|  |
| --- |
|  |
| World Talent Show |
| A C++ Program For Song And Dance Competition |
|  |
|  |
|  |

|  |
| --- |
|  |

**ACKNOWLEDGEMENT**

The acknowledgement page is the most beautiful page of any book.

I, Mohamed Hafeez of class XII, find words quite inadequate to express my gratitude to a lot of people. First and foremost, I would like to thank the Almighty. Then, I would like to thank my inspirational Computer Science teacher, Mrs Lovely Binesh, campus head Mrs Serita Christina and the beloved Principal of my school, Mrs J. Jaibala for their constant encouragement, abundant guidance and help throughout the practical sessions.

I would like to once again specially thank my Computer Science teacher for making my practical sessions both enjoyable and informative that provoked inquisitiveness and built upon my skill as a programmer throughout the academic year 2016-17.

I would like to thank my friends who took interest in rectifying the errors and supported me through the course of this project.Above all, I thank my family members for all the assistance at home that helped me complete this project in time.

I would also like to take this opportunity to thank all those people who have directly or indirectly aided me in making of this project.

Thank you very much.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **SERIAL** | **TOPIC** | **PAGE** |
| 1 | Introduction | 1 |
| 2 | Why This Project | 3 |
| 3 | Hardware And Software Requirements | 4 |
| 4 | Classes And Functions | 5 |
| 5 | Algorithm | 10 |
| 6 | Source Code | 15 |
| 7 | Screen Shots | 27 |
| 8 | Drawbacks | 31 |
| 9 | Bibliography | 32 |

**INTRODUCTION**

The aim of this project is to create an interactive C++ program to act as application software for singing and dancing competitions that could maintain participant details and update the same in accordance to the fact that a contestant is selected or not.

# Addition Of Contestants

There is an option in the main menu to add new contestants. This function opens the corresponding file, singing or dancing, by checking the value of ‘status’. It then checks the count of each and accordingly, opens the file in ‘out’ mode or ‘app’ mode.

# Dance Menu

This option takes the user to another menu with options to choose a contestant which randomly chooses the next contestant to perform using the random( ) function with parameter as the count of dancers. It consists of perform option which leads to options to “select” the current contestant, “not select” them and put them in wait list by assigning their status as ‘S’, ‘N’ and ‘W’ respectively in the file.

# Singing Menu

This option consists of every option that dance menu had, difference being the file in which it stores records.

# Wait List

While a contestant is being declared wait-listed, they are added to a dynamic linked for faster access if in case, they were to be called upon to give a second chance.

**ADVANTAGES**

* **Data File Handling**

**Implementation of data file handling makes this program reliable and quick. It also ensures that there is no data loss during any phase of execution of the program.**

* **Classes**

Use of class to describe each object prevents unintentional changes by other objects and puts into effect data hiding and data abstraction.

* **Memory**

**This C++ program is of very less size, in order of few kilobytes, and thus consumes very less memory.**

* **Speed**

C++ code translates very easily and effectively into fast assembly code, and from there into machine code.

* **Portability**

C++ is a portable language, supporting multiple platforms for both running and programming.

* **User-Friendly**

This program is designed to be user-friendly with menus at various levels to guide the user to the aimed destination.

**WHY THIS PROJECT**

This is the modern age where commercialization is ruling the world. For every bend and turn, we come across a singing or a dancing competition that promises homes and a large sum of money as winning. Almost every channel owns a singing or a dancing show or sometimes both together terming it a ‘talent search’. These shows rely almost entirely on paper trail that consists of instructions and winners list penned down by the event managers and their assistants.

This system is often unreliable and inefficient. There is quite a lot of room for human errors and is subject to emotional bias. There is that possibility of missing those planner sheets. Writing down or printing them consumes quite a lot paper that adds on to the existing environmental problems besides being cost-inefficient. Overall, it is a slow system as well.

Computerization is the way to go. A computer program makes the whole system fast, reliable and effective. It is cheap yet efficient. It also paves way to put into effect measures of security to preserve the integrity of the information and prevent unauthorized alterations to the information pertaining to the whole event.

Thus this project was taken up to help at the time of such huge-hectic events and to make the whole process of event management less stressful and aid in smooth progression of the show.

**HARDWARE AND SOFTWARE USED IN DEVELOPMENT PHASE**

# Turbo C++ : 3.1 (or higher)

# Operating System: Windows 7 Professional

# System Type : 64-bit

# Processor : 1.80 GHz

# Hard Disk : 465 GB

# Memory (RAM) : 1 GB

# Monitor : 1366x768 pixels

# Keyboard : Standard PS/2

**CLASSES AND FUNCTIONS**

**Header Files Used**

<fstream.h>

* File.read( )
* File.write( )

<conio.h>

* clrscr( )
* getch( )

<process.h>

* exit( )

<dos.h>

* delay( )

<stdio.h>

* gets( )

<stdlib.h>

* random( )
* randomize( )

<string.h>

* strcpy( )
* strcmp( )

Class ‘contestant’

|  |
| --- |
| **contestant** |
| Data Members:  intcontno  char name[30]  intage  char cat  char status  Member Functions:  void add( )  void display( )  void addstat(char)  int returnno( ) |

|  |
| --- |
| **waiting** |
| Data Members:  intcontno  char cate  waiting \*next |

Structure ‘waiting’

Members Of Class ‘contestant’

* contno

This integer variable stores the contestant number.

* name

This string variable stores the name of the contestant.

* age

This integer variable stores the age of the contestant.

* cat

This character variable stores the value ‘D’ or ‘S’ each standing for dancing and singing respectively.

* status

This character variable stores the value ‘S’ for selected, ‘N’ for not selected and ‘W’ for wait list.

* void add( )

This function is called to take in the participant details such as their contestant number, name, age, category.

* void display( )

This function is called to display a candidate’s detail such as their contestant number, name, age, category along with their status, whether or not are they selected.

* void addstat(char)

This function is called to assign the value for ‘status’ as ‘S’, ‘N’ or ‘W’.

* int returnno( )

This function returns the value of ‘contno’ to be compared against that candidate number who is to be assigned the status ‘S’,’N’ or ‘W’.

Members Of Structure ‘waiting’

* contno

This integer variable stores the contestant number of the candidate who is on the waiting list.

* cate

This character variable stores the contestant category who is on the waiting list.

* next

This is self-referential pointer points to the next candidate in the queue.

User-defined Functions

* void menu();

This is the first function to be called from main( ) function. It is responsible for displaying the list of actions such as dance menu, singing menu, view waitlist, view contestant details and add more contestants.

* void dance();

This function is the first option in the main menu. This displays a list of options to choose a candidate randomly, performance menu to make decisions such as select, not select or put in waiting list.

* void sing();

This function, the next option in the menu, displays the same set of operations as dance ( ).

* void contest();

This function is called to view the details of all the candidates in the file.

* void viewwait();

This function is used to view the linearly linked queue of candidates who are in waiting list.

* void add();

This function is used to add a contestant detail to the file.

* void randcont(int&);

This function randomly generates a contestant number to decide on which candidate is to perform next.

* void perform();

Before a performance begins, this function is called which displays the options to select, not select or put a candidate in the waiting list.

* void selects();

The purpose of this function is to access a contestant’s record from file and change their status to “selected” or “s” for singing.

* void selectd();

The purpose of this function is to access a contestant’s record from file and change their status to “selected” or “s” for dancing.

* void wait(int&);

This function is called when a candidate is put in the waiting list. This function maintains a dynamic list of all waiting list candidates.

* void notsels();

The purpose of this function is to access a contestant’s record from file and change their status to “not selected” or “n” for singing.

* void notseld();

The purpose of this function is to access a contestant’s record from file and change their status to “not selected” or “n” for dancing.

* void count();

This function counts

**ALGORITHM**

1. Start process
2. Define a class ‘contestant’ with ‘contno’ of type integer, ‘name’ of type string, ‘age’ of type integer, ‘cat’ of type character and ‘status’ of type character as its data members and ‘add( )’, ‘display( )’, ‘addstat(char)’ and returnno( ) as its member functions.Declare add() function as its friend. Also declare a global object ‘C1’ of its type.
3. Define a self-referential structure ‘waiting’ with ‘contno’ of type integer and ‘next’ as a self-referential pointer with ‘front’ and ‘rear’ pointers as its global objects.
4. Declare global integer variables ‘c’, ‘counts’,’countd’ with value ‘0’ each and ‘y’ with value 3 along with a global file object ‘f’ of type fstream.
5. Declare a string variable ‘pass’ and accept the password character by character using getch().
6. Set the number of records in counts and countd by reading through files contests.dat and contestd.dat and display the menu of actions that can be taken by the user.
7. If the user’s choice is dance menu, invoke dance() function.

* Display the list of actions that can be taken by the user.
* If the user’s choice is to choose a contestant, invoke randcont() function with parameter ‘1’.
* Generate a random contestant number from 1 to number of contestants in dancing, assign it to ‘c’ and display it.
* Else if the user’s choice is to select or not select a contestant, invoke perform() function with parameter ‘1’.
* Display the list of actions that can be taken by the user.
* If the user’s choice is to declare a contestant selected, invoke selectd() function.
* Open the file “contestd.dat” in in mode using file object f and open the file “temp.dat” in out mode using the file object f1.
* While(f.read((char\*)&C1,sizeof(C1))) check if C1.returnno()==c and invoke C1.addstat() function with parameter ‘S’.
* Assign status as ‘S’.
* Write C1 object into the file in f1.
* Close the files f1 and f.
* Remove “contestd.dat” and rename “temp.dat” as “contestd.dat”.
* Invoke menu().
* Else if the user’s choice is to wait list a contestant, invoke wait() function with parameter ‘1’.
* Open the file “contestd.dat” in in mode and open the file “temp.dat” in out mode using the file objects f and f1.
* While(f.read((char\*)&C1,sizeof(C1))) check if C1.returnno()==c and invoke C1.addstat() function with parameter ‘W’.
* Assign status as ‘W’.
* Write C1 object into the file in f1.
* Close the files f1 and f.
* Remove “contestd.dat” and rename “temp.dat” as “contestd.dat”.
* Declare a dynamic object ‘temp’ of type waiting and assign value ‘D’ to temp->cate and temp->contno=c.
* Add the temporary file to the linked queue as rear->next=temp and rear=temp.
* Invoke menu().
* Else if the user’s choice is to declare a contestant not selected, invoke notseld() function.
* Open the file “contestd.dat” in in mode using file object f and open the file “temp.dat” in out mode using the file object f1.
* While(f.read((char\*)&C1,sizeof(C1))) check if C1.returnno()==c and invoke C1.addstat() function with parameter ‘N’.
* Assign status as ‘N’.
* Write C1 object into the file in f1.
* Close the files f1 and f.
* Remove “contestd.dat” and rename “temp.dat” as “contestd.dat”.
* Invoke menu().
* Else if the user’s choice is to go back to main menu, invoke menu().
* Else if the user’s choice is to exit the program, invoke exit(0) function.
* Else if the user’s choice is to switch to singing menu, invoke sing() function.
* Display the list of actions that can be taken by the user.
* If the user’s choice is to choose a contestant, invoke randcont() function with parameter ‘0’.
* Generate a random contestant number from 1 to number of contestants in dancing, assign it to ‘c’ and display it.
* Else if the user’s choice is to select or not select a contestant, invoke perform() function with parameter ‘0’.
* Display the list of actions that can be taken by the user.
* If the user’s choice is to declare a contestant selected, invoke selects() function.
* Open the file “contests.dat” in in mode using file object f and open the file “temp.dat” in out mode using the file object f1.
* While(f.read((char\*)&C1,sizeof(C1))) check if C1.returnno()==c and invoke C1.addstat() function with parameter ‘S’.
* Assign status as ‘S’.
* Write C1 object into the file in f1.
* Close the files f1 and f.
* Remove “contests.dat” and rename “temp.dat” as “contests.dat”.
* Invoke menu().
* Else if the user’s choice is to wait list a contestant, invoke wait() function with parameter ‘0’.
* Open the file “contests.dat” in in mode and open the file “temp.dat” in out mode using the file objects f and f1.
* While(f.read((char\*)&C1,sizeof(C1))) check if C1.returnno()==c and invoke C1.addstat() function with parameter ‘W’.
* Assign status as ‘W’.
* Write C1 object into the file in f1.
* Close the files f1 and f.
* Remove “contests.dat” and rename “temp.dat” as “contests.dat”.
* Declare a dynamic object ‘temp’ of type waiting and assign value ‘D’ to temp->cate and temp->contno=c.
* Add the temporary file to the linked queue as rear->next=temp and rear=temp.
* Invoke menu().
* Else if the user’s choice is to declare a contestant not selected, invoke notsels() function.
* Open the file “contests.dat” in in mode using file object f and open the file “temp.dat” in out mode using the file object f1.
* While(f.read((char\*)&C1,sizeof(C1))) check if C1.returnno()==c and invoke C1.addstat() function with parameter ‘N’.
* Assign status as ‘N’.
* Write C1 object into the file in f1.
* Close the files f1 and f.
* Remove “contests.dat” and rename “temp.dat” as “contests.dat”.
* Invoke menu().
* Else if the user’s choice is to go back to dance menu, invoke dance() function.
* Else if the user’s choice is to go back to main menu, invoke menu().
* Else if the user’s choice is to exit the program, invoke exit(0) function.
* Else if the user’s choice is to go back to main menu, invoke menu().
* Else if the user’s choice is to exit the program, invoke exit(0) function.

1. Else if the user’s choice is singing menu, invoke sing().
2. Else if the user’s choice is to view all the candidate details, invoke contest() function.

* Open the file “contests.dat” in in mode using file object f.
* While(f.read((char\*)&C1,sizeof(C1))) invoke C1.display().
* Display contestant number, name, age, category and status in the competition.
* Close the file f.
* Open the file “contestd.dat” in in mode using file object f.
* While(f.read((char\*)&C1,sizeof(C1))) invoke C1.display().
* Close the file f.

1. Else if the user’s choice is to view the wait list, invoke viewwait() function.

* Create a dynamic object tempo of type waiting and assign front to it.
* While(tempo!=null) display tempo->contno and temp->cate. Assign temp=temp->next.

1. Else if the user’s choice is to add new candidates, invoke add() function.

* Invoke C1.add().
* if(C1.cat==’S’) check if counts==0, open “contests.dat” in out mode using file object f and write the object C1 into file. Otherwise open the file in app mode and do the same.
* Else check if countd==0, open “contestd.dat” in out mode using file object f and write the object into file. Otherwise open the file in app mode and do the same.

1. Else if the user’s choice is to exit the program, invoke exit(0) function.
2. End process.

**SOURCE CODE**

//A C++ Program For Singing/Dance Competition

#include<fstream.h>

#include<conio.h>

#include<process.h>

#include<string.h>

#include<stdlib.h>

#include<dos.h>

#include<stdio.h>

/\*Function Prototypes\*/

void add(); //To add a contestant to the file

void selects(); //To set status as ’S’ for a singing contestant

voidselectd(); //To set status as ‘S’ for a dancing contestant

void wait(int&); //To add a candidate to waiting list

voidnotsels(); //To set status as ‘N’ for a singing contestant

voidnotseld(); //To set status as ‘N’ for a dancing contestant

void menu(); //Displays a list of action that can be taken by the user

void dance(); //Displays a list of operations for a dancing contestant

void sing(); //Displays a list of operations for a singing contestant

voidrandcont(int&); /\*To randomly generate a contestant number who’d perform next\*/

void contest(); //To view the details of all participants in file

void perform(int); /\*To display a list of operations to be done after a performance\*/

voidviewwait(); //Displays the dynamic wait list

void count(); //To set the number of records in counts and countd

intc,counts=0,countd=0,y=3; //Global Objects

fstream f;

struct waiting

{

intcontno;

charcate;

waiting \*next;

}\*front,\*rear;

class contestant

{

intcontno;

char name[30];

int age;

char cat;

char status;

public:

contestant()

{

contno=0;

strcpy(name,"NULL");

age=0;

cat='0';

status='0';

}

~contestant()

{}

void add();

void display();

voidaddstat(char);

int returnno();

friend void add();

}C1;

int contestant::returnno()

{

returncontno;

}

void contestant::add()

{

clrscr();

gotoxy(27,1);

cout<<"World Talent Show 2017!";

cout<<"\n\tEnter The Contestant Number : ";

cin>>contno;

cout<<"\n\tEnter The Name : ";

gets(name);

cout<<"\n\tEnter The Age : ";

cin>>age;

cout<<"\n\tEnter The Category(Dance-D/Sing-S): ";

cin>>cat;

}

void contestant::display()

{

clrscr();

gotoxy(27,1);

cout<<"World Talent Show 2017!";

cout<<"\n\tContestant Number : "<<contno;

cout<<"\n\tName : "<<name;

cout<<"\n\tAge : "<<age;

cout<<"\n\tCategory : "<<cat;

cout<<"\n\tStatus : "<<status;

}

void contestant::addstat(char a)

{

status=a;

}

void main()

{

clrscr();

char pass[7],s;

front=NULL;

rear=NULL;

count();

gotoxy(27,1);

cout<<"World Talent Show 2017!";

cout<<"\n\n\n\t\tEnter The Password : ";

pass[6]='\0';

for(int i=0;i<6;i++)

{

pass[i]=getch();

}

if(strcmp(pass,"pG9677")==0)

menu();

else

{

cout<<"\n\n\t\tAuthentication Error!";

getch();

exit(0);

}

}

void menu()

{

int o;

do

{

clrscr();

gotoxy(27,1);

cout<<"World Talent Show 2017!";

cout<<"\n\n\t1.Dance";

cout<<"\n\n\t2.Singing";

cout<<"\n\n\t3.Contestants Area";

cout<<"\n\n\t4.View Waiting List";

cout<<"\n\n\t5.Add Candidates";

cout<<"\n\n\t6.Exit";

cout<<"\n\tChoose Your Option : ";

cin>>o;

switch(o)

{

case 1:dance();

break;

case 2:sing();

break;

case 3:contest();

break;

case 4:viewwait();

break;

case 5:add();

break;

case 6:exit(0);

default:cout<<"\n\tEnter A Valid Choice!";

}

delay(1000);

}while(o!=6);

}

void count()

{

f.open("contests.dat",ios::in|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

counts++;

f.close();

f.open("contestd.dat",ios::in|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

countd++;

f.close();

}

void add()

{

C1.add();

if(C1.cat=='S')

{

if(counts==0)

{

f.open("contests.dat",ios::out|ios::binary);

f.write((char\*)&C1,sizeof(C1));

counts++;

f.close();

}

else

{

f.open("contests.dat",ios::app|ios::binary);

f.write((char\*)&C1,sizeof(C1));

counts++;

f.close();

}

}

else

{

if(countd==0)

{

f.open("contestd.dat",ios::out|ios::binary);

f.write((char\*)&C1,sizeof(C1));

countd++;

f.close();

}

else

{

f.open("contestd.dat",ios::app|ios::binary);

f.write((char\*)&C1,sizeof(C1));

countd++;

f.close();

}

}

}

void dance()

{

int p;

do

{

clrscr();

gotoxy(27,1);

cout<<"World Talent Show 2017!";

cout<<"\n\n\t1.Choose A Contestant";

cout<<"\n\n\t2.Perform";

cout<<"\n\n\t3.Singing Menu";

cout<<"\n\n\t4.Main Menu";

cout<<"\n\n\t5.Exit";

cout<<"\n\n\tChoose Your Option : ";

cin>>p;

switch(p)

{

case 1:randcont(1);

break;

case 2:perform(1);

break;

case 3:sing();

break;

case 4:menu();

break;

case 5:exit(0);

default:cout<<"\n\tEnter A Valid Choice!";

}

delay(1000);

}while(p!=5);

}

void sing()

{

int q;

do

{

clrscr();

gotoxy(27,1);

cout<<"World Talent Show 2017!";

cout<<"\n\n\t1.Choose A Contestant";

cout<<"\n\n\t2.Perform";

cout<<"\n\n\t3.Dance Menu";

cout<<"\n\n\t4.Main Menu";

cout<<"\n\n\t5.Exit";

cout<<"\n\n\tChoose Your Option : ";

cin>>q;

switch(q)

{

case 1:randcont(0);

break;

case 2:perform(0);

break;

case 3:dance();

break;

case 4:menu();

break;

case 5:exit(0);

default:cout<<"\n\tEnter A Valid Choice!";

}

delay(1000);

}while(q!=5);

}

void contest()

{

f.open("contests.dat",ios::in|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

{

C1.display();

getch();

}

f.close();

f.open("contestd.dat",ios::in|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

{

C1.display();

getch();

}

f.close();

}

voidrandcont(int&a)

{

clrscr();

randomize();

if(a==0)

{

c=(random(counts)+1);

gotoxy(27,1);

cout<<"World Talent Show 2017!";

cout<<"\n\n\t\tLucky Contestant Is : "<<c;

}

else

{

c=(random(countd)+1);

gotoxy(27,1);

cout<<"World Talent Show 2017!";

cout<<"\n\n\t\tLucky Contestant Is : "<<c;

}

}

void perform(int x)

{

int r;

do

{

clrscr();

gotoxy(27,1);

cout<<"World Talent Show 2017!";

cout<<"\n\n\t1.Selected";

cout<<"\n\n\t2.Wait List";

cout<<"\n\n\t3.Not Selected";

cout<<"\n\n\t4.Main Menu";

cout<<"\n\n\t5.Exit";

cout<<"\n\n\tChoose Your Option : ";

cin>>r;

delay(3000);

switch(r)

{

case 1:if(x==0)

selects();

else

selectd();

break;

case 2:wait(x);

break;

case 3:if(x==0)

notsels();

else

notseld();

break;

case 4:menu();

break;

case 5:exit(0);

default:cout<<"\n\tEnter A Valid Choice!";

}

delay(1000);

}while(r!=5);

}

void selects()

{

ofstream f1;

f.open("contests.dat",ios::in|ios::binary);

f1.open("temp.dat",ios::out|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

{

if(C1.returnno()==c)

C1.addstat('S');

f1.write((char\*)&C1,sizeof(C1));

}

f1.close();

f.close();

remove("contests.dat");

rename("temp.dat","contests.dat");

clrscr();

gotoxy(34,12);

cout<<"Selected! :)";

getch();

menu();

}

voidselectd()

{

ofstream f1;

f.open("contestd.dat",ios::in|ios::binary);

f1.open("temp.dat",ios::out|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

{

if(C1.returnno()==c)

C1.addstat('S');

f1.write((char\*)&C1,sizeof(C1));

}

f1.close();

f.close();

remove("contestd.dat");

rename("temp.dat","contestd.dat");

clrscr();

gotoxy(34,12);

cout<<"Selected! :)";

getch();

menu();

}

void wait(int&b)

{

clrscr();

ofstream f1;

if(b==0)

{

f.open("contests.dat",ios::in|ios::binary);

f1.open("temp.dat",ios::out|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

{

if(C1.returnno()==c)

C1.addstat('W');

f1.write((char\*)&C1,sizeof(C1));

}

f1.close();

f.close();

remove("contests.dat");

rename("temp.dat","contests.dat");

}

else

{

f.open("contestd.dat",ios::in|ios::binary);

f1.open("temp.dat",ios::out|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

{

if(C1.returnno()==c)

C1.addstat('W');

f1.write((char\*)&C1,sizeof(C1));

}

f1.close();

f.close();

remove("contestd.dat");

rename("temp.dat","contestd.dat");

}

waiting \*temp=new waiting;

if(temp==NULL)

{

cout<<"\nOut Of Memory!";

menu();

}

else

{

temp->contno=c;

if(b==0)

temp->cate='S';

else

temp->cate='D';

temp->next=NULL;

}

if(front==NULL)

{

front=temp;

rear=temp;

}

else

{

rear->next=temp;

rear=temp;

}

delete temp;

clrscr();

gotoxy(34,12);

cout<<"Wait Listed! :|";

getch();

menu();

}

voidnotsels()

{

ofstream f1;

f.open("contests.dat",ios::in|ios::binary);

f1.open("temp.dat",ios::out|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

{

if(C1.returnno()==c)

C1.addstat('N');

f1.write((char\*)&C1,sizeof(C1));

}

f1.close();

f.close();

remove("contests.dat");

rename("temp.dat","contests.dat");

clrscr();

gotoxy(34,12);

cout<<"Not Selected! :(";

getch();

menu();

}

voidnotseld()

{ ofstream f1;

f.open("contestd.dat",ios::in|ios::binary);

f1.open("temp.dat",ios::out|ios::binary);

while(f.read((char\*)&C1,sizeof(C1)))

{

if(C1.returnno()==c)

C1.addstat('N');

f1.write((char\*)&C1,sizeof(C1));

}

f1.close();

f.close();

remove("contestd.dat");

rename("temp.dat","contestd.dat");

clrscr();

gotoxy(34,12);

cout<<"Not Selected! :(";

getch();

menu();

}

voidviewwait()

{ waiting \*tempo=front;

clrscr();

gotoxy(27,1);

cout<<"World Talent Show 2017!";

if(tempo==NULL)

{

cout<<"\n\tEmpty List!";

getch();

menu();

}

else

{

gotoxy(18,y);

cout<<"Contestant Number";

gotoxy(50,y);

cout<<"Category";

while(tempo!=NULL)

{

gotoxy(18,y+2);

cout<<tempo->contno;

gotoxy(50,y+2);

cout<<tempo->cate;

tempo=tempo->next;

y+=2;

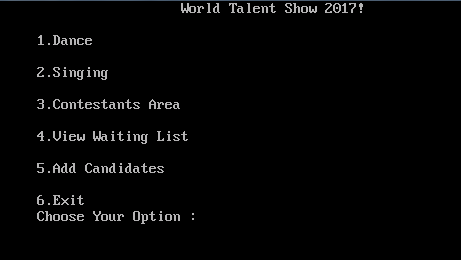
delay(200);

}

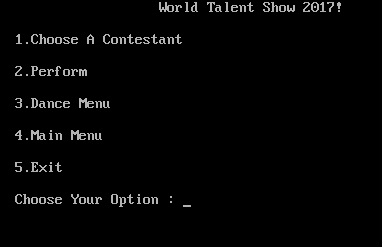
} }

**SCREEN SHOTS**



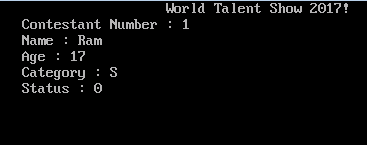


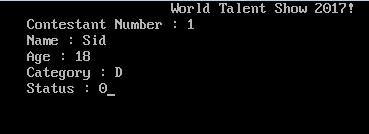


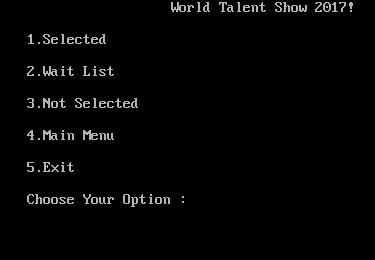




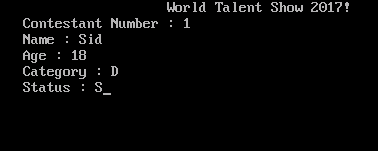


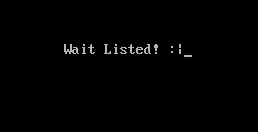


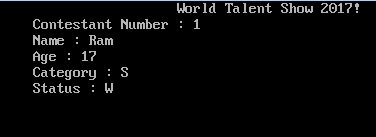




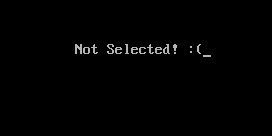


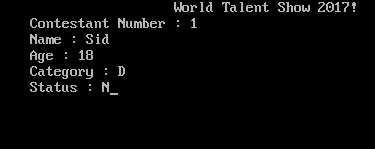












**DRAWBACKS**

* The program assumes that the contestant details are entered with sequential contestant numbers starting from 1. Otherwise it would fail.
* There is room for visual effects such as transition between different screens.
* Lack of colours and varying fonts makes the output screens less appealing and attractive.
* There is no option to search for a specific contestant and thus, it consumes time to access a particular contestant details amongst huge number of records.
* The wait list is designed to be temporary and thus gets cleared at the end of process. One needs to access the candidate details to get that information.

**BIBLIOGRAPHY**

**Bibliography**

* Computer Science With C++ (XI) – SumitaArora
* Computer Science With C++ (XII) – SumitaArora

**Webliography**

* www.learncpp.com
* www.cplusplus.com
* www.cprogramming.com