# Ex No 5.1 Write a Java program to create a new array list, add some colors and print the collection.

**Aim**

To write a Java program to create a new array list, add some colors and print the collection.

# Algorithm

1. Import Required - Package Import java.util.ArrayList.
2. Create an ArrayList - Declare and initialize an ArrayList<String>.
3. Add Colors to the List - Use add() method to insert color names (e.g., "Red", "Blue", "Green").
4. Print the Collection - Use a loop or System.out.println() to display the colors.
5. End the Program - Ensure the program executes successfully.

# Program

import java.util.ArrayList;

public class ColorList {

public static void main(String[] args) {

// Create an ArrayList to store colors ArrayList<String> colors = new ArrayList<>();

// Add colors to the ArrayList colors.add("Red");

colors.add("Blue"); colors.add("Green"); colors.add("Yellow"); colors.add("Purple");

// Print the ArrayList System.out.println("Color List: " + colors);

}

}

# Output

**Result**

Thus a java program to create a new array list, add some colors and print the collection has been completed successfully and output is verified.

# Ex No 5.2 Write a Java program to shuffle elements in array list

**Aim**

To write a Java program to shuffle elements in array list

# Algorithm

1. Import Required Packages - Import java.util.ArrayList and java.util.Collections.
2. Create an ArrayList - Declare and initialize an ArrayList<String>.
3. Add Elements to the List - Use add() method to insert elements (e.g., colors, numbers, etc.).
4. Shuffle the Elements - Use Collections.shuffle() to randomize the order of elements.
5. Print the Shuffled List - Use System.out.println() to display the shuffled elements.
6. End the Program - Ensure the program runs successfully.

# Program

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

public class ShuffleArrayList {

public static void main(String[] args) {

// Create an ArrayList and add elements ArrayList<String> colors = new ArrayList<>(); colors.add("Red");

colors.add("Blue"); colors.add("Green"); colors.add("Yellow"); colors.add("Purple");

// Print original list System.out.println("Original List: " + colors);

// Shuffle the ArrayList Collections.shuffle(colors);

// Print shuffled list System.out.println("Shuffled List: " + colors);

}

}

# Output

**Result**

Thus, writing a Java program to shuffle elements in an array list has been completed successfully and output is verified.

# Ex No 5.3 Write a Java program to iterate through all elements in a linked list

**Aim**

To write a Java program to iterate through all elements in a linked list

# Algorithm

1. Import Required Packages - Import java.util.LinkedList and java.util.Iterator.
2. Create a LinkedList - Declare and initialize a LinkedList<String>.
3. Add Elements to the List - Use add() method to insert elements.
4. Iterate Using a Loop - Use a for-each loop or for loop to access each element.
5. Iterate Using an Iterator - Create an Iterator and use a while loop with hasNext().
6. Print Each Element - Use System.out.println() to display elements.
7. End the Program - Ensure smooth execution.

# Program

**im**port java.util.LinkedList; import java.util.Iterator;

public class LinkedListIteration {

public static void main(String[] args) {

// Create a LinkedList and add elements LinkedList<String> colors = new LinkedList<>(); colors.add("Red");

colors.add("Blue"); colors.add("Green"); colors.add("Yellow"); colors.add("Purple");

// Method 1: Using a for-each loop System.out.println("Iterating using for-each loop:"); for (String color : colors) {

System.out.println(color);

}

// Method 2: Using an Iterator System.out.println("\nIterating using Iterator:"); Iterator<String> iterator = colors.iterator(); while (iterator.hasNext()) {

System.out.println(iterator.next());

}

// Method 3: Using a for loop with get(index) System.out.println("\nIterating using for loop with index:"); for (int i = 0; i < colors.size(); i++) {

System.out.println(colors.get(i));

}

}

}

# Output

**Result**

Thus, writing a Java program to iterate through all elements in a linked list has been completed successfully and output is verified.

# Ex No 5.4 Write a Java program to create an ArrayList of Student ( id,name,dept,age) objects and search for particular Student objects based on id number.

**Aim**

To write a Java program to create an ArrayList of Student ( id,name,dept,age) objects and search for particular Student objects based on id number.

# Algorithm

1. Import Required Packages - Import java.util.ArrayList.
2. Define a Student Class - Create a class with attributes: id, name, dept, and age. Define a constructor to initialize the attributes.
3. Create an ArrayList of Student Objects - Declare and initialize ArrayList<Student>.
4. Add Student Objects to the List - Use add() method to insert multiple student records.
5. Search for a Student by ID - Use a loop (for or for-each) to iterate through the list.
6. Compare each student's id with the given ID.
7. Print the Student Details if Found If a match is found, display student details using System.out.println().
8. Handle Case When Student is Not Found - Print a message if no matching student is found.
9. End the Program - Ensure smooth execution.

# Program

import java.util.ArrayList; import java.util.Scanner;

// Student class class Student {

int id;

String name; String department; int age;

// Constructor

public Student(int id, String name, String department, int age) { this.id = id;

this.name = name; this.department = department; this.age = age;

}

// Display student details public void display() {

System.out.println("ID: " + id + ", Name: " + name + ", Dept: " + department + ", Age: " +

age);

}

}

public class StudentSearch {

public static void main(String[] args) {

// Create an ArrayList of Student objects ArrayList<Student> students = new ArrayList<>();

// Adding Student objects

students.add(new Student(101, "Alice", "Computer Science", 20)); students.add(new Student(102, "Bob", "Mechanical", 22)); students.add(new Student(103, "Charlie", "Electrical", 21)); students.add(new Student(104, "David", "Civil", 23));

// User input for searching

Scanner scanner = new Scanner(System.in); System.out.print("Enter Student ID to search: "); int searchId = scanner.nextInt();

scanner.close();

// Search for the student boolean found = false;

for (Student student : students) { if (student.id == searchId) {

System.out.println("Student Found:"); student.display();

found = true; break;

}

}

if (!found) {

System.out.println("Student with ID " + searchId + " not found.");

}

}

}

# Output

**Result**

Thus, writing a Java program to create an ArrayList of Student ( id,name,dept,age) objects and search for particular Student objects based on id number. has been completed successfully and output is verified.

# Ex No 5.5 Write a Java program to create an ArrayList which will be able to store only char and String but not any other data type.

**Aim**

To write a Java program to create an ArrayList which will be able to store only char and String but not any other data type.

# Algorithm

1. Create an ArrayList of Type Character and String - Use ArrayList<Object> to store both Character and String values.
2. Add Only char and String Values - Use instanceof to ensure only Character and String types are added.
3. Iterate and Display Elements
4. Loop through the list and print the stored values.

# Program

import java.util.ArrayList;

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

// Create an ArrayList that can store only Character and String ArrayList<Object> list = new ArrayList<>();

// Adding elements addElement(list, 'A'); // Character addElement(list, "Hello"); // String addElement(list, 'B'); addElement(list, "Java");

// Attempting to add an integer (should not be allowed) addElement(list, 100); // This should be rejected

// Print the valid elements

System.out.println("Valid Char & String List: " + list);

}

// Method to add only Character or String to the ArrayList

public static void addElement(ArrayList<Object> list, Object element) { if (element instanceof Character || element instanceof String) {

list.add(element);

} else {

System.out.println("Error: Only Character and String are allowed! Attempted to add: " + element);

}

}

}

# Output

**Result**

Thus, writing a Java program to create an ArrayList which will be able to store only char and String but not any other data type. has been completed successfully and output is verified.

# Ex No 5.6 Write a Java program using Queue Collection for Cinema Ticket Sale.

**Aim**

To write a Java program using Queue Collection for Cinema Ticket Sale.

# Algorithm

1. Create a Queue for Customers - Use Queue<String> with LinkedList to store customer names in the order they arrive.
2. Process Ticket Sales - Use poll() to remove and serve customers one by one.
3. Display the Queue Status - Print the queue before and after ticket sales to show remaining customers.

# Program

import java.util.LinkedList; import java.util.Queue; import java.util.Scanner;

class CinemaTicketQueue {

public static void main(String[] args) {

// Create a Queue for customers

Queue<String> ticketQueue = new LinkedList<>(); Scanner scanner = new Scanner(System.in);

// Adding customers to the queue

System.out.println("Enter customer names (type 'done' to stop): "); while (true) {

String name = scanner.nextLine();

if (name.equalsIgnoreCase("done")) break; ticketQueue.add(name);

}

// Processing ticket sales System.out.println("\nProcessing ticket sales..."); while (!ticketQueue.isEmpty()) {

String customer = ticketQueue.poll(); // Serve the first customer System.out.println("Ticket sold to: " + customer);

}

System.out.println("\nAll tickets sold. Queue is empty!"); scanner.close();

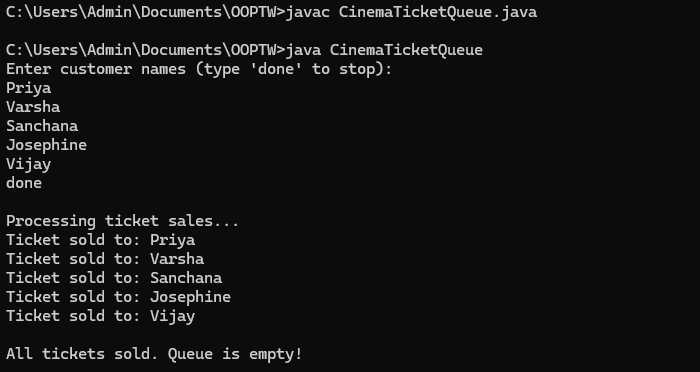
}

}

# Output

How It Works:

* Customers enter their names → They are added to the Queue.
* First customer in line gets served first (poll() method).
* Loop continues until all customers are served.



# Result

Thus, writing a Java program using Queue Collection for Cinema Ticket Sale has been completed successfully and output is verified.