

# Rajalakshmi Engineering College

Name: Kavyasri M  
Email: 240701248@rajalakshmi.edu.in  
Roll no: 240701248  
Phone: 6383586337  
Branch: REC  
Department: CSE - Section 4  
Batch: 2028  
Degree: B.E - CSE

Scan to verify results



## 2024\_28\_III\_OOPS Using Java Lab

### 2028\_REC\_OOPS using Java\_Week 9\_Q2

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Vikram loves listening to music and wants to create a simple playlist manager using Java Collections. The playlist supports the following operations:

"ADD <song>" Adds the song to the end of the playlist. "REMOVE <song>" Removes the first occurrence of the song from the playlist. If the song is not found, do nothing. "SHOW" Displays all songs in the playlist in order. If the playlist is empty, print "EMPTY". "NEXT" Moves to the next song in the playlist and prints its name. If the playlist is empty, print "EMPTY".

The playlist maintains a "current song" position that starts at the first song when it's added. The NEXT command moves to the next song and prints it, wrapping around to the first song after reaching the last song. When removing songs, the current position adjusts accordingly to maintain

proper navigation.

Help Vikram implement this playlist manager.

### ***Input Format***

The first line of the input consists of an integer  $n$ , the number of operations.

The next  $n$  lines, each containing a command:

- "ADD <song>"
- "REMOVE <song>"
- "SHOW"
- "NEXT"

### ***Output Format***

For each "SHOW" command, print the songs in order, separated by spaces.

For each "NEXT" command, print the next song in the playlist.

If no song exists, print "EMPTY".

Refer to the sample output for formatting specifications.

### ***Sample Test Case***

Input: 7

ADD song1

ADD song2

SHOW

NEXT

REMOVE song2

SHOW

NEXT

Output: song1 song2

song2

song1

song1

### ***Answer***

```

import java.util.*;
class PlaylistManager {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        LinkedList<String> playlist = new LinkedList<>();
        String currentSong = null;
        for (int i = 0; i < n; i++) {
            String command = sc.nextLine().trim();

            if (command.startsWith("ADD")) {
                String song = command.split(" ")[1];
                playlist.add(song);
                if (currentSong == null) {
                    currentSong = song;
                }
            }
            else if (command.startsWith("REMOVE")) {
                String song = command.split(" ")[1];
                if (!playlist.isEmpty()) {
                    Iterator<String> it = playlist.iterator();
                    boolean removed = false;
                    while (it.hasNext()) {
                        String s = it.next();
                        if (s.equals(song)) {
                            it.remove();
                            removed = true;
                            break;
                        }
                    }
                    if (removed) {
                        if (playlist.isEmpty()) {
                            currentSong = null;
                        } else if (song.equals(currentSong)) {
                            currentSong = playlist.peekFirst();
                        }
                    }
                }
            }
            else if (command.equals("SHOW")) {
                if (playlist.isEmpty()) {

```

```

        System.out.print("EMPTY ");
    } else {
        for (String s : playlist) {
            System.out.print(s + " ");
        }
    }
}
else if (command.equals("NEXT")) {
    if (playlist.isEmpty()) {
        System.out.print("EMPTY ");
    } else {
        Iterator<String> it = playlist.iterator();
        boolean found = false;
        String first = playlist.peekFirst();

        while (it.hasNext()) {
            String s = it.next();
            if (s.equals(currentSong)) {
                found = true;
                if (it.hasNext()) {
                    currentSong = it.next();
                } else {
                    currentSong = first;
                }
                break;
            }
        }
        if (!found) currentSong = first; // fallback
        System.out.print(currentSong + " ");
    }
}
}
}
sc.close();
}
}

```

**Status :** Correct

**Marks :** 10/10