# Dayananda Sagar Academy of Technology & Management

Opp. Art of living, Udayapura, Kanakapura road, Bengaluru- 560082

(Affiliated to VTU, Belagavi & Approved by AICTE, New Delhi)

## Department of Information Science & Engineering

**Accredited by NBA, New Delhi**



# 2023-2024

# ADVANCED JAVA

# Laboratory Manual

# BIS402

Compiled by:

**Dr Veena R S (Associate Professor)**

**Prof. Ashwini R (Assistant Professor)**

**Prof. Namitha K V (Assistant Professor)**

# TABLE OF CONTENTS

| SL. No. | Lab Experiments | | |
|---|---|---|---|
| 1. | Implement a java program to demonstrate creating an ArrayList, adding elements, removing elements, sorting elements of ArrayList. Also illustrate the use of toArray() method. | | |
| 2. | Develop a program to read random numbers between a given range that are multiples of 2 and 5, sort the numbers according to tens place using comparator. | | |
| 3. | Implement a java program to illustrate storing user defined classes in collection. | | |
| 4. | Implement a java program to illustrate the use of different types of string class constructors. | | |
| 5. | Implement a java program to illustrate the use of different types of character extraction, string comparison, string search and string modification methods. | | |
| 6. | Implement a java program to illustrate the use of different types of String Buffer methods | | |
| 7. | Demonstrate a swing event handling application that creates 2 buttons Alpha and Beta and displays the text "Alpha pressed" when alpha button is clicked and "Beta pressed" when beta button is clicked. | | |
| 8. | .<br>A program to display greeting message on the browser "Hello UserName", "How Are You?", accept username from the client using servlet. | | |
| 9. | A servlet program to display the name, USN, and total marks by accepting student detail | | |
| 10. | A Java program to create and read the cookie for the given cookie name as "EMPID" and its value as "AN2356". | | |
| 11. | Write a JAVA Program to insert data into Student DATA BASE and retrieve info based on particular queries(For example update, delete, search etc...). | | |
| 12. | A program to design the Login page and validating the USER_ID and PASSWORD using JSP and DataBase.Additional Lab Experiments | | |

**DAYANANDA SAGAR ACADEMY OF TECHNOLOGY & MANAGEMENT**

**(Affiliated to Visvesvaraya Technological University,Belagavi & Approved by AICTE,New Delhi) Opp. Art of Living, Udayapura, Kanakapura Road, Bangalore – 560082 DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING**

**Accredited  by NBA, New Delhi**

## INSTITUTION VISION AND MISSION

### Vision of the Institution

To strive at creating the institution a center of highest caliber of learning, so as to create an overall intellectual atmosphere with each deriving strength from the other to be the best of engineers, scientists with management &design skills.

## Mission of the Institution:

- To serve its region, state, the nation and globally by preparing students to make
- meaningful contributions in an increasing complex global society challenges.
- To encourage, reflection on and evaluation of emerging needs and priorities with state of art infrastructure at institution.
- To support research and services establishing enhancements in technical, health, economic, human and cultural development.
- To establish inter disciplinary center of excellence, supporting/ promoting student's implementation.
- To increase the number of Doctorate holders to promote research culture on campus.
- To establish IIPC, IPR, EDC, innovation cells with functional MOU's supporting student's quality growth.

# DAYANANDA SAGAR ACADEMY OF TECHNOLOGY & MANAGEMENT

**(Affiliated to Visvesvaraya Technological University,Belagavi & Approved by AICTE,New Delhi) Opp. Art of Living, Udayapura, Kanakapura Road, Bangalore – 560082 DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING**

*Accredited by NBA, New Delhi*

## QUALITY POLICY

Dayananda Sagar Academy of Technology and Management aims at achieving academic excellence through continuous improvement in all spheres of Technical and Management education. In pursuit of excellence cutting-edge and contemporary skills are imparted to the utmost satisfaction of the students and the concerned stakeholders.

## OBJECTIVES & GOALS

- Creating an academic environment to nurture and develop competent entrepreneurs, leaders and professionals who are socially sensitive and environmentally conscious.

- Integration of Outcome Based Education and cognitive teaching and learning strategies to enhance learning effectiveness.

- Developing necessary infrastructure to cater to the changing needs of Business and Society.

- Optimum utilization of the infrastructure and resources to achieve excellence in all areas of relevance.

- Adopting learning beyond curriculum through outbound activities and creative assignments.

- Imparting contemporary and emerging techno-managerial skills to keep pace with the changing global trends.

- Facilitating greater Industry-Institute Interaction for skill development and employability enhancement.

- Establishing systems and processes to facilitate research, innovation and entrepreneurship for holistic development of students.

- Implementation of Quality Assurance System in all Institutional processes

## DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING
**Accredited by NBA, New Delhi**

## VISION OF THE DEPARTMENT

Impart magnificent learning atmosphere establishing innovative practices among the students aiming to strengthen their software application knowledge and technical skills.

## MISSION OF THE DEPARTMENT

**M1:** To deliver quality technical training on software application domain.

**M2:** To nurture teamwork in order to transform individual as responsible leader and entrepreneur for future trends.

**M3:** To inculcate research practices in teaching thus ensuring research blend among students.

**M4:** To ensure more doctorates in the department, aiming at professional strength.

**M5:** To inculcate the core information science engineering practices with hardware blend by providing advanced laboratories.

**M6:** To establish innovative labs, start-ups and patent culture.

## Program Educational Objectives (PEOs)

**PEO1:** Graduates shall have successful careers as information science engineers and will be able to lead and manage teams across the globe.

**PEO2:** Graduates shall be professional in engineering practice and shall demonstrate good problem solving, communication skills and contribute to address societal issues.

**PEO3:** Graduates shall be pursuing distinctive education, entrepreneurship and research in an excellent environment which helps in the process of life-long learning.

# DAYANANDA SAGAR ACADEMY OF TECHNOLOGY & MANAGEMENT

**(Affiliated to Visvesvaraya Technological University,Belagavi & Approved by AICTE,New Delhi) Opp. Art of Living, Udayapura, Kanakapura Road, Bangalore – 560082 DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING**
Accredited by NBA, New Delhi

# Program Outcomes (POs)

**PO1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

**PO2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

**PO3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

**PO4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

**PO5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

**PO6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

**PO7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

**PO8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

**PO9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

**PO10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions. **PO11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**PO12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage independent and life-long learning in the broadest context of technological change.

\

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY & ANAGEMENT
**(Affiliated to Visvesvaraya Technological University,Belagavi & Approved by AICTE,New Delhi)** Opp. Art of Living, Udayapura, Kanakapura Road, Bangalore- 560082

## DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING
**Accredited  by NBA, New Delhi**

SUBJECT: **ADVANCED JAVA** SUBJECT

CODE: **BIS402** SEMESTER: **IV**

# Course Outcomes

At the end of the course the student will be able to:

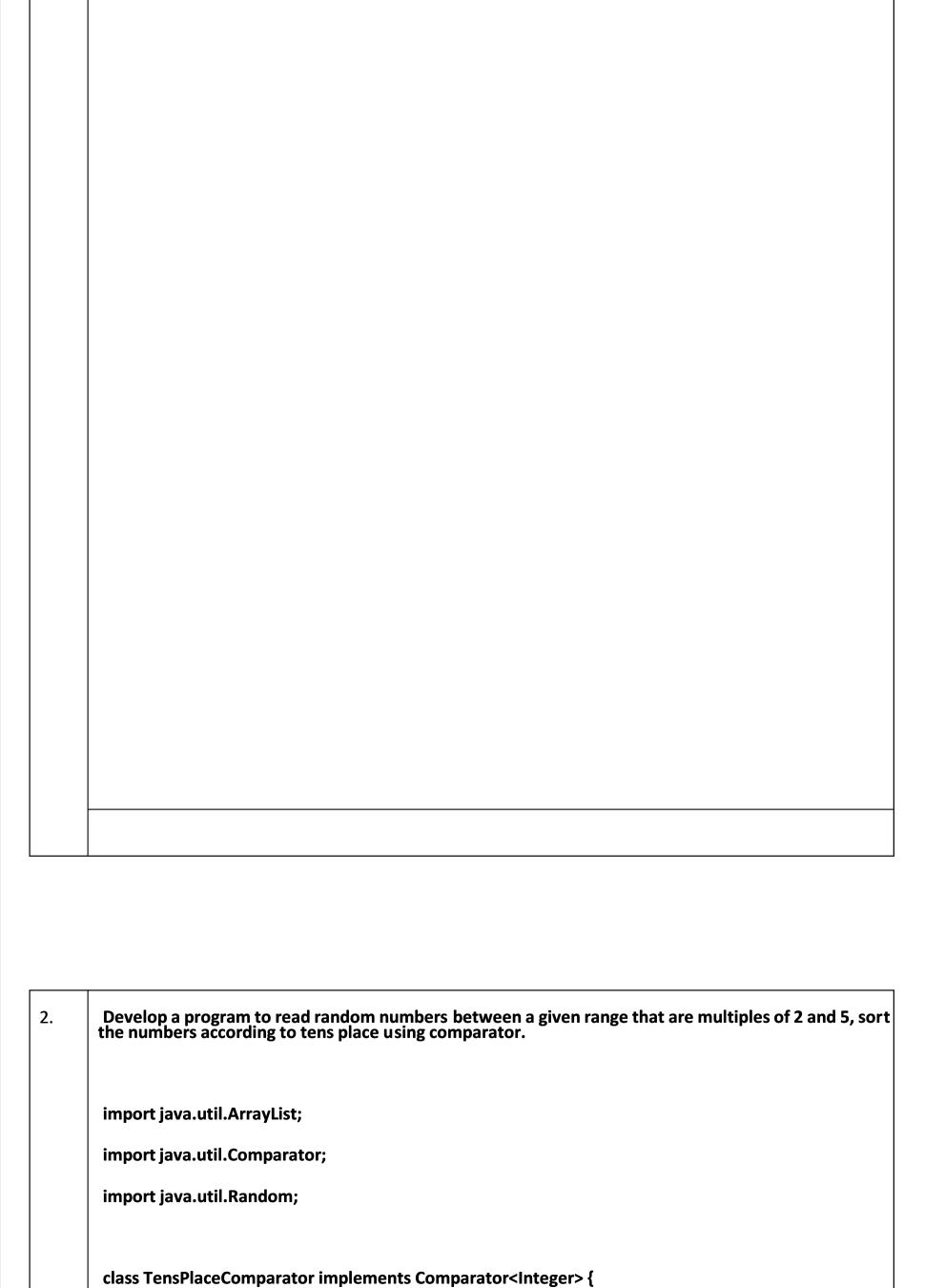| CO1 | solve the given problem using appropriate collection class/interface |
|------|---------------------------------------------------------------------|
| CO2 | Demonstrate the concepts of String operations in Java |
| CO3 | Demonstrate by Apply the concepts of Swings to build Java applications |
| CO4 | Implementation web based applications using Java servlets, JSP and JDBC to build database applications |

| SL. No. | Lab Experiments |
|---------|-----------------|
| 1. | Implement a java program to demonstrate creating an ArrayList, adding elements, removing elements, sorting elements of ArrayList. Also illustrate the use of toArray() method. |

```java
import java.util.ArrayList;
import java.util.Collections;

public class ArrayListDemo {
    public static void main(String[] args) {
        // Creating an ArrayList
        ArrayList<String> arrayList = new ArrayList<>();

        // Adding elements to the ArrayList
        arrayList.add("Apple");
        arrayList.add("Banana");
        arrayList.add("Orange");
        arrayList.add("Mango");
        arrayList.add("Pineapple");

        // Displaying the ArrayList before sorting
        System.out.println("ArrayList before sorting: " + arrayList);

        // Sorting elements of ArrayList
        Collections.sort(arrayList);

        // Displaying the ArrayList after sorting
        System.out.println("ArrayList after sorting: " + arrayList);

        // Removing an element from the ArrayList
        arrayList.remove("Orange");

        // Displaying the ArrayList after removing an element
```

```java
        System.out.println("ArrayList after removing an element: " + arrayList);

        // Using toArray() method
        String[] array = arrayList.toArray(new String[0]);
```

```java
        // Displaying the converted array
        System.out.println("Array elements: ");
        for (String element : array) {
            System.out.println(element);
        }
    }
}
```

Output:

ArrayList before sorting: [Apple, Banana, Orange, Mango, Pineapple]

ArrayList after sorting: [Apple, Banana, Mango, Orange, Pineapple]

ArrayList after removing an element: [Apple, Banana, Mango, Pineapple]

Array elements:

Apple

Banana

Mango
Pineapple

Java program demonstrating creating an ArrayList, adding elements, removing elements, sorting elements of ArrayList, and using the `toArray()` method:

This program creates an ArrayList of strings, adds elements to it, sorts the elements, removes an element, and then converts the ArrayList to an array using the `toArray()` method.

| | |
|---|---|
| | |

2. **Develop a program to read random numbers between a given range that are multiples of 2 and 5, sort the numbers according to tens place using comparator.**

```
import java.util.ArrayList;

import java.util.Comparator;

import java.util.Random;


class TensPlaceComparator implements Comparator<Integer> {

    @Override
```

```java
    public int compare(Integer num1, Integer num2) {

        int tensPlace1 = (num1 % 100) / 10; // Extracting tens place of num1

        int tensPlace2 = (num2 % 100) / 10; // Extracting tens place of num2

        return Integer.compare(tensPlace1, tensPlace2);

    }

}


public class RandomNumberSort {

    public static void main(String[] args) {

        int lowerBound = 10;

        int upperBound = 1000;

        int count = 20; // Number of random numbers to generate


        ArrayList<Integer> numbers = new ArrayList<>();

        Random random = new Random();


        // Generate random numbers between lowerBound and upperBound

        for (int i = 0; i < count; i++) {
```

```java
            int randomNumber = random.nextInt(upperBound - lowerBound) + lowerBound;

            if (randomNumber % 2 == 0 && randomNumber % 5 == 0) {

                numbers.add(randomNumber); // Add only multiples of 2 and 5

            }

        }


        // Display original list

        System.out.println("Original list:");

        for (int number : numbers) {

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

        }

        System.out.println();
```

```java
        // Sort the list based on tens place using Comparator

        numbers.sort(new TensPlaceComparator());


        // Display sorted list

        System.out.println("\nSorted list based on tens place:");

        for (int number : numbers) {

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

        }

    }

}
```

Output:

Original list:

Sorted list based on tens place:

=== Code Execution Successful ===

Java program that reads random numbers between a given range, filters out the numbers that are multiples of 2 and 5, sorts the numbers according to their tens place using a custom comparator:

This program generates random numbers between a given range, filters out the numbers that are multiples of 2 and 5, and then sorts these numbers according to their tens place using a custom comparator (`TensPlaceComparator`).

| | |
|---|---|
| 3. | Implement a java program to illustrate storing user defined classes in collection

```java
import java.util.ArrayList;

class Person {

    private String name;

    private int age;

    public Person(String name, int age) {

        this.name = name;

        this.age = age;

    }


    public String getName() {

        return name;
```
 |

```java
    }

    public int getAge() {

        return age;

    }



    @Override

    public String toString() {
        return "Person{" +

            "name='" + name + '\''

+

            ", age=" + age +

            '}';

    }

}


public class Main {

    public static void

main(String[] args) {

        // Create a list to store
Person objects

        ArrayList<Person>
peopleList = new ArrayList<>();
```

```java
        // Add some Person objects
to the list

        peopleList.add(new
Person("Alice", 30));

        peopleList.add(new
Person("Bob", 25));

        peopleList.add(new
Person("Charlie", 35));



        // Display the contents of
the list

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

        for (Person person :
peopleList) {


System.out.println(person);

        }

    }

}
```

Output:

People in the list:

Person{name='Alice', age=30}

Person{name='Bob', age=25}

Person{name='Charlie', age=35}

=== Code Execution Successful====

Java program illustrating storing user-defined classes in a collection. In this example, we'll create a `Person`

class and store instances of this class in an `ArrayList`:

In this program:

- We define a `Person` class with `name` and `age` attributes.
- We override the `toString()` method in the `Person` class to provide a meaningful representation of a `Person` object.
- In the `Main` class, we create an `ArrayList` named `peopleList` to store `Person` objects.
- We create some `Person` objects and add them to the `peopleList`.
- Finally, we iterate over the `peopleList` and print out each `Person` object using their `toString()` method.

| 4. | Implement a java program to illustrate the use of different types of string class constructors The most commonly used constructors of the String class are as follows: |
|---|---|

```java
public class StringConstructorExample {
    public static void main(String[] args) {
        // Using no-argument constructor
        String str1 = new String();
        System.out.println("String created using no-argument constructor: \"" + str1 + "\"");

        // Using char array constructor
        char[] charArray = {'H', 'e', 'l', 'l', 'o'};
        String str2 = new String(charArray);
        System.out.println("String created using char array constructor: \"" + str2 + "\"");

        // Using byte array constructor with specified character encoding
        byte[] byteArray = {72, 101, 108, 108, 111}; // ASCII values of 'Hello'
        String str3 = new String(byteArray);
        System.out.println("String created using byte array constructor with default encoding: \"" + str3 + "\"");
        String str4 = new String(byteArray, 0, 5); // Using only first 5 bytes
        System.out.println("String created using byte array constructor with specified encoding and length: \"" + str4 + "\"");

        // Using StringBuilder constructor
        StringBuilder stringBuilder = new StringBuilder("Hello");

        String str5 = new String(stringBuilder);
        System.out.println("String created using StringBuilder constructor: \"" + str5 + "\"");

        // Using StringBuffer constructor
        StringBuffer stringBuffer = new StringBuffer("Hello");
        String str6 = new String(stringBuffer);
        System.out.println("String created using StringBuffer constructor: \"" + str6 + "\"");
    }
}
```

Output:

String created using no-argument constructor: ""

String created using char array constructor: "Hello"

String created using byte array constructor with default encoding: "Hello"

String created using byte array constructor with specified encoding and length: "Hello"

String created using StringBuilder constructor: "Hello"

String created using StringBuffer constructor: "Hello"

: Java program that illustrates the use of different types of constructors available in the `String` class:

In this program

- The no-argument constructor creates an empty string.

- The char array constructor creates a string from the given character array.

- The byte array constructor creates a string from the given byte array using the default character encoding (UTF-8 in most cases).

- The byte array constructor with specified encoding and length creates a string from a subset of the byte array.

- The StringBuilder constructor creates a string from the contents of the `StringBuilder` object.

- The StringBuffer constructor creates a string from the contents of the `StringBuffer` object.

| 5. | Implement a java program to illustrate the use of different types of character extraction, string comparison, string search and string modification methods. |
|----|---|

```java
public class StringMethodsExample {
    public static void main(String[] args) {
        // Character extraction
        String str = "Hello, World!";
        char firstChar = str.charAt(0);
        char lastChar = str.charAt(str.length() - 1);
        System.out.println("First character: " + firstChar);
        System.out.println("Last character: " + lastChar);

        // String comparison
        String str1 = "hello";
        String str2 = "HELLO";
        System.out.println("str1.equals(str2): " + str1.equals(str2));
        System.out.println("str1.equalsIgnoreCase(str2): " + str1.equalsIgnoreCase(str2));

        // String search
        String sentence = "The quick brown fox jumps over the lazy dog";
        boolean containsFox = sentence.contains("fox");
        boolean startsWithThe = sentence.startsWith("The");
        boolean endsWithDog = sentence.endsWith("dog");
        int indexOfFox = sentence.indexOf("fox");
        int lastIndexOfThe = sentence.lastIndexOf("The");
        System.out.println("Contains 'fox': " + containsFox);
        System.out.println("Starts with 'The': " + startsWithThe);
        System.out.println("Ends with 'dog': " + endsWithDog);
        System.out.println("Index of 'fox': " + indexOfFox);
        System.out.println("Last index of 'The': " + lastIndexOfThe);

        // String modification
        String originalString = "   Hello, World!   ";
```

```java
        String trimmedString = originalString.trim();
        String lowerCaseString = originalString.toLowerCase();
        String upperCaseString = originalString.toUpperCase();
        String replacedString = originalString.replace("World", "Universe");
        String substring = originalString.substring(7, 12); // Extract substring from index 7 to 11
        System.out.println("Trimmed string: \"" + trimmedString + "\"");
```

```
        System.out.println("Trimmed string: \"" + trimmedString + "\"");
        System.out.println("Lowercase string: \"" + lowerCaseString + "\"");
        System.out.println("Uppercase string: \"" + upperCaseString + "\"");
        System.out.println("Replaced string: \"" + replacedString + "\"");
        System.out.println("Substring: \"" + substring + "\"");
    }
}
```

Output:

First character: H

Last character: !

str1.equals(str2): false

str1.equalsIgnoreCase(str2): true

Contains 'fox': true

Starts with 'The': true

Ends with 'dog': true

Index of 'fox': 16

Last index of 'The': 0

Trimmed string: "Hello, World!"

Lowercase string: "   hello, world!   "

Uppercase string: "   HELLO, WORLD!   "

Replaced string: "   Hello, Universe!   "

Substring: "o, Wo"

This program demonstrates:

- Character extraction using the `charAt()` method.
- Substring extraction using the `substring()` method.
- String comparison using the `equals()` and `equalsIgnoreCase()` methods.
- String search using the `contains()` method.
- String modification using the `trim()`, `replace()`, `toUpperCase()`, and `toLowerCase()` methods.