DATA STRUCTURES
&
ALGORITHMS

DSA group project

NAME	SURNAME	STUDENT NUMBER
Rasheeda Iithete	Iithete	222131330
Kulaumone	Hamuulu	222028696
Sakina	Malumalu	221140794
Denzel	Makururu	222001372
Justianus	Kanyinga	222062754
Jojo	Ntesa	222022701
Tashinga	Manunure	222002905

SECTION A

QUESTION

A Namibian music start-up is looking for an efficient music player algorithm for a mobile application. Suppose you have 100 tracks on your list. The playlist works on the concept of linked list. Tracks are played one by one (best example of singly linked list. Track are connected and you can move from track four to five, but you cannot go back. 1. Implement functionality to play tracks in both directions (following the behavior of double linked list). 2. In addition, add the functionality to allow for the playing of tracks on repeat (circular). 3. Implement the functionality to add and remove tracks from playlist 4. The application should also allow for searches on playlist. E.g., a user could search for a specific track.

Answer

The algorithm that will best satisfy the above problem is as follow.

Start

```
Initialize music player
initialize playlist initialize tracklist
initialize repeat function
initialize search function
initialize add/remove function
play music player
play tracks from playlist
repeat tracks as necessary
search for specific tracks
add/remove tracks as necessary
exit
end
// Now that we done with the initializations we move on to the actual code

Get create_playlist()
Get track
tracks = 100
Get linked list
Get count
count=0

linked_list = create_new_linked_list()
dowhile(count< tracks)
```

```
add_track(linked_list, count)
count++
end do
return linked_list
end function
Get play_track(linked_list, track_number)
Get current_track,head,counter,track_number
counter =1
track_number=100
current_track = linked_list.head
dowhile(counter <=track_number)
current_track = current_track.next
counter++
end do
play_audio(current_track.data)
end function
Get add_track(linked_list, track_number)
Get new_track
new_track = create_new_track(track_number)
if (linked_list.head ==null) then
linked_list.head = new_track
else
current_track = linked_list.head
dowhile(current_track.next != null )
current_track = current_track.next
end do
current_track.next = new_track
end if
end function
Get remove_track(linked_list, track_number)
current_track = linked_list.head
previous_track = null
count1=0
dowhile(count1<=track_number )
previous_track := current_track
current_track := current_track.next
counter1++
    end for
    if (previous_track = null) then
        linked_list.head = current_track.next
    else
        previous_track.next = current_track.next
    end if
end function
Get search_playlist(linked_list, track_number)
    current_track = linked_list.head
    dowhile (current_track != null)
        if( current_track.data = track_number) then
            return true
        end if
        current_track = current_track.next
    end while
    return false
end function
```

```
Stop
```

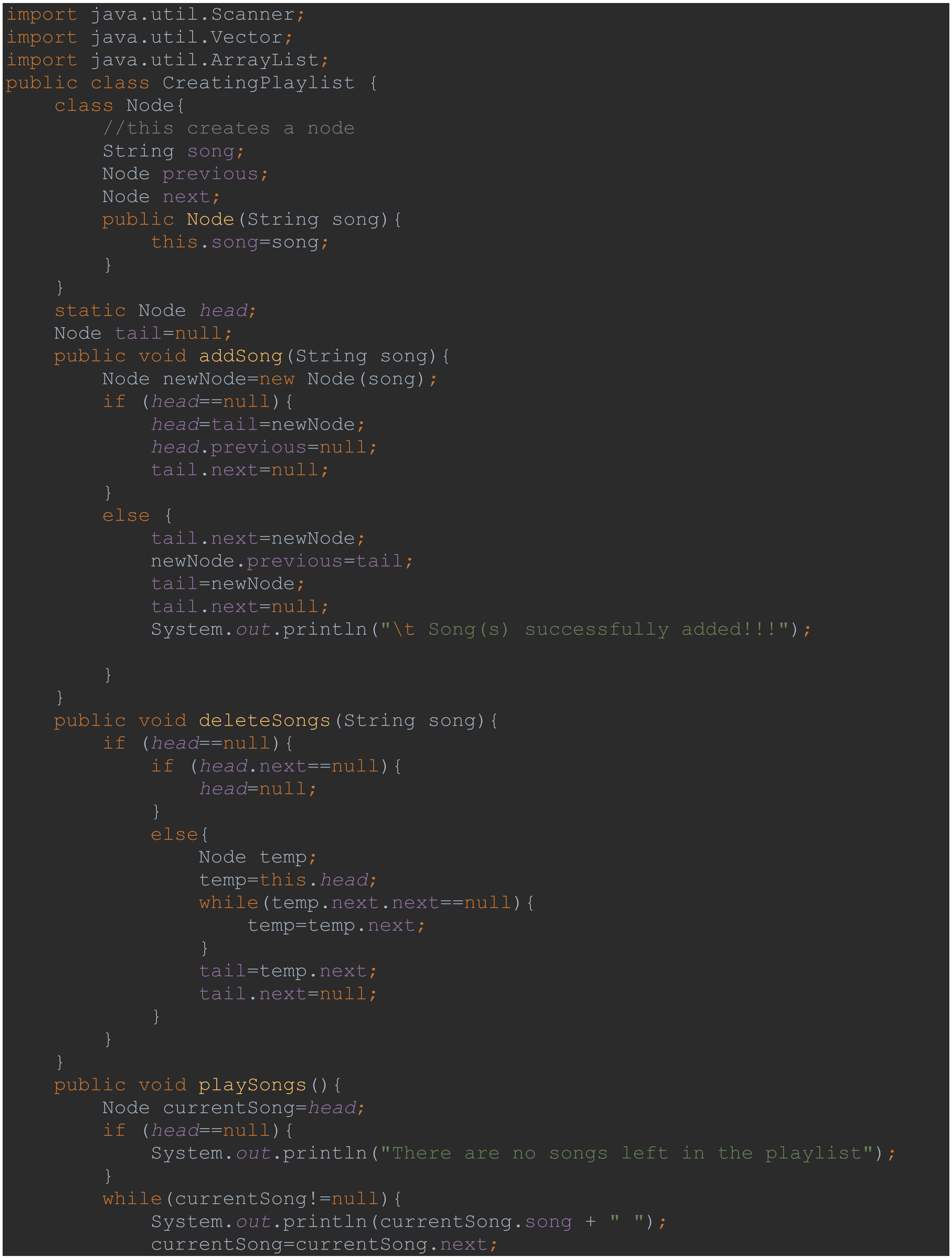
The music player, playlist, tracklist, repeat function, search feature, and add/remove feature are all initialized by the code above, which serves as a quick explanation of the methodology.
The music player then starts playing the songs on the playlist.
You can conduct searches for songs, and the music can be played back as necessary.
The final choice is to add or remove tracks as necessary.

SECTION B

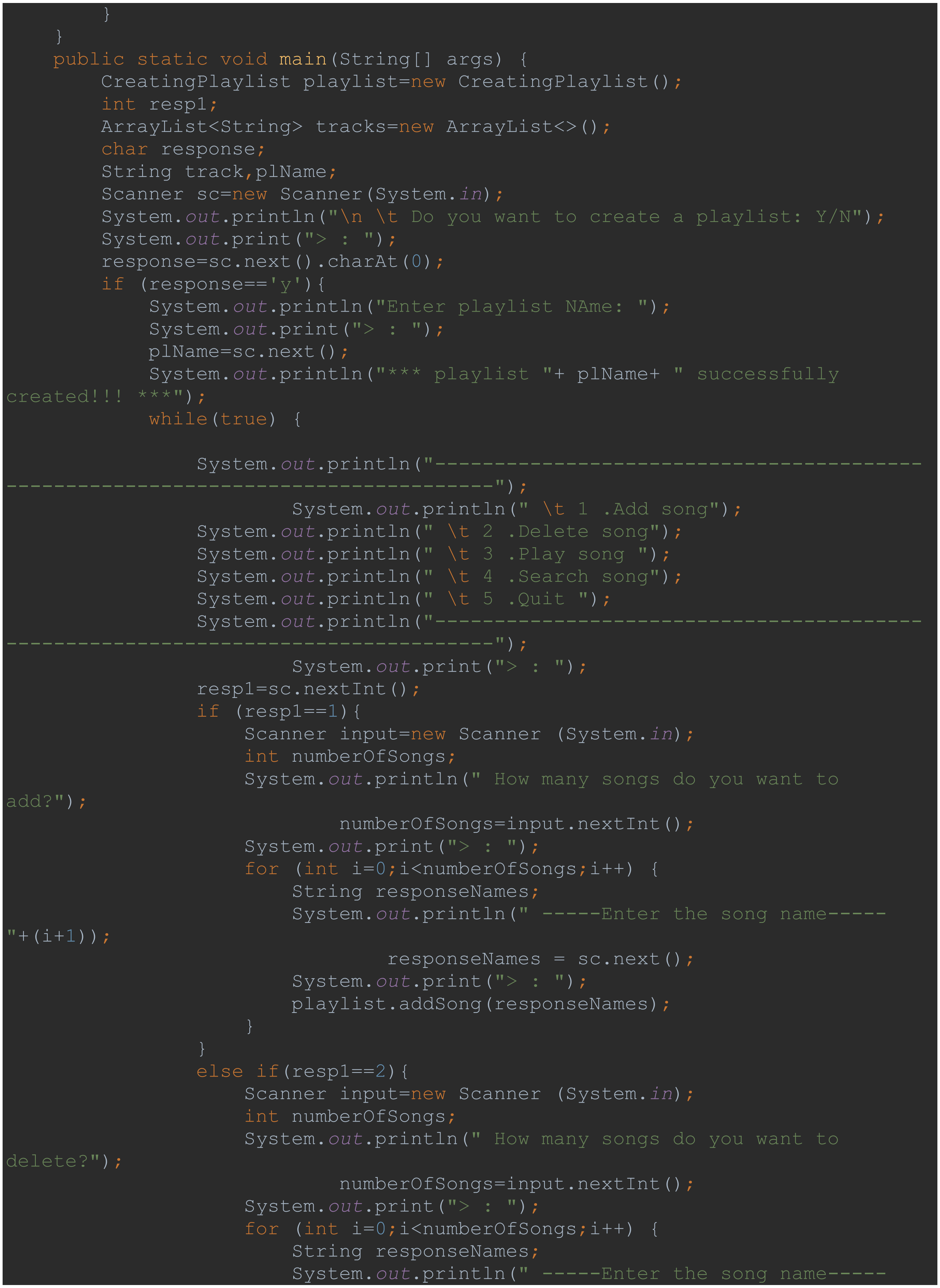
QUESTION
Practical implementation of the program designed in section A

Use ideas from programming to implement your solution algorithm designed in section A. Students are free to use any programming language of their choice to implement the data structure and the operations specified.

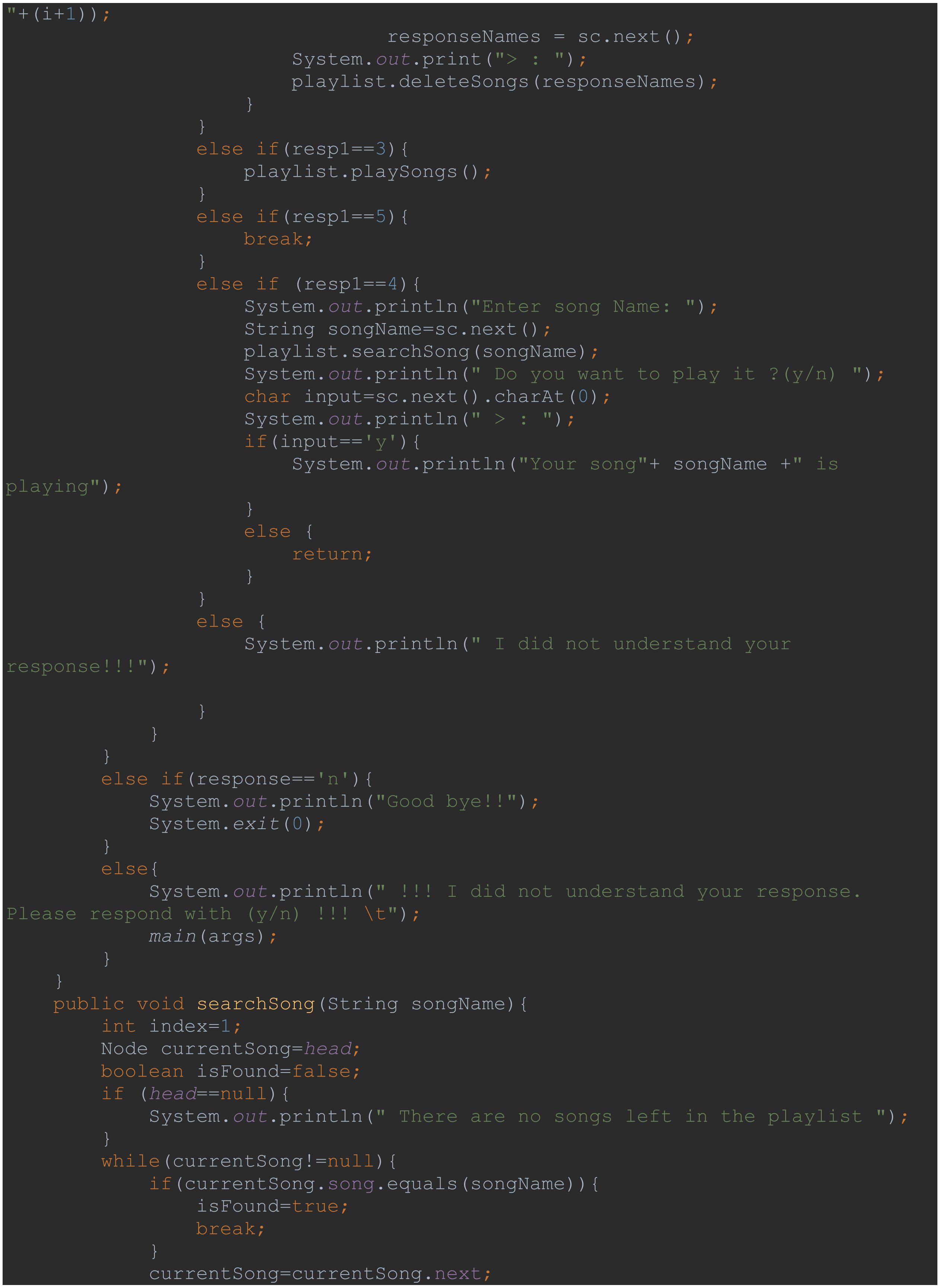
Answer



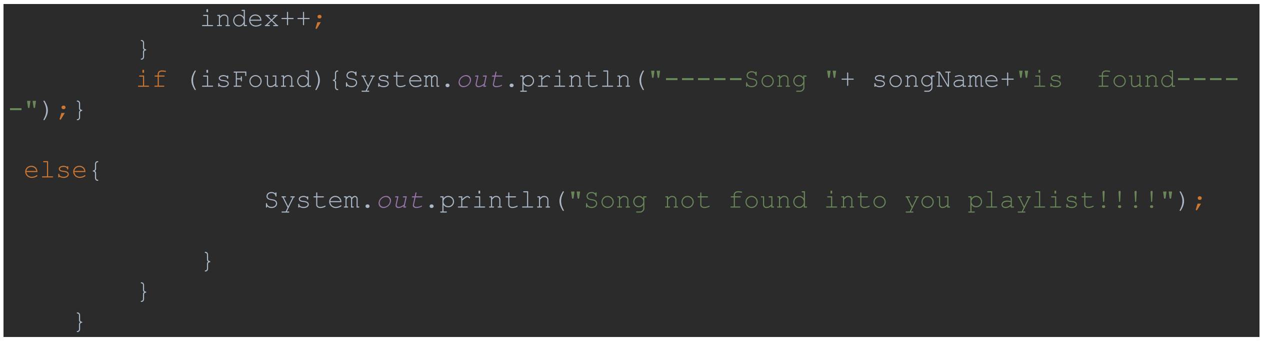
```
import java.util.Scanner;
import java.util.Vector;
import java.util.ArrayList;
public class CreatingPlaylist {
    class Node{
        //this creates a node
        String song;
        Node previous;
        Node next;
        public Node(String song){
            this.song=song;
        }
    }
    static Node head;
    Node tail=null;
    public void addSong(String song){
        Node newNode=new Node(song);
        if (head==null){
            head=tail=newNode;
            head.previous=null;
            tail.next=null;
        }
        else {
            tail.next=newNode;
            newNode.previous=tail;
            tail=newNode;
            tail.next=null;
            System.out.println("\t Song(s) successfully added!!!");

        }
    }
    public void deleteSongs(String song){
        if (head==null){
            if (head.next==null){
                head=null;
            }
            else{
                Node temp;
                temp=this.head;
                while(temp.next.next==null){
                    temp=temp.next;
                }
                tail=temp.next;
                tail.next=null;
            }
        }
    }
    public void playSongs(){
        Node currentSong=head;
        if (head==null){
            System.out.println("There are no songs left in the playlist");
        }
        while(currentSong!=null){
            System.out.println(currentSong.song + " ");
            currentSong=currentSong.next;
```



```
        }
    }
    public static void main(String[] args) {
        CreatingPlaylist playlist=new CreatingPlaylist();
        int resp1;
        ArrayList<String> tracks=new ArrayList<>();
        char response;
        String track,plName;
        Scanner sc=new Scanner(System.in);
        System.out.println("\n \t Do you want to create a playlist: Y/N");
        System.out.print("> : ");
        response=sc.next().charAt(0);
        if (response=='y'){
            System.out.println("Enter playlist NAme: ");
            System.out.print("> : ");
            plName=sc.next();
            System.out.println("*** playlist "+ plName+ " successfully 
created!!! ***");
            while(true) {

                System.out.println("-----------------------------------------
-----------------------------------------");
                        System.out.println(" \t 1 .Add song");
                System.out.println(" \t 2 .Delete song");
                System.out.println(" \t 3 .Play song ");
                System.out.println(" \t 4 .Search song");
                System.out.println(" \t 5 .Quit ");
                System.out.println("-----------------------------------------
-----------------------------------------");
                        System.out.print("> : ");
                resp1=sc.nextInt();
                if (resp1==1){
                    Scanner input=new Scanner (System.in);
                    int numberOfSongs;
                    System.out.println(" How many songs do you want to 
add?");
                            numberOfSongs=input.nextInt();
                    System.out.print("> : ");
                    for (int i=0;i<numberOfSongs;i++) {
                        String responseNames;
                        System.out.println(" -----Enter the song name-----
"+(i+1));
                                responseNames = sc.next();
                        System.out.print("> : ");
                        playlist.addSong(responseNames);
                    }
                }
                else if(resp1==2){
                    Scanner input=new Scanner (System.in);
                    int numberOfSongs;
                    System.out.println(" How many songs do you want to 
delete?");
                            numberOfSongs=input.nextInt();
                    System.out.print("> : ");
                    for (int i=0;i<numberOfSongs;i++) {
                        String responseNames;
                        System.out.println(" -----Enter the song name-----
```



```
"+(i+1));
                                responseNames = sc.next();
                        System.out.print("> : ");
                        playlist.deleteSongs(responseNames);
                    }
                }
                else if(resp1==3){
                    playlist.playSongs();
                }
                else if(resp1==5){
                    break;
                }
                else if (resp1==4){
                    System.out.println("Enter song Name: ");
                    String songName=sc.next();
                    playlist.searchSong(songName);
                    System.out.println(" Do you want to play it ?(y/n) ");
                    char input=sc.next().charAt(0);
                    System.out.println(" > : ");
                    if(input=='y'){
                        System.out.println("Your song"+ songName +" is 
playing");

                    }
                    else {
                        return;
                    }
                }
                else {
                    System.out.println(" I did not understand your 
response!!!");

                }
            }
        }
        else if(response=='n'){
            System.out.println("Good bye!!");
            System.exit(0);
        }
        else{
            System.out.println(" !!! I did not understand your response. 
Please respond with (y/n) !!! \t");
            main(args);
        }
    }
    public void searchSong(String songName){
        int index=1;
        Node currentSong=head;
        boolean isFound=false;
        if (head==null){
            System.out.println(" There are no songs left in the playlist ");
        }
        while(currentSong!=null){
            if(currentSong.song.equals(songName)){
                isFound=true;
                break;
            }
            currentSong=currentSong.next;
```



```
            index++;
        }
        if (isFound){System.out.println("-----Song "+ songName+"is  found----
-");}

 else{
                System.out.println("Song not found into you playlist!!!!");

            }
        }
    }
```