(num);  }  }  // Method using a lower-bounded wildcard (can accept List of any superclass of Integer)  public static void addIntegers(List<? super Integer> list) {  list.add(42); // Valid, as Integer is a subclass of Number  // list.add(3.14); // Invalid, as 3.14 is not a subclass of Integer  }    public static void main(String[] args) {    // Creating lists of different types  List<Integer> intList = new ArrayList<>();  intList.add(10);  intList.add(20);  intList.add(30);  List<Double> doubleList = new ArrayList<>();  doubleList.add(3.14);  doubleList.add(2.718);  List<Number> numberList = new ArrayList<>();  numberList.add(5);  numberList.add(10.5);  numberList.add(7.25f);  // Demonstrating the use of unbounded wildcard  System.out.println("Unbounded Wildcard:");  printList(intList);  printList(doubleList);  printList(numberList);    System.out.println("\nUpper-Bounded Wildcard (Number or its subclasses):");  printNumberList(intList);  printNumberList(doubleList);  printNumberList(numberList);    // Demonstrating the use of lower-bounded wildcard  System.out.println("\nLower-Bounded Wildcard:");  List<Object> objectList = new ArrayList<>();  addIntegers(objectList); // Valid, as Object is a superclass of Integer  printList(objectList); // Output the content of objectList  }  } |

Output:



**Practical 2**

4. 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.

|  |
| --- |
| package Brijesh116;  import java.util.ArrayList;  import java.util.List;  public class ForEachList {  public static void main(String[] args) {  List<String> numbers = new ArrayList<>();  numbers.add("Hello");  numbers.add("This");  numbers.add("is");  numbers.add("Brijesh");  numbers.add("Prasad");  for (String number : numbers) {  System.out.println(number);  }  }  } |

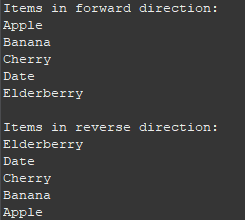
Output:



5. 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.

|  |
| --- |
| package Brijesh116;  import java.util.List;  import java.util.ListIterator;  import java.util.ArrayList;  public class ListIteratorExample {  public static void main(String[] args) {  // Create a list of items  List<String> itemList = new ArrayList<>();  itemList.add("Apple");  itemList.add("Banana");  itemList.add("Cherry");  itemList.add("Date");  itemList.add("Elderberry");    // Get a ListIterator for the list  ListIterator<String> listIterator = itemList.listIterator();    // Print items in forward direction using ListIterator  System.out.println("Items in forward direction:");  while (listIterator.hasNext()) {  System.out.println(listIterator.next());  }    // Print items in reverse direction using ListIterator  System.out.println("\nItems in reverse direction:");  while (listIterator.hasPrevious()) {  System.out.println(listIterator.previous());  }  }  } |

Output:

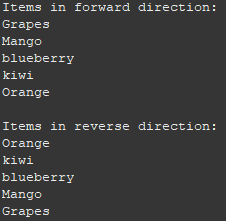


**Practical 3**

6. 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.

|  |
| --- |
| package Brijesh116;  import java.util.\*;  public class SetIteratorExample {  public static void main(String[] args) {  // Create a Set of Strings  Set<String> itemSet = new LinkedHashSet<>(); // LinkedHashSet to maintain insertion order  itemSet.add("Grapes");  itemSet.add("Mango");  itemSet.add("blueberry");  itemSet.add("kiwi");  itemSet.add("Orange");  // Print items in forward direction using Iterator  System.out.println("Items in forward direction:");  Iterator<String> iterator = itemSet.iterator();  while (iterator.hasNext()) {  System.out.println(iterator.next());  }  // To print the items in reverse, we need to convert the Set to a List (because Set does not support reverse iteration)  List<String> itemList = new ArrayList<>(itemSet);  // Print items in reverse direction using Iterator  System.out.println("\nItems in reverse direction:");  ListIterator<String> listIterator = itemList.listIterator(itemList.size());  while (listIterator.hasPrevious()) {  System.out.println(listIterator.previous());  }  }  } |

Output:



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

a. Add items in the set.

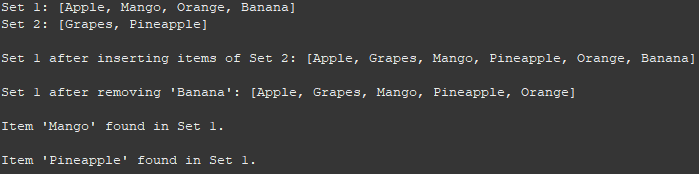
b. Insert items of one set in to other set.

c. Remove items from the set

d. Search the specified item in the set

|  |
| --- |
| package Brijesh116;  import java.util.HashSet;  import java.util.Set;  import java.util.Iterator;  public class SetOperations {  public static void main(String[] args) {  // Creating a Set to hold items (using HashSet)  Set<String> set1 = new HashSet<>();  Set<String> set2 = new HashSet<>();  // a. Add items in the set  set1.add("Apple");  set1.add("Banana");  set1.add("Orange");  set1.add("Mango");    set2.add("Grapes");  set2.add("Pineapple");  // Display the sets  System.out.println("Set 1: " + set1);  System.out.println("Set 2: " + set2);  // b. Insert items of one set into another set  set1.addAll(set2);  System.out.println("\nSet 1 after inserting items of Set 2: " + set1);  // c. Remove items from the set  set1.remove("Banana");  System.out.println("\nSet 1 after removing 'Banana': " + set1);  // d. Search for a specified item in the set  String searchItem = "Mango";  if (set1.contains(searchItem)) {  System.out.println("\nItem '" + searchItem + "' found in Set 1.");  } else {  System.out.println("\nItem '" + searchItem + "' not found in Set 1.");  }  // Trying to search for an item not in the set  searchItem = "Pineapple";  if (set1.contains(searchItem)) {  System.out.println("\nItem '" + searchItem + "' found in Set 1.");  } else {  System.out.println("\nItem '" + searchItem + "' not found in Set 1.");  }  }  } |

Output:



**Practical 4**

8. 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

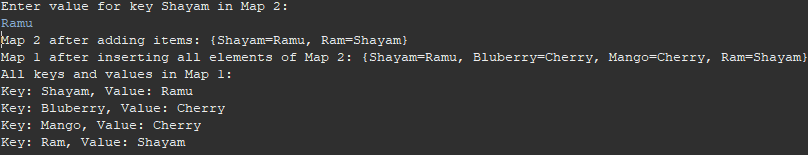
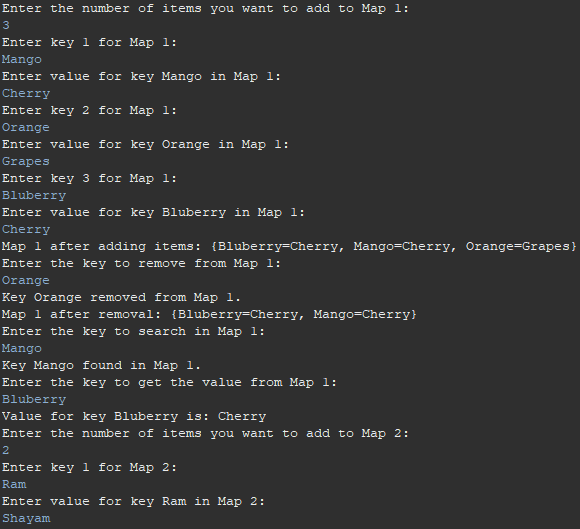
d. Get value of the specified key

e. Insert map elements of one map in to other map.

f. Print all keys and values of the map.

|  |
| --- |
| package Brijesh116;  import java.util.HashMap;  import java.util.\*;  public class MapOperationsDemo {  public static void main(String[] args) {  // Scanner object to read user input  Scanner scanner = new Scanner(System.in);  // Create two maps for demonstration  Map<String, String> map1 = new HashMap<>();  Map<String, String> map2 = new HashMap<>();  // a. Add items in the map  System.out.println("Enter the number of items you want to add to Map 1:");  int n = scanner.nextInt();  scanner.nextLine(); // Consume the newline character  for (int i = 0; i < n; i++) {  System.out.println("Enter key " + (i + 1) + " for Map 1:");  String key = scanner.nextLine();  System.out.println("Enter value for key " + key + " in Map 1:");  String value = scanner.nextLine();  map1.put(key, value);  }  System.out.println("Map 1 after adding items: " + map1);  // b. Remove items from the map  System.out.println("Enter the key to remove from Map 1:");  String keyToRemove = scanner.nextLine();  if (map1.remove(keyToRemove) != null) {  System.out.println("Key " + keyToRemove + " removed from Map 1.");  } else {  System.out.println("Key " + keyToRemove + " not found in Map 1.");  }  System.out.println("Map 1 after removal: " + map1);  // c. Search for a specific key in the map  System.out.println("Enter the key to search in Map 1:");  String keyToSearch = scanner.nextLine();  if (map1.containsKey(keyToSearch)) {  System.out.println("Key " + keyToSearch + " found in Map 1.");  } else {  System.out.println("Key " + keyToSearch + " not found in Map 1.");  }  // d. Get value of the specified key  System.out.println("Enter the key to get the value from Map 1:");  String keyToGetValue = scanner.nextLine();  String value = map1.get(keyToGetValue);  if (value != null) {  System.out.println("Value for key " + keyToGetValue + " is: " + value);  } else {  System.out.println("Key " + keyToGetValue + " not found in Map 1.");  }  // e. Insert map elements of one map into another map  System.out.println("Enter the number of items you want to add to Map 2:");  int m = scanner.nextInt();  scanner.nextLine(); // Consume the newline character  for (int i = 0; i < m; i++) {  System.out.println("Enter key " + (i + 1) + " for Map 2:");  String key = scanner.nextLine();  System.out.println("Enter value for key " + key + " in Map 2:");  String value1 = scanner.nextLine();  map2.put(key, value1);  }  System.out.println("Map 2 after adding items: " + map2);  // Insert all items of map2 into map1  map1.putAll(map2);  System.out.println("Map 1 after inserting all elements of Map 2: " + map1);  // f. Print all keys and values of the map  System.out.println("All keys and values in Map 1:");  for (Map.Entry<String, String> entry : map1.entrySet()) {  System.out.println("Key: " + entry.getKey() + ", Value: " + entry.getValue());  }  scanner.close();  }  } |

Output:



**Practical 5**

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

|  |
| --- |
| package Brijesh116;  public class HelloWorldLambda {  public static void main(String[] args) {  // Using Lambda Expression to implement the functional interface  Runnable helloWorld = () -> System.out.println("Hello World");  // Executing the lambda expression  helloWorld.run();  }  } |

Output:



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

|  |
| --- |
| package Brijesh116;  import java.util.function.Function;  public class LambdaSingleParameter {  public static void main(String[] args) {  Function<Integer, Integer> squareFunction = (x) -> x \* x;  int result = squareFunction.apply(5);  System.out.println("The square of 5 is: " + result);  }  } |

Output:



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

|  |
| --- |
| package Brijesh116;  import java.util.function.BiFunction;  public class LambdaMultipleParameters {  public static void main(String[] args) {  BiFunction<Integer, Integer, Integer> addNumbers = (a, b) -> a + b;  int result = addNumbers.apply(10, 20);  System.out.println("The sum of 10 and 20 is: " + result);  }  } |

Output:



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

a. Convert Fahrenheit to Celcius

b. Convert Kilometers to Miles.

|  |
| --- |
| package Brijesh116;  import java.util.function.Function;  public class FahrenheitToCelsius {  public static void main(String[] args) {  Function<Double, Double> fahrenheitToCelsius = (fahrenheit) -> (fahrenheit - 32) \* 5 / 9;  double fahrenheit = 110;  double celsius = fahrenheitToCelsius.apply(fahrenheit);  System.out.println(fahrenheit + "°F is " + celsius + "°C");  }  } |

Output:



|  |
| --- |
| package Brijesh116;  import java.util.function.Function;  public class KilometersToMiles {  public static void main(String[] args) {  Function<Double, Double> kilometersToMiles = (kilometers) -> kilometers \* 0.621371;  double kilometers = 10;  double miles = kilometersToMiles.apply(kilometers);  System.out.println(kilometers + " kilometers is " + miles + " miles");  }  } |

Output:



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

|  |
| --- |
| package Brijesh116;  public class LambdaWithReturn {  public static void main(String[] args) {  java.util.function.BiFunction<Integer, Integer, Integer> add = (a, b) -> {  return a + b;  };  int result = add.apply(5, 3);  System.out.println("The sum is: " + result);  }  } |

Output:



|  |
| --- |
| package Brijesh116;  public class LambdaWithoutReturn {  public static void main(String[] args) {  java.util.function.BiFunction<Integer, Integer, Integer> multiply = (a, b) -> a \* b;  int result = multiply.apply(5, 3);  System.out.println("The product is: " + result);  }  } |

Output:



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

|  |
| --- |
| package Brijesh116;  public class StringConcatenation {  public static void main(String[] args) {  java.util.function.BiFunction<String, String, String> concatenate = (str1, str2) -> str1 + str2;  String result = concatenate.apply("Hello, ", "World!");  System.out.println("Concatenated String: " + result);  }  } |

Output:



**Practical 6**

15. Write Programs  to demonstrate different Implicit Objects

Out, Request, Session

|  |
| --- |
| <%@pagelanguage="java"contentType="text/html;charset=ISO-8859-1"  pageEncoding="ISO-8859-1"%>  <!DOCTYPEhtml>  <html>  <head>  <metacharset="ISO-8859-1">  <title>Implicit</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:

A close-up of a text

Description automatically generated

16. 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:

A close up of a sign

Description automatically generated

17. 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--%>  <%!**int**count=0;%>  <%--JSPScriptlet--%>  <%  count++;  out.println("ThisisaExampleofscriptlet.Countisnow:"+count);  %>  <%--JSPExpression--%>  <p>ThisisanExampleofDirectiveexpression.Thevalueofcountisnow: <%= count %></p>  <%@includefile=*"footer.jsp"*%><!--Directivetoincludefooter-->  </body>  </body>  </html> |

Output:

A close-up of a purple and white rectangle

Description automatically generated

18. 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*](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:

A screenshot of a computer

Description automatically generated

19. 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=**new**Cookie("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:

A white background with black text

Description automatically generated

20. Write a program to demonstrate JSTL Tags

|  |
| --- |
| <%@taglib uri=*"*[*http://java.sun.com/jsp/jstl/core*](http://java.sun.com/jsp/jstl/core)*"*prefix=*"c"*%>  <%@tagliburi=*"*[*http://java.sun.com/jsp/jstl/fmt*](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:forEachvar=*"i"*begin=*"1"*end=*"5"*>  <p>Number:${i}</p>  </c:forEach>  </body>  </html> |

Output:

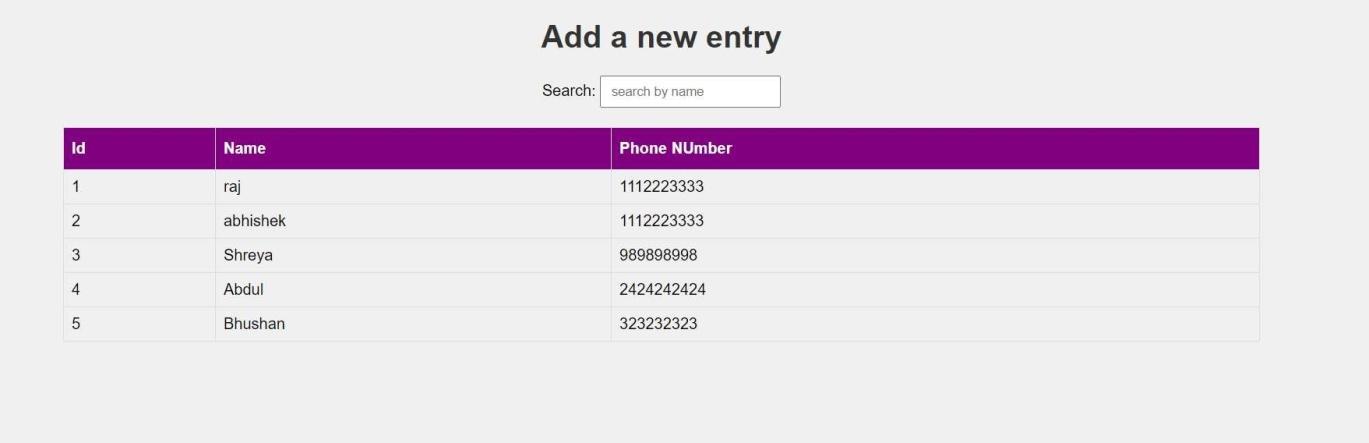
A white background with black text

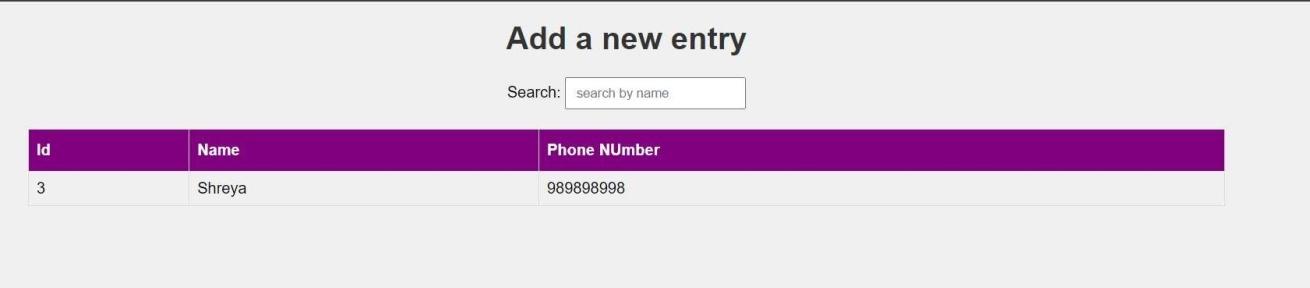
Description automatically generated

21. 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;  }  th{  }  padding-top: 12px; 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>  <%@ 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>  <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: 1px solid #ddd;  padding: 8px;  }  th {  padding-top: 12px;  padding-bottom: 12px;  text-align: left;  background-color: purple;  color: white;  }  input {  height: 20px;  padding: 5px 10px;  }  </style>  </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>  <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>  </html> |

Output:





22. 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,sans-serif*; 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:*border-box*;  }  **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>  <inputtype=*"text"*placeholder=*"EnterHobbies"*name=*"hobbies"*required>  <labelfor=*"address"*><b>Address</b></label>  <inputtype=*"text"*placeholder=*"EnterAddress"*name=*"address"*required>  <buttontype=*"submit"*>Register</button>  </div>  </body>  </html> |

Output:

A registration form with purple squares

Description automatically generated

**Practical 7**

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

|  |
| --- |
| Singer.java  **package**com.example.SpringTest;  **publicclass**Singer{ String name; **int** age;  **public**StringgetName(){  **return**name;  }  **publicvoid**setName(Stringname){  **this**.name=name;  }  **publicint**getAge(){  **return**age;  }  **publicvoid**setAge(**int**age){  **this**.age=age;  }  **void**displayInfo()  {  System.***out***.println("Name:"+name+"Age:"+age);  }  }  ApplicationContext.xml  <?xmlversion=*"1.0"*encoding=*"UTF-8"*?>  <beans xmlns=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)*"*xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"*xmlns:context=*"*[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)*"*xmlns:p=*"*[*http://www.springframework.org/schema/p*](http://www.springframework.org/schema/p)*"*xmlns:c=*"*[*http://www.springframework.org/schema/c*](http://www.springframework.org/schema/c)*"*xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)[*http://www.springframework.org/schema/context/spring-context.xsd*](http://www.springframework.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  **package**com.example.SpringTest;  **import**org.springframework.context.ApplicationContext;  **import**org.springframework.context.support.ClassPathXmlApplicationContext;  **publicclass**SingerTest{  **privatestatic**ApplicationContextctx;  **public static void** main(String[] args) { ApplicationContext context = **new** ClassPathXmlApplicationContext("Appctx.xml"); Singertemp=(Singer)ctx.getBean("Singer"); s1.displayInfo();  }  } |

**Output:**  
A screenshot of a computer

Description automatically generated

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

|  |
| --- |
| PojoClass  **package**MCA;  **publicclass**Zoro{  **private** String name; **privatedouble**height; **private int** swords;  //setterandgettermethods  **public**StringgetName(){  **return**name;  }  **publicvoid**setName(Stringname){  **this**.name=name;  }  **publicdouble**getHeight(){  **return**height;  }  **publicvoid**setHeight(**double**height){  **this**.height=height;  }  **publicint**getSwords(){  **return**swords;  }  **publicvoid**setSwords(**int**swords){  **this**.swords=swords;  }  **public**Zoro(Stringname,**double**height,**int**swords){  **super**(); **this**.name= name;  **this**.height=height;  **this**.swords=swords;  }  **public**Zoro(){  **super**();  }  **Public** StringtoString(){  **return**"nameofCharacter="+name+",heightofCharacter="  +height+",No.ofswords="+swords;  }  }  ApplicationContext.xml  <?xmlversion=*"1.0"*encoding=*"UTF-8"*?>  <beansxmlns=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)*"*xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"*xmlns:context=*"*[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)*"*xmlns:p=*"*[*http://www.springframework.org/schema/p*](http://www.springframework.org/schema/p)*"*xmlns:c=*"*[*http://www.springframework.org/schema/c*](http://www.springframework.org/schema/c)*"*xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)[*http://www.springframework.org/schema/context/spring-context.xsd*](http://www.springframework.org/schema/context/spring-context.xsd)*"*>  <beanclass=*"MCA.Zoro"*name=*"zoro"*p:name=*"PirateHunterRoronoaZoro"*  p:height=*"6.2"*p:swords=*"3"*/>  </beans>  MainClass  **package**MCA;  **import**org.springframework.context.ApplicationContext;  **import**org.springframework.context.support.ClassPathXmlApplicationContext;  **publicclass**test{  **publicstaticvoid**main(String[]args){  ApplicationContextcontext=**new**  ClassPathXmlApplicationContext("MCA/mcaConfig.xml"  ); Zoro temp = (Zoro) context.getBean("zoro"); System.out.println(temp);  }  } |

MavenDependencies  
A list of words on a white background

Description automatically generated

Output:



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

|  |
| --- |
| PojoClass  **package**MCA;  **publicclass**luffy{  **private** String name; **private int** gears; **privatedouble**height;  **public**luffy(Stringname,**int**gears,**double**height){  **super**(); **this**.name= name;  **this**.gears=gears;  **this**.height=height;  }  @Override  **public**StringtoString(){  **return**"Charactername="+name+",No.ofgears="+ gears + ", height = "+ height + "]";  }  }  ApplicationContext.xml  <?xmlversion=*"1.0"*encoding=*"UTF-8"*?>  <beans xmlns=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)*"*xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"*xmlns:context=*"*[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)*"*xmlns:p=*"*[*http://www.springframework.org/schema/p*](http://www.springframework.org/schema/p)*"*xmlns:c=*"*[*http://www.springframework.org/schema/c*](http://www.springframework.org/schema/c)*"*xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)[*http://www.springframework.org/schema/context/spring-context.xsd*](http://www.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  **package**MCA;  **import**org.springframework.context.ApplicationContext;  **import**org.springframework.context.support.ClassPathXmlApplicationContext;  **publicclass**test{  **publicstaticvoid**main(String[]args){  ApplicationContextcontext=**new**  ClassPathXmlApplicationContext("MCA/mcaConfig.xml");  luffytemp=(luffy)context.getBean("luffy"); System.out.println(temp);  }  } |

Output:

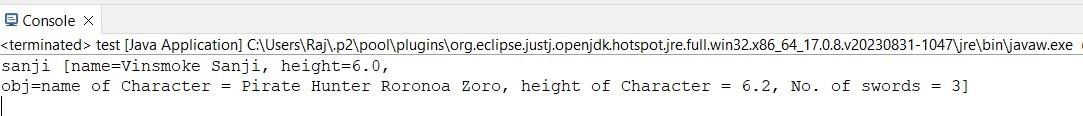
A screenshot of a computer

Description automatically generated

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

|  |
| --- |
| PojoClass  **package**MCA;  **publicclass**sanji{  **private** String name; **privatedouble**height; **private** Zoro obj;  **public**StringgetName(){  **return**name;  }  **publicvoid**setName(Stringname){  **this**.name=name;  }  **publicdouble**getHeight(){  **return**height;  }  **publicvoid**setHeight(**double**height){  **this**.height=height;  }  **public**ZorogetObj(){  **return**obj;  }  **publicvoid**setObj(Zoroobj){  **this**.obj=obj;  }  **public**sanji(Stringname,**double**height,Zoroobj){  **super**(); **this**.name= name;  **this**.height=height;  **this**.obj=obj;  }  **public**sanji(){  **super**();  //**TODO**Auto-generatedconstructorstub  }  @Override  **public**StringtoString(){  **return**"sanji[name="+name+",height="+height+",\nobj="  +obj+"]";  }  }  ReferenceClass  **package**MCA;  **publicclass**Zoro{  **private** String name; **privatedouble**height; **private int** swords;  //setter and getter methods  **public**StringgetName(){  **return**name;  }  **publicvoid**setName(Stringname){  **this**.name=name;  }  **publicdouble**getHeight(){  **return**height;  }  **publicvoid**setHeight(**double**height){  **this**.height=height;  }  **publicint**getSwords(){  **return**swords;  }  **publicvoid**setSwords(**int**swords){  **this**.swords=swords;  }  //Constructor  **public**Zoro(Stringname,**double**height,**int**swords){  **super**(); **this**.name= name;  **this**.height=height;  **this**.swords=swords;  }  **public** Zoro(){  **super**();  }  //to string method  @Override  **public**StringtoString(){  **return**"nameofCharacter="+name+",heightofCharacter="  +height+",No.ofswords="+swords;  }  }  ApplicationContext.xml  <?xmlversion=*"1.0"*encoding=*"UTF-8"*?>  <beans xmlns=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)*"*xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"*xmlns:context=*"*[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)*"*xmlns:p=*"*[*http://www.springframework.org/schema/p*](http://www.springframework.org/schema/p)*"*xmlns:c=*"*[*http://www.springframework.org/schema/c*](http://www.springframework.org/schema/c)*"*xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)[*http://www.springframework.org/schema/context/spring-context.xsd*](http://www.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  **package**MCA;  **import**org.springframework.context.ApplicationContext;  **import**org.springframework.context.support.ClassPathXmlApplicationContext;  **publicclass**test{  **publicstaticvoid**main(String[]args){  ApplicationContextcontext=**new**  ClassPathXmlApplicationContext("MCA/mcaConfig.xml");  sanjitemp=(sanji)context.getBean("sanji"); System.out.println(temp);  }  } |

Output:



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

|  |
| --- |
| PojoClass  **package**MCA;  **publicclass**ussop{  **private** String Name; **privatedouble**height; **private** luffy obj;  @Override  **public**StringtoString(){  **return**"ussop[Name="+Name+",height="+height+",\nobj="  +obj+"]";  }  **public**ussop(Stringname,**double**height,luffyobj){  **super**();Name= name;  **this**.height=height;  **this**.obj=obj;  }  }  ReferenceClass  **package**MCA;  **publicclass**luffy{  **private** String name; **private int** gears; **privatedouble**height;  **public**luffy(Stringname,**int**gears,**double**height){  **super**(); **this**.name= name;  **this**.gears=gears;  **this**.height=height;  }  @Override  **public**StringtoString(){  **return**"Charactername="+name+",No.ofgears="+ gears + ", height = "+ height + "]";  }  }  ApplicationContext.xml  <?xmlversion=*"1.0"*encoding=*"UTF-8"*?>  <beans xmlns=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)*"*xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"*xmlns:context=*"*[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)*"*xmlns:p=*"*[*http://www.springframework.org/schema/p*](http://www.springframework.org/schema/p)*"*xmlns:c=*"*[*http://www.springframework.org/schema/c*](http://www.springframework.org/schema/c)*"*xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)[*http://www.springframework.org/schema/context/spring-context.xsd*](http://www.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  **package**MCA;  **import**org.springframework.context.ApplicationContext;  **import**org.springframework.context.support.ClassPathXmlApplicationContext;  **publicclass**test{  **publicstaticvoid**main(String[]args){  ApplicationContextcontext=**new**  ClassPathXmlApplicationContext("MCA/mcaConfig.xml");  ussoptemp=(ussop)context.getBean("ussop"); System.out.println(temp);  }  } |

Output:  
A screenshot of a computer

Description automatically generated

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

|  |
| --- |
| PojoClass  **package**MCA;  **import**java.util.\*;  **publicclass**strawHat{  **private**Stringname;  **private** List<String>crewName; **private** Set<String>bounty; **private** Map<String, String>ability;  **public**strawHat(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  **public**StringtoString(){  **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*](http://www.springframework.org/schema/beans)*"*xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"*xmlns:context=*"*[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)*"*xmlns:p=*"*[*http://www.springframework.org/schema/p*](http://www.springframework.org/schema/p)*"*xmlns:c=*"*[*http://www.springframework.org/schema/c*](http://www.springframework.org/schema/c)*"*xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)[*http://www.springframework.org/schema/context/spring-context.xsd*](http://www.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>  <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-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  **package**MCA;  **import**org.springframework.context.ApplicationContext;  **import**org.springframework.context.support.ClassPathXmlApplicationContext;  **publicclass**test{  **publicstaticvoid**main(String[]args){  ApplicationContextcontext=**new**  ClassPathXmlApplicationContext("MCA/mcaConfig.xml");  strawHattemp=(strawHat)context.getBean("strawHat"); System.out.println(temp);  }  } |

Output:  
A close up of a text

Description automatically generated

29. **Write a program to demonstrate Auto wiring.**

|  |
| --- |
| PojoClass  **package**MCA;  **publicclass**chopper{  **private**ZoroZoro;  **public**ZorogetZoro(){  **return**Zoro;  }  **publicvoid**setZoro(Zorozoro){Zoro=zoro;  }  **public**chopper(MCA.Zorozoro){  **super**();Zoro= zoro;  }  **public**chopper(){  **super**();  }  @Override  **public**StringtoString(){  **return**"chopper[Zoro="+Zoro+"]";  }  }  ReferenceClass  **package**MCA;  **publicclass**Zoro{  **private** String name; **privatedouble**height; **private int** swords;  //setter and getter methods  **public**StringgetName(){  **return**name;  }  **publicvoid**setName(Stringname){  **this**.name=name;  }  **publicdouble**getHeight(){  **return**height;  }  **publicvoid**setHeight(**double**height){  **this**.height=height;  }  **publicint**getSwords(){  **return**swords;  }  **publicvoid**setSwords(**int**swords){  **this**.swords=swords;  }  //Constructor  **public**Zoro(Stringname,**double**height,**int**swords){  **super**(); **this**.name= name;  **this**.height=height;  **this**.swords=swords;  }  **public**Zoro(){  **super**();  }  //to string  method @Override  **public**StringtoString(){  **return**"nameofCharacter="+name+",heightofCharacter="  +height+",No.ofswords="+swords;  }  }  ApplicationContext.xml  <?xmlversion=*"1.0"*encoding=*"UTF-8"*?>  <beans xmlns=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)*"*xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"*xmlns:context=*"*[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)*"*xmlns:p=*"*[*http://www.springframework.org/schema/p*](http://www.springframework.org/schema/p)*"*xmlns:c=*"*[*http://www.springframework.org/schema/c*](http://www.springframework.org/schema/c)*"*xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)[*http://www.springframework.org/schema/context/spring-context.xsd*](http://www.springframework.org/schema/context/spring-context.xsd)*"*>  <beanclass=*"MCA.Zoro"*name=*"Zoro"*p:name=*"PirateHunterRoronoaZoro"*  p:height=*"6.2"*p:swords=*"3"*/>  <beanclass=*"MCA.chopper"*name=*"chopper"*autowire=*"byType"*/>  </beans>  MainClass  **package**MCA;  **import**org.springframework.context.ApplicationContext;  **import**org.springframework.context.support.ClassPathXmlApplicationContext;  **publicclass**test{  **publicstaticvoid**main(String[]args){  ApplicationContextcontext=**new**  ClassPathXmlApplicationContext("MCA/mcaConfig.xml");  choppertemp=(chopper)context.getBean("chopper"); System.out.println(temp);  }  } |

Output:  
A close up of a text

Description automatically generated

**Practical 8**

**30. 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/4.0.0<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](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;  **publicinterface**Guitar{  **publicvoid**makeSong();  }  TargetObject  **package**aop;  **publicclass**brook**implements**Guitar{  **publicvoid**makeSong(){  System.***out***.println("Song Started"); System.***out***.println("Song Ended");  }  }  AspectClass  **package**aop;  **import** org.aspectj.lang.annotation.After; **import** org.aspectj.lang.annotation.AfterReturning;**import** org.aspectj.lang.annotation.AfterThrowing; **import** org.aspectj.lang.annotation.Around; **import** org.aspectj.lang.annotation.Aspect; **import** org.aspectj.lang.annotation.Before;  @Aspect  **publicclass**mcaAspect{  @Before("execution(\*brook.makeSong())")  **publicvoid**beforeSong(){  System.***out***.println("YahooYahoo:IambeforeAspect");  }  }  ConfigurationClass  <?xmlversion=*"1.0"*encoding=*"UTF-8"*?>  <beans xmlns=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)*"*xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"*xmlns:context=*"*[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)*"*xmlns:aop=*"*[*http://www.springframework.org/schema/aop*](http://www.springframework.org/schema/aop)*"*  xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)[*http://www.springframework.org/schema/aop*](http://www.springframework.org/schema/aop)[*http://www.springframework.org/schema/aop/spring-aop.xsd*](http://www.springframework.org/schema/aop/spring-aop.xsd)*"*>  <aop:aspectj-autoproxy/>  <beanname=*"brook"*class=*"aop.brook"*/>  <beanname=*"mcaaspect"*class=*"aop.mcaAspect"*/>  </beans>  MainClass  **package**aop;  **import**org.springframework.context.ApplicationContext;  **import**org.springframework.context.support.ClassPathXmlApplicationContext;  **publicclass**App{  **public static void** main(String[] args) { ApplicationContext context= **new**  ClassPathXmlApplicationContext("aop/aopConfig.xml"); Guitar temp = (Guitar) context.getBean("brook"); temp.makeSong();  }  } |

Output:  
A screenshot of a computer

Description automatically generated

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

|  |
| --- |
| AspectClass  **package**aop;  **import** org.aspectj.lang.annotation.After; **import** org.aspectj.lang.annotation.AfterReturning;**import** org.aspectj.lang.annotation.AfterThrowing; **import** org.aspectj.lang.annotation.Around; **import** org.aspectj.lang.annotation.Aspect; **import** org.aspectj.lang.annotation.Before;  @Aspect  **publicclass**mcaAspect{  @After("execution(\*brook.makeSong())")  **publicvoid**afterSong(){  System.***out***.println("YahooYahoo:IamAfterAspect");  }  } |

Output:  
A screenshot of a computer

Description automatically generated

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

|  |
| --- |
| AspectClass  **package**aop;  **import** org.aspectj.lang.annotation.After; **import** org.aspectj.lang.annotation.AfterReturning;**import** org.aspectj.lang.annotation.AfterThrowing; **import** org.aspectj.lang.annotation.Around; **import** org.aspectj.lang.annotation.Aspect; **import** org.aspectj.lang.annotation.Before;  @Aspect  **publicclass**mcaAspect{  @Around("execution(\*brook.makeSong())")  **publicvoid**aroundSong(){  System.***out***.println("YahooYahoo:AroundAspect");  }  } |

Output:  
A screenshot of a computer

Description automatically generated

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

|  |
| --- |
| AspectClass  **package**aop;  **import** org.aspectj.lang.annotation.After; **import** org.aspectj.lang.annotation.AfterReturning;**import** org.aspectj.lang.annotation.AfterThrowing; **import** org.aspectj.lang.annotation.Around; **import**  org.aspectj.lang.annotation.Aspect; **import** org.aspectj.lang.annotation.Before; **import** org.aspectj.lang.annotation.Pointcut;  @Aspect  **publicclass**mcaAspect{  @AfterReturning("execution(\*brook.makeSong())")  **publicvoid**AfterReturnSong(){  System.***out***.println("YahooYahoo:AfterReturingAspect");  }  } |

Output:  
A screenshot of a computer

Description automatically generated

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

|  |
| --- |
| AspectClass  **package**aop;  **import** org.aspectj.lang.annotation.After; **import** org.aspectj.lang.annotation.AfterReturning;**import** org.aspectj.lang.annotation.AfterThrowing; **import** org.aspectj.lang.annotation.Around; **import**  org.aspectj.lang.annotation.Aspect; **import** org.aspectj.lang.annotation.Before; **import** org.aspectj.lang.annotation.Pointcut; @Aspect  **publicclass**mcaAspect{  @Pointcut("execution(\*brook.makeSong(..))")  **privatevoid**selectAll(){}  @AfterThrowing(pointcut="selectAll()",throwing="error")  **public void** afterThrowingAdvice(IllegalArgumentException error){ System.***out***.println("YahooYahoo:Therehasbeenanexception:");  }  }  TargetClass  **package**aop;  **publicclass**brook**implements**Guitar{  **publicvoid**makeSong(){  System.***out***.println("Song Started"); System.***out***.println("Song Ended");  **thrownew**IllegalArgumentException("Anerroroccurredwhile making the song.");  }  } |

Output:  
A screenshot of a computer

Description automatically generated

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

|  |
| --- |
| AspectClass  **package**aop;  **import** org.aspectj.lang.annotation.After; **import** org.aspectj.lang.annotation.AfterReturning;**import** org.aspectj.lang.annotation.AfterThrowing; **import** org.aspectj.lang.annotation.Around; **import**  org.aspectj.lang.annotation.Aspect; **import** org.aspectj.lang.annotation.Before; **import** org.aspectj.lang.annotation.Pointcut;  @Aspect  **publicclass**mcaAspect{  @Pointcut("execution(\*brook.makeSong())")  **publicvoid**songPointCut(){  System.***out***.println("YahooYahoo:Iampointcut");  }  @AfterReturning("songPointCut()")  **publicvoid**afterSong(){  System.***out***.println("YahooYahoo:UsedBYPointcut");  }  } |

Output:  
A screenshot of a computer

Description automatically generated

**Practical 9**

**36. 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/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](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*](http://www.springframework.org/schema/beans)*"*xmlns:xsi=*"*[*http://www.w3.org/2001/XMLSchema-instance*](http://www.w3.org/2001/XMLSchema-instance)*"*  xmlns:context=*"*[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)*"*xmlns:p=*"*[*http://www.springframework.org/schema/p*](http://www.springframework.org/schema/p)*"*xmlns:c=*"*[*http://www.springframework.org/schema/c*](http://www.springframework.org/schema/c)*"*xsi:schemaLocation=*"*[*http://www.springframework.org/schema/beans*](http://www.springframework.org/schema/beans)[*http://www.springframework.org/schema/beans/spring-beans.xsd*](http://www.springframework.org/schema/beans/spring-beans.xsd)[*http://www.springframework.org/schema/context*](http://www.springframework.org/schema/context)[*http://www.springframework.org/schema/context/spring-*](http://www.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  **package**com.mca;  **import**org.springframework.context.ApplicationContext;  **import**org.springframework.context.support.ClassPathXmlApplicationContext;  **import**org.springframework.jdbc.core.JdbcTemplate;  **publicclass**App  {  **publicstaticvoid**main(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  **int**result1=temp.update(query1,2,"zoro","1.2Billion");System.***out***.println("Numberofrecords insetred "+ result1);  **int**result2=temp.update(query2,"4billion",1);System.***out***.println("Numberofrecordsupdated "+ result2);  **int**result3=temp.update(query3,5);System.***out***.println("NumberofrecordsDeleted "+ result3);  }  } |

Output:  
A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

37. **Write a program to demonstrate Prepared Statement in Spring JDBCT emplate.**

|  |
| --- |
| MainClass  **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;  **publicclass**App  {  **publicstaticvoid**main(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(?,?,?)";  **int**result=temp.update(**new**PreparedStatementCreator(){  @Override  **public**PreparedStatementcreatePreparedStatement(Connectioncon)**throws**  SQLException{  }  });  PreparedStatementps=con.prepareStatement(query1); ps.setInt(1, 3);  ps.setString(2, "zoro"); ps.setString(3,"1.1Billion");  **return**ps; |

Output:  
A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**Practical 10**

**38. Write a program to create a simple Spring Boot application that print same message.**

**Step1:**

Go to Spring Initializr. Select the type of project (Maven). Choosethelanguage(Java).SelecttheSpringBootversion.

Fillin theproject metadata. Add thenecessary dependencies (atleast spring- boot- starter-web). Click on “Generate” to download the project.

A screenshot of a computer

Description automatically generated

**Step2:**

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”.

A screenshot of a software project

Description automatically generated  
  
A screenshot of a computer

Description automatically generated

|  |
| --- |
| MainClass  **package**com.mca.spring;  **import**org.springframework.boot.SpringApplication;  **import**org.springframework.boot.autoconfigure.SpringBootApplication;  **import**org.springframework.web.bind.annotation.GetMapping;  **import**org.springframework.web.bind.annotation.RestController;  @SpringBootApplication  **publicclass**myApplication{  **publicstaticvoid**main(String[]args){SpringApplication.run(myApplication.**class**,args);  }  beone!";  }  @RestController  **public class** controller { @GetMapping("/") **public**Stringquote(){  **return**"Hero?No!We’repirates!Iloveheroes,butIdon’twanna  }  } |

Output:  
A screenshot of a computer

Description automatically generated  
  
A screenshot of a phone

Description automatically generated