**FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG**

**Department of Computer Science and Engineering**

|  |  |  |  |
| --- | --- | --- | --- |
| **Academic Year** | **2025-2026** | **Estimated Time** | **Experiment No. 4 – 02 Hours** |
| **Course & Semester** | **S.E. CSE** | **Subject Name** | **Object Oriented**  **Programming with Java Lab** |
| **Module No.** | **04** | **Chapter Title** | Arrays and Vector |
| **Experiment Type** | **Software Performance** | **Subject Code** | 25PCC12CS07 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of Student** | Atharva Dharmendra Jagtap | **Roll No.** | 10937 |
| **Date of Performance.:** |  | **Date of Submission.:** |  |
| **CO Mapping** | **CO1. Demonstrate Proficiency in Fundamentals of Java.**  **CO2. Apply Object-Oriented Programming Principles to given problem.** | | |

**Objective of Experiment:**

To explore and demonstrate the concepts of Arrays, Array Lists and Vectors in Java.

**Pre-Requisite:** Any programming language like C, C++

**Tools:** Java IDE

**Theory:**

**A. Arrays**

Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value. To declare an array, define the variable type with **square brackets.**

* String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
* int[] myNum = {10, 20, 30, 40};

You can access an array element by referring to the index number.

* System.out.println(cars[0]); // Outputs Volvo

To change the value of a specific element, refer to the index number.

* cars[0] = "Opel";

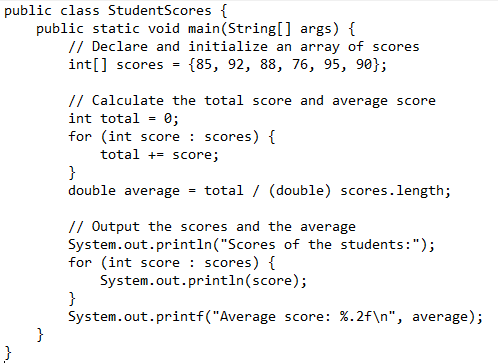
To find out how many elements an array has, use the length property.

* System.out.println(cars.length); // Outputs 4

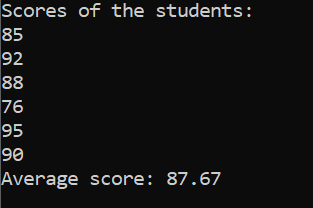
In Java, a matrix is typically implemented using a 2D array. The following is the syntax to declare and initialize a 2D array in Java.

* int[][] matrix = { {1, 2, 3}, {4, 5, 6}, {7, 8, 9} };

Simple program to demonstrate the concept of Arrays in java

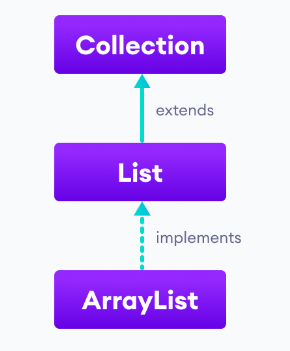


OUTPUT:



**B. Array List**

In Java, we use the ArrayList class to implement the functionality of resizable-arrays. It implements the List interface of the collections framework.



In Java, we need to declare the size of an array before we can use it. Once the size of an array is declared, it's hard to change it. To handle this issue, we can use the ArrayList class. It allows us to create resizable arrays. Unlike arrays, arraylists can automatically adjust their capacity when we add or remove elements from them. Hence, arraylists are also known as dynamic arrays.

Before using ArrayList, we need to import the java.util.ArrayList package first. Here is how we can create arraylists in Java.

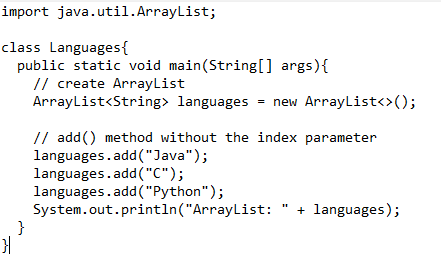
* ArrayList<Type> arrayList= new ArrayList<>();

Here, Type indicates the type of an arraylist. For example, Integer or String.

**Basic Operations on ArrayList**

The ArrayList class provides various methods to perform different operations on arraylists. Some commonly used operations are:

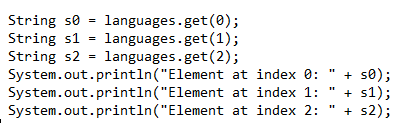
1. Add elements: To add a single element to the arraylist, we use the add() method of the ArrayList class.
2. Access elements: To access an element from the arraylist, we use the get() method of the ArrayList class.
3. Change elements: To change elements of the arraylist, we use the set() method of the ArrayList class.
4. Remove elements: To remove an element from the arraylist, we can use the remove() method of the ArrayList class.
5. Adding elements to an ArrayList



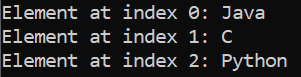
OUTPUT:



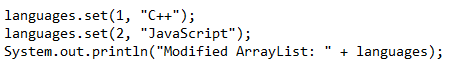
1. Access ArrayList elements



OUTPUT:



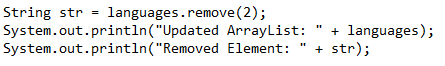
1. Change ArrayList elements



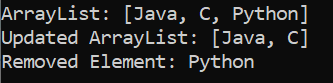
OUTPUT:



1. Remove ArrayList Elements



OUTPUT:



**C. Vectors**

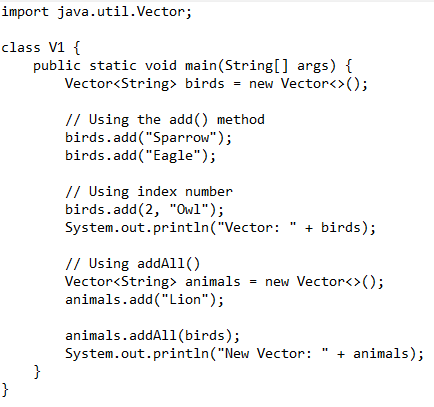
The Vector class implements a growable array of objects. Like an array, it contains components that can be accessed using an integer index. However, the size of a Vector can grow or shrink as needed to accommodate adding and removing items after the Vector has been created. The Vector class is an implementation of the List interface that allows us to create resizable-arrays similar to the ArrayList class.

* Syntax to create vectors in Java: Vector<Type> vector = new Vector<>(); Type indicates the type of a linked list. For example: Integer , String.

ArrayList is not synchronized whereas Vector is synchronized. So, ArrayList is faster than Vector. ArrayList prefers the iterator interface to traverse the components whereas Vector prefers Enumeration or Iterator interface to traverse the elements.

Some common methods when working with vectors:

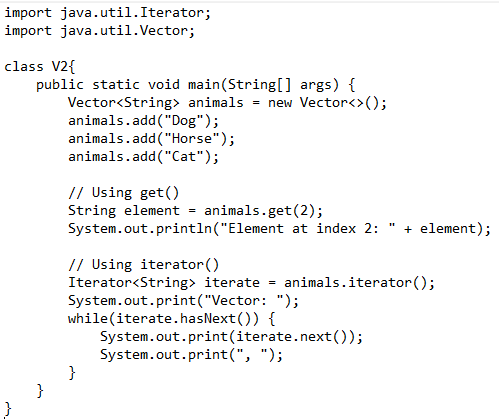
1. Add Elements to Vector.
   * add(element) - adds an element to vectors
   * add(index, element) - adds an element to the specified position
   * addAll(vector) - adds all elements of a vector to another vector



**OUTPUT:**



1. Access vector Elements
   * get(index) - returns an element specified by the index
   * iterator() - returns an iterator object to sequentially access vector elements

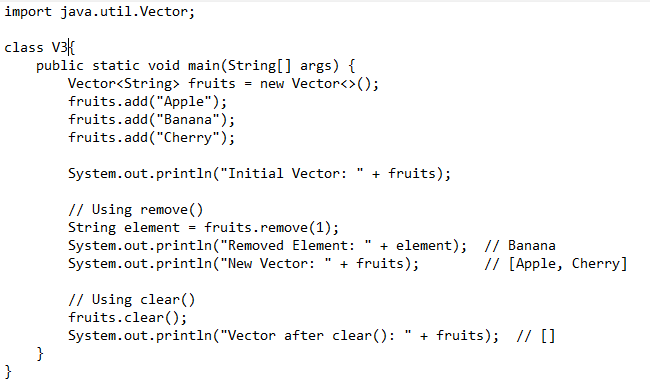


**OUTPUT:**

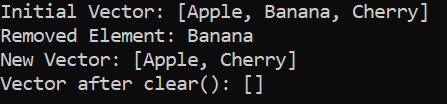


1. Remove Vector Elements

* remove(index) - removes an element from specified position
* removeAll() - removes all the elements
* clear() - removes all elements. It is more efficient than removeAll()



**OUTPUT:**



**Problem Statement:**

***Contact Management App:***

Design and implement **Java application** to demonstrate the use of **Array, ArrayList, and Vector** with real-life scenarios. The program should provide the following functionalities:

1. **Array (Employee Database Management):**

* Store Salary of 5 Employees using an **array**.
* Find the **highest, lowest, and average Salary**.

1. **ArrayList (Product Database Management):**

* Use an **ArrayList** to maintain a list of product names in a store.
* Allow the user to **add, remove, change, and display** product dynamically.

1. **Vector (Library Management System):**

* Use a **Vector** to store book details (b\_id, name, price).
* Allow the user to **add new book, search by name, and display all books**.

**Code:**

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

import java.util.Vector;

public class Main {

    private static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {

        System.out.println("--- Data Structure Demonstration ---");

        // 1. Array: Employee Database Management

        System.out.println("\n===== 1. Employee Salary Management (Array) =====");

        employeeSalaryManagement();

        // 2. ArrayList: Product Database Management

        System.out.println("\n===== 2. Product Inventory Management (ArrayList) =====");

        productInventoryManagement();

        // 3. Vector: Library Management System

        System.out.println("\n===== 3. Library Management System (Vector) =====");

        libraryManagementSystem();

        scanner.close(); // Close the scanner when all operations are done

        System.out.println("\n--- Demonstration Complete ---");

    }

    // --- 1. Array: Employee Salary Management ---

    public static void employeeSalaryManagement() {

        // Store Salary of 5 Employees using an array.

        double[] salaries = new double[5]; // Array to store 5 salaries

        System.out.println("Please enter the salaries for 5 employees:");

        for (int i = 0; i < salaries.length; i++) {

            System.out.print("Enter salary for Employee " + (i + 1) + ": $");

            while (!scanner.hasNextDouble()) {

                System.out.println("Invalid input. Please enter a numeric salary.");

                scanner.next(); // Consume invalid input

                System.out.print("Enter salary for Employee " + (i + 1) + ": $");

            }

            salaries[i] = scanner.nextDouble();

            scanner.nextLine(); // Consume newline

        }

        System.out.println("\n--- Employee Salary Report ---");

        // Display all salaries

        System.out.print("All Salaries: [");

        for (int i = 0; i < salaries.length; i++) {

            System.out.printf("$%.2f", salaries[i]);

            if (i < salaries.length - 1) {

                System.out.print(", ");

            }

        }

        System.out.println("]");

        // Find highest, lowest, and average Salary

        if (salaries.length == 0) {

            System.out.println("No salaries to process.");

            return;

        }

        double highestSalary = salaries[0];

        double lowestSalary = salaries[0];

        double totalSalary = 0;

        for (double salary : salaries) {

            if (salary > highestSalary) {

                highestSalary = salary;

            }

            if (salary < lowestSalary) {

                lowestSalary = salary;

            }

            totalSalary += salary;

        }

        double averageSalary = totalSalary / salaries.length;

        System.out.printf("Highest Salary: $%.2f%n", highestSalary);

        System.out.printf("Lowest Salary: $%.2f%n", lowestSalary);

        System.out.printf("Average Salary: $%.2f%n", averageSalary);

        System.out.println("------------------------------------");

    }

    // --- 2. ArrayList: Product Database Management ---

    public static void productInventoryManagement() {

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

        productList.add("Laptop");

        productList.add("Mouse");

        productList.add("Keyboard");

        productList.add("Monitor");

        int choice;

        do {

            System.out.println("\nProduct Management Menu:");

            System.out.println("1. Add Product");

            System.out.println("2. Remove Product");

            System.out.println("3. Change Product Name");

            System.out.println("4. Display All Products");

            System.out.println("5. Back to Main Menu");

            System.out.print("Enter your choice: ");

            while (!scanner.hasNextInt()) {

                System.out.println("Invalid input. Please enter a number.");

                scanner.next(); // Consume invalid input

                System.out.print("Enter your choice: ");

            }

            choice = scanner.nextInt();

            scanner.nextLine(); // Consume newline

            switch (choice) {

                case 1:

                    System.out.print("Enter product name to add: ");

                    String newProduct = scanner.nextLine();

                    if (!newProduct.trim().isEmpty()) {

                        productList.add(newProduct);

                        System.out.println("'" + newProduct + "' added successfully.");

                    } else {

                        System.out.println("Product name cannot be empty.");

                    }

                    break;

                case 2:

                    if (productList.isEmpty()) {

                        System.out.println("No products to remove.");

                        break;

                    }

                    System.out.print("Enter product name to remove: ");

                    String productToRemove = scanner.nextLine();

                    if (productList.remove(productToRemove)) {

                        System.out.println("'" + productToRemove + "' removed successfully.");

                    } else {

                        System.out.println("'" + productToRemove + "' not found in the list.");

                    }

                    break;

                case 3:

                    if (productList.isEmpty()) {

                        System.out.println("No products to change.");

                        break;

                    }

                    System.out.print("Enter current product name to change: ");

                    String oldName = scanner.nextLine();

                    int index = productList.indexOf(oldName);

                    if (index != -1) {

                        System.out.print("Enter new product name: ");

                        String newName = scanner.nextLine();

                        if (!newName.trim().isEmpty()) {

                            productList.set(index, newName);

                            System.out.println("Product name changed from '" + oldName + "' to '" + newName + "'.");

                        } else {

                            System.out.println("New product name cannot be empty.");

                        }

                    } else {

                        System.out.println("'" + oldName + "' not found in the list.");

                    }

                    break;

                case 4:

                    displayProducts(productList);

                    break;

                case 5:

                    System.out.println("Returning to main menu.");

                    break;

                default:

                    System.out.println("Invalid choice. Please enter a number between 1 and 5.");

            }

        } while (choice != 5);

    }

    private static void displayProducts(ArrayList<String> products) {

        if (products.isEmpty()) {

            System.out.println("Product list is empty.");

            return;

        }

        System.out.println("\n--- Current Products ---");

        for (int i = 0; i < products.size(); i++) {

            System.out.println((i + 1) + ". " + products.get(i));

        }

        System.out.println("------------------------");

    }

    // --- 3. Vector: Library Management System ---

    // Inner class to represent a Book

    static class Book {

        String bookId;

        String name;

        double price;

        public Book(String bookId, String name, double price) {

            this.bookId = bookId;

            this.name = name;

            this.price = price;

        }

        @Override

        public String toString() {

            return String.format("ID: %s | Name: %-25s | Price: $%.2f", bookId, name, price);

        }

    }

    public static void libraryManagementSystem() {

        Vector<Book> library = new Vector<>();

        // Add some initial books

        library.add(new Book("B001", "The Lord of the Rings", 25.99));

        library.add(new Book("B002", "Pride and Prejudice", 12.50));

        library.add(new Book("B003", "1984", 15.00));

        int choice;

        do {

            System.out.println("\nLibrary Management Menu:");

            System.out.println("1. Add New Book");

            System.out.println("2. Search Book by Name");

            System.out.println("3. Display All Books");

            System.out.println("4. Back to Main Menu");

            System.out.print("Enter your choice: ");

            while (!scanner.hasNextInt()) {

                System.out.println("Invalid input. Please enter a number.");

                scanner.next(); // Consume invalid input

                System.out.print("Enter your choice: ");

            }

            choice = scanner.nextInt();

            scanner.nextLine(); // Consume newline

            switch (choice) {

                case 1:

                    System.out.print("Enter Book ID: ");

                    String id = scanner.nextLine();

                    System.out.print("Enter Book Name: ");

                    String name = scanner.nextLine();

                    System.out.print("Enter Book Price: $");

                    while (!scanner.hasNextDouble()) {

                        System.out.println("Invalid input. Please enter a numeric price.");

                        scanner.next(); // Consume invalid input

                        System.out.print("Enter Book Price: $");

                    }

                    double price = scanner.nextDouble();

                    scanner.nextLine(); // Consume newline

                    if (!id.trim().isEmpty() && !name.trim().isEmpty() && price > 0) {

                        library.add(new Book(id, name, price));

                        System.out.println("Book '" + name + "' added successfully.");

                    } else {

                        System.out.println("Invalid book details. ID, Name cannot be empty, Price must be positive.");

                    }

                    break;

                case 2:

                    System.out.print("Enter book name to search: ");

                    String searchName = scanner.nextLine();

                    boolean found = false;

                    for (Book book : library) {

                        if (book.name.equalsIgnoreCase(searchName)) { // Case-insensitive search

                            System.out.println("Book Found: " + book);

                            found = true;

                            break; // Stop after finding the first match

                        }

                    }

                    if (!found) {

                        System.out.println("Book '" + searchName + "' not found in the library.");

                    }

                    break;

                case 3:

                    displayBooks(library);

                    break;

                case 4:

                    System.out.println("Returning to main menu.");

                    break;

                default:

                    System.out.println("Invalid choice. Please enter a number between 1 and 4.");

            }

        } while (choice != 4);

    }

    private static void displayBooks(Vector<Book> books) {

        if (books.isEmpty()) {

            System.out.println("The library is empty.");

            return;

        }

        System.out.println("\n--- All Books in Library ---");

        System.out.println(String.format("%-10s | %-25s | %s", "Book ID", "Book Name", "Price"));

        System.out.println("---------------------------------------------------------------");

        for (Book book : books) {

            System.out.println(String.format("%-10s | %-25s | $%.2f", book.bookId, book.name, book.price));

        }

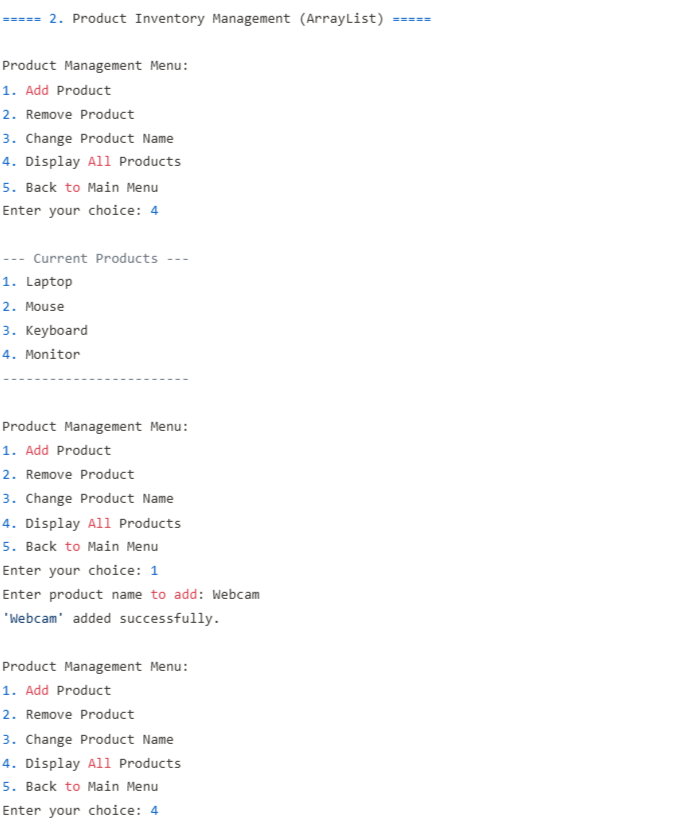
        System.out.println("---------------------------------------------------------------");

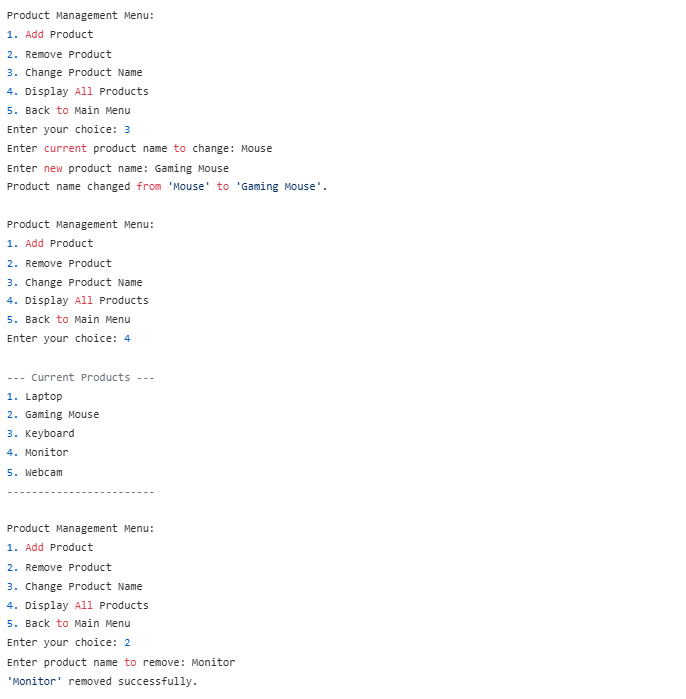
    }

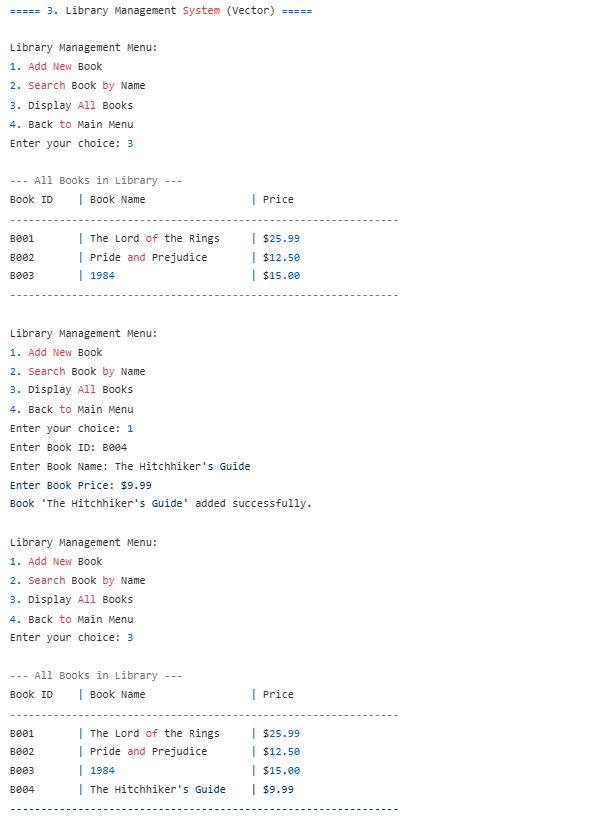
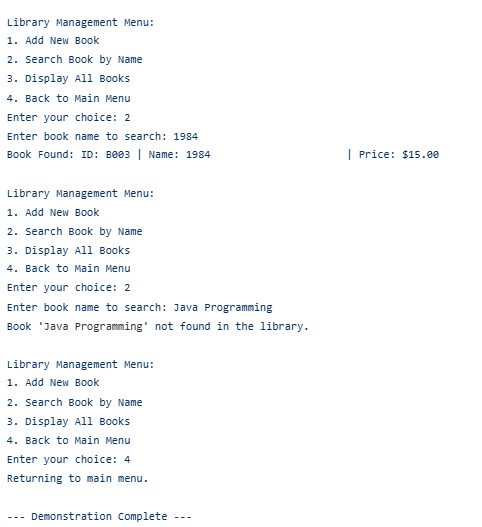
}

**Output:**

****

****

****

**** ****

**Post Lab Question:**

Add all programs implemented during practical session.

|  |  |  |  |
| --- | --- | --- | --- |
| **On time Completion and Submission (2)** | **Knowledge of the topic (4)** | **Implementation and Output (4)** | **Total (10)** |
|  |  |  |  |

|  |  |
| --- | --- |
| **References:Study Materials**  [**https://www.w3schools.com/java/**](https://www.w3schools.com/java/)  [**https://www.geeksforgeeks.org/java/**](https://www.geeksforgeeks.org/java/)  <https://www.codecademy.com/learn/learn-java>  https://www.programiz.com/java-programming/arraylist | **Video Channels**:  [**https://www.youtube.com/user/programmingwithmosh**](https://www.youtube.com/user/programmingwithmosh)  [**https://www.youtube.com/c/TheNetNinja**](https://www.youtube.com/c/TheNetNinja)  [**https://www.youtube.com/c/Freecodecamp**](https://www.youtube.com/c/Freecodecamp)  [**https://www.youtube.com/user/Simplilearn**](https://www.youtube.com/user/Simplilearn) |
| **Study Materials used for Demo**  <Add links here> | |

**Note:-students are expected to paste screenshot of the program with output**