**a**

**\*\***COLLECTIONS\*\*

# Develop an application which has an Arraylist of StudentNames, after a few days a few students changed their branch, help the faculty to update the list in the ascending order.

Requirements

1)Create an Arraylist of Strings

2)Add 5 students to the list

3)Using Iterator display the list

1. Add values at the location 0 and 1
2. Remove elements of your choice and display the final list

6)Display the list in Ascending order

**Aim:** The aim of this Java program is to demonstrate the use of ArrayList and various operations like adding, removing elements, and sorting. Specifically, it starts with an ArrayList of student names, adds new students, removes some students, and then sorts the final list in ascending order.

**Algorithm:**

1. Create an ArrayList of Strings to store student names.
2. Add five student names to the ArrayList.
3. Use an iterator to display the original list of student names.
4. Add two student names at positions 0 and 1 in the ArrayList.
5. Remove two student names of your choice from the ArrayList.
6. Display the final list of student names after the removal.
7. Sort the ArrayList in ascending order using **Collections.sort()**.
8. Display the sorted list of student names.

**Program Explanation:**

1. We start by creating an ArrayList of Strings named **obj** to store student names.
2. We add five student names to the **obj** ArrayList using the **add()** method.
3. We use a for-each loop to display the original list of student names.
4. Two more student names, "Anjali" and "Justin," are added at positions 0 and 1 in the ArrayList using the **add(index, element)** method.
5. We remove two student names, "Kevin" and "Karen," from the ArrayList using the **remove(element)** method.
6. The ArrayList is displayed again to show the final list of student names after removal.
7. To sort the list in ascending order, we use the **Collections.sort()** method.
8. Finally, we display the sorted list to show the student names in alphabetical order.

This program demonstrates the basic operations of ArrayList, such as adding, removing, and sorting elements, making it a useful example for understanding ArrayList manipulation in Java.

**Program:**

**package** labsheet10;

**import** java.util .\*;

**public** **class** ArrayListDemo {

**public** **static** **void** main(String args[]) {

/\* Creating ArrayList of type "String" which means

we can only add "String" elements \*/

ArrayList<String> obj = **new** ArrayList<String>();

/\*This is how we add elements to an ArrayList\*/

obj.add("Sushmitha");

obj.add("Kevin");

obj.add("Karen");

obj.add("Steve");

obj.add("Anuja");

// Displaying elements

System.***out***.println("Original ArrayList:");

**for**(String str:obj)

System.***out***.println(str);

obj.add(0, "Anjali");

obj.add(1, "Justin");

// Displaying elements

System.***out***.println("ArrayList after add operation:");

**for**(String str:obj)

System.***out***.println(str);

//Remove elements from ArrayList like this

obj.remove("Kevin"); //Removes "Chaitanya" from ArrayList o bj.remove("Karen"); //Removes "Harry" from ArrayList

// Displaying elements

System.***out***.println("ArrayList after remove operation:");

**for**(String str:obj)

System.***out***.println(str);

//Remove element from the specified index

obj.remove(1); //Removes Second element from the List

// Displaying elements

System.***out***.println("Final ArrayList:");

**for**(String str:obj)

System.***out***.println(str);

//Sorting

Collections.*sort*(obj);

System.***out***.println("Sorted Student List:");

**for**(String str:obj)

System.***out***.println(str);

}

}

**Output:**

Original ArrayList:

Sushmitha

Kevin

Karen

Steve

Anuja

ArrayList after add operation:

Anjali

Justin

Sushmitha

Kevin

Karen

Steve

Anuja

ArrayList after remove operation:

Anjali

Justin

Sushmitha

Karen

Steve

Anuja

Final ArrayList:

Anjali

Sushmitha

Karen

Steve

Anuja

Sorted Student List:

Anjali

Anuja

Karen

Steve

Sushmitha

1. As we are maintaining social distancing in the crowded area, queuing is the common system followed everywhere. Implement a queue for any number of persons, so that the first person in the queue will be given access first. Implement the operation of a queue without using Arrays.

**Aim:**

The aim of this Java program is to implement a basic queue data structure using an **ArrayList** without using arrays. The program allows users to insert people into the queue, remove people from the front of the queue (which simulates people being granted access), and display the current queue of people waiting in a crowded area while maintaining social distancing.

**Algorithm:**

1. Create a class **QueueArrayList** to represent the queue with an **ArrayList** to store the names of people in the queue. Maintain a variable **front** to keep track of the front of the queue.
2. Implement the following methods in the class:
   * **insert(name)**: Add a person's name to the end of the queue.
   * **delete()**: Remove and return the person at the front of the queue.
   * **display()**: Display the names of people in the queue.
3. In the **main** method:
   * Create an instance of **QueueArrayList**.
   * Use a loop to repeatedly display a menu of options and perform actions based on user input.
   * The options include inserting a person into the queue, removing a person from the front of the queue, displaying the current queue, and exiting the program.

**Program Explanation:**

1. The **QueueArrayList** class is defined with an **ArrayList** named **queueArrayList** to store the names of people and an integer variable **front** to represent the front of the queue.
2. The **insert(name)** method adds a person's name to the end of the queue by calling **add(name)** on the **queueArrayList**.
3. The **delete()** method removes and returns the person at the front of the queue by calling **remove(front)** on the **queueArrayList**.
4. The **display()** method uses a **ListIterator** to iterate through the **queueArrayList** and print the names of people in the queue.
5. In the **main** method, we create a **QueueArrayList** object, set up a menu for user interaction, and use a loop to handle user choices.
6. The menu includes options for inserting a person into the queue, removing a person from the front of the queue (simulating granting access), displaying the current queue, and exiting the program.
7. The program continues to run until the user chooses to exit (choosing option 4).

This program simulates a basic queue system using an **ArrayList** and allows users to perform queue-related operations such as adding and removing people from the queue and viewing the current queue.

**Program:**

**package** labsheet10;

**import** java.util.\*;

**import** java.util.ListIterator;

**import** java.util.Scanner;

**public** **class** QueueArrayList {

ArrayList<String> queueArrayList = **new** ArrayList<String>();

**int** front=0;

**public** **void** insert(String name){

queueArrayList.add(name);

}

**public** String delete(){

**return** queueArrayList.remove(front);

}

**public** **void** display(){

ListIterator<String> iterator = queueArrayList.listIterator(); **while**(iterator.hasNext()){

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

}

}

**public** **static** **void** main(String[] args) {

**int** choice=0;

Scanner scanner = **new** Scanner(System.***in***);

QueueArrayList qa = **new** QueueArrayList();

**do**{

System.***out***.println("1. Insert 2. Delete 3.Display 4.Exit");

System.***out***.println("Enter your choice");

choice = scanner.nextInt();

scanner.nextLine();

**switch**(choice){

**case** 1: System.***out***.println("Enter the name of the person to be in the queue");

String name = scanner.nextLine();

qa.insert(name); **break**;

**case** 2: System.***out***.println("Removed from Queue : "+

qa.delete());

**break**;

**case** 3: System.***out***.println("Waiting in Queue now...");

qa.display();

**break**;

**case** 4: System.*exit*(0);

}

}**while**(choice<=3);

}

}

**Output:**

1. Insert 2. Delete 3.Display 4.Exit

Enter your choice

1

Enter the name of the person to be in the queue

Sushmitha

1. Insert 2. Delete 3.Display 4.Exit

Enter your choice

1

Enter the name of the person to be in the queue

Kevin

1. Insert 2. Delete 3.Display 4.Exit

Enter your choice

1

Enter the name of the person to be in the queue

Karen

1. Insert 2. Delete 3.Display 4.Exit

Enter your choice

3

Waiting in Queue now...

Sushmitha

Kevin

Karen

1. Insert 2. Delete 3.Display 4.Exit

Enter your choice

2

Removed from Queue : Sushmitha

1. Insert 2. Delete 3.Display 4.Exit

Enter your choice

3

Waiting in Queue now...

Kevin

Karen

1. Insert 2. Delete 3.Display 4.Exit

Enter your choice

2

Removed from Queue : Kevin

1. Insert 2. Delete 3.Display 4.Exit

Enter your choice

2

Removed from Queue : Karen

1. Insert 2. Delete 3.Display 4.Exit

Enter your choice

2