**PROJECT REPORT**

**Name:**  Misbah Uddin Faroque

**Registration No.:** 18701020

**Title:** Music Player

**Course Name:** Data Structures & Algorithms

**Course Code:** CSE-312

**Semester:** 3rd

**INDEX**

1. Problem Statement
2. Software Used
3. Functions / Modules
4. List of Errors Encountered while coding the project
5. Key or challenging logic in the project
6. Prerequisites
7. Project Code
8. Elaborate Sample Input and Output Screenshots

**PROBLEM STATEMENT**

To make a basic music player using the knowledge of various Data structures learnt as part of course CSE2003

Data structures used include a doubly linked list, stacks and queues.

File handling to read and write songs has also been used.

**SOFTWARE REQUIREMENT**

CODE BLOCKS 13.12

or

DEV C++ 5.11

**FUNCTIONS**

1. tofile() –

Function to work on playslist.txt.

2. add\_node() –

Function that adds songs to the list using a linked list.

3. add\_node\_file() –

Function that adds songs to the playlist to linked list from the data passed in addplaylist() function.

4. delete\_file() –

Function to delete song from text file playlist.txt.

5. del\_node() –

Function that deletes the last song from the input linked list.

6. printlist() –

Function that displays the input songs of the playlist.

7. count\_nodes() –

Function that tracks the number of inputs in the linked list.

8. del\_pos() –

Function that deletes songs from the linked list using the position of that song.

9. search1() –

Function that takes song input and linearly searches through the linked list and finds matching case.

10. push() –

Function that pushes the last played track of play() function into a stack to store and create a recently played list.

11. display() -

Function to display the stack generated in push() function.

12. play() -

Function to search input song and show if it can be played. It then passes the song to push() function to be added to recently played list.

13. recent() -

Function that calls display() function.

14. topelement() –

Function that displays the last played song.

15. addplaylist() –

Function that opens text file playlist.txt and passes data to add\_node\_file() function.

16. del\_search() –

Function to search input song and delete it from the list.

17. deletemenu() –

Function to invoke del\_search() or del\_pos() functions depending on user input.

18. main() –

Function that invokes all other functions of the project based on user defined input.

**PROJECT CODE**

#include<iostream>

#include<string.h>

#include<stdlib.h>

#include<stdio.h>

#include<fstream>

using namespace std;

struct node

{

char song[100];

struct node \*next;

struct node \*prev;

}\*top,\*temp,\*top1;

void tofile(char a[])

{

fstream f1;

f1.open("playlist.txt",ios::out|ios::app);

f1<<a<<endl;

f1.close();

}

void add\_node(struct node \*first)

{

char a[100];

while(first->next!=NULL)

{

first=first->next;

}

first->next=(struct node\*)malloc(sizeof(struct node));

first->prev=first;

first=first->next;

cout<<"\n\a\a\a\aEnter Song name- ";

scanf("%s",&a);

strcpy(first->song,a);

tofile(a);

first->next=NULL;

}

void add\_node\_file(struct node \*first,string a)

{

while(first->next!=NULL)

{

first=first->next;

}

first->next=(struct node\*)malloc(sizeof(struct node));

first->prev=first;

first=first->next;

strcpy(first->song,a.c\_str());

first->next=NULL;

}

void delete\_file(char a[])

{

fstream f1,f2;

string line;

int x=0;

f1.open("playlist.txt",ios::in|ios::out);

f2.open("temp.txt",ios::in|ios::out);

while(!f1.eof())

{

getline(f1,line);

if(strcmp(a,line.c\_str())!=0)

f2<<line<<endl;

else if (strcmp(a,line.c\_str())==0)

x=1;

}

f1.close();

f2.close();

remove("playlist.txt");

rename("temp.txt","playlist.txt");

if(x==0)

{

cout << "#Song not found." << endl;

}

else

{

cout << "=>Song has been deleted." << endl;

}

}

void del\_node(struct node \*first)

{

while((first->next)->next!=NULL)

{

first=first->next;

}

struct node \*temp;

temp=(first->next)->next;

first->next=NULL;

free(temp);

cout<<"Deleted"<<endl;

}

void printlist(struct node \*first)

{

cout<<"\nPlaylist Name- ";

while(first->next!=NULL)

{

cout<<first->song<<endl;

first=first->next;

}

cout<<first->song<<endl;

}

void count\_nodes(struct node \*first)

{

int i=0;

while (first->next!=NULL)

{

first=first->next;

i++;

}

i++;

cout<<"\nTotal songs- "<<i-1<<endl;

}

struct node \*del\_pos(struct node \*pointer, int pos)

{

struct node \*n1,\*prev1,\*temp;

prev1= ( struct node \*)malloc(sizeof(node));

temp= (struct node \*)malloc(sizeof(node));

int i=0;

if(pos==1)

{

temp=pointer;

delete\_file(temp->song);

pointer=pointer->next;

pointer->prev = NULL;

free(temp);

printf("\n=>The list has been updated\n++Use the display function to check\n");

return pointer;

}

while(i<pos-1)

{

prev1=pointer;

pointer=pointer->next;

i++;

}

if(pointer->next==NULL)

{

temp=pointer;

delete\_file(temp->song);

prev1->next->prev=NULL;

prev1->next=NULL;

free(temp);

printf("\n=>The list has been updated\n++Use the display function to check\n");

}

else

{

temp=pointer;

delete\_file(temp->song);

prev1->next=temp->next;

temp->next->prev=prev1;

free(temp);

printf("\n=>The list has been updated\n++Use the display function to check\n");

}

}

void search1(struct node \*first)

{

char song[100];

cout<<"\n\a\a\a\aEnter song To be Searched- ";

scanf("%s",&song);

int flag=0;

while(first!=NULL)

{

if(strcmp(first->song,song)==0)

{

cout<<"\n\a\a\a\a#Song Found"<<endl;

flag++;

break;

}

else

{

first=first->next;

}

}

if(flag==0)

{

cout<<"\n\a\a\a\a#Song Not found"<<endl;

}

}

void create()

{

top = NULL;

}

void push(char data[])

{

if (top == NULL)

{

top =(struct node \*)malloc(sizeof(struct node));

top->next = NULL;

strcpy(top->song,data);

}

else if (strcmp(top->song,data)!=0)

{

temp =(struct node \*)malloc(sizeof(struct node));

temp->next = top;

strcpy(temp->song,data);

top = temp;

}

}

void display()

{

top1 = top;

if (top1 == NULL)

{

printf("\n\a\a\a\a=>NO recently played tracks.\n");

return;

}

printf("\n\a\a\a\a#Recently played tracks-\n");

while (top1 != NULL)

{

printf("%s", top1->song);

printf("\n");

top1 = top1->next;

}

}

void play(struct node \*first)

{

char song[100];

printlist(first);

cout<<"\n\a\a\a\aChoose song you wish to play- ";

scanf("%s",song);

int flag=0;

while(first!=NULL)

{

if(strcmp(first->song,song)==0)

{

cout<<"\n\a\a\a\a=>Now Playing......"<<song<<endl;

flag++;

push (song);

break;

}

else

{

first=first->next;

}

}

if(flag==0)

{

cout<<"\n\a\a\a\a#Song Not found"<<endl;

}

}

void recent()

{

display();

}

void topelement()

{

top1=top;

if(top1==NULL)

{

printf("\n\a\a\a\a#NO last played tracks.\n");

return;

}

cout<<"\n=>Last Played Song - "<<top->song<<endl;

}

void addplaylist(struct node \*start)

{

fstream f1;

string line;

f1.open("playlist.txt",ios::in);

while(!f1.eof())

{

getline(f1,line);

add\_node\_file(start,line);

}

cout<<"=>Playlist Added"<<endl;

f1.close();

}

void del\_search(struct node \*start)

{

char song[100];

printlist(start);

cout<<"\n\a\a\a\aChoose song you wish to delete- ";

scanf("%s",song);

int flag=0;

while(start!=NULL)

{

if(strcmp(start->song,song)==0)

{

cout<<"\n\a\a\a\a#Song Found"<<endl;

struct node \*temp;

temp= (struct node \*)malloc(sizeof(node));

temp=start;

delete\_file(temp->song);

temp->prev->next=temp->next;

temp->next->prev=temp->prev;

free(temp);

flag++;

break;

}

else

{

start=start->next;

}

}

if(flag==0)

{

cout<<"\n\a\a\a\a#Song Not found"<<endl;

}

}

void deletemenu(struct node \*start)

{

int c;

cout<<"Delete song?\n1.By Search\n2.By Position"<<endl;

cout<<"\a\a\a\aEnter your choice-";

cin>>c;

switch(c)

{

case 1:del\_search(start);

break;

case 2:int pos;

printf("\n\a\a\a\aEnter the song position : ");

scanf("%d",&pos);

del\_pos(start,pos-1);

break;

}

}

main()

{

int choice,loc;

char song[100];

struct node \*start,\*hold;

start=(struct node \*) malloc(sizeof(struct node));

cout<<"\t\t\t\a\a\a\a\*\*WELCOME\*\*"<<endl;

cout<<"\n\*\*please use '\_' for space."<<endl;

cout<<"\n\n\a\a\a\aEnter your playlist name- ";

cin.getline(start->song,100);

start->next=NULL;

hold=start;

create();

do{

cout<<"\n1.Add New Song\n2.Delete Song\n3.Display Entered Playlist\n4.Total Songs\n5.Search Song\n6.Play Song\n7.Recently Played List\n8.Last Played\n9. Add From File\n10.Exit"<<endl;

cout<<("\n\a\a\a\aEnter your choice- ");

cin>>choice;

switch(choice)

{

case 1:add\_node(start);

break;

case 2:deletemenu(start);

break;

case 3:printlist(start);

break;

case 4:count\_nodes(hold);

break;

case 5:search1(start);

break;

case 6:play(start);

break;

case 7:recent();

break;

case 8:topelement();

break;

case 9:addplaylist(start);

break;

case 10:exit(0);

}

}while(choice!=11);

}

**SAMPLE INPUT AND OUTPUT**

