INTRODUCTION

The C++programming language was developed at AT&T Bell laboratories in the early igSo's by Bjarne Stroustrup. He found out "C" lacking for simulating and decided to extend the language by adding features from his favourite language. SimtaSj was one of the earliest object-oriented languages. Bjarne Stroustrup called it "C with classes" originally. The name C++ was coined by Rick Mascitti where"++"is the C increment operator. Ever since its birth, C++ evolved to cope with problems encountered by users, and through discussions at AT&T.
The major reasons for its success is the support for object oriented programming, which is most near to real world situations.

File Handling in C++

Files are a means to store data in a storage device. C++ file handling provides a mechanism to store output of a program in a file and read from a file on the disk.
Data file handling has been effectively used in the program.

This program uses the concept of object-oriented programming and data file handling.

The database is a collection of interrelated data to serve multiple applications. That is database programs create files of information. So we see that files are worked with most, inside the program.

ABOUT THE PROJECT

The project is designed to serve as a Movie Database Program in C++.
The user can access the data of a particular movie by creating a database and perform the following operations-

1. Search for a particular record
2. Count the number of records
3. Insert a record in the file
4. Delete a particular record
5. Edit/ modify a record
6. Reading the data from the file as per the given options

CLASS DEFINITION

Data Members

1. title(string)- Title of the movie
2. director(string)- Director of the movie
3. year(integer)- Year of release
4. runtime(integer)- Total runtime
5. genre(string)- Movie genre
6. actor(string)- Lead actor
7. producer(string)- Producer of the movie
8. age_res(character)- Age restriction
9. budget(float)- Movie budget
10. boxoffice(float)- Total profit at the boxoffice
11. Rating_IMDB(float)- IMDB rating
12. Rating_Rotten(float)- Rotten tomatoes rating
13. MovieID(integer)- Movie ID

Member Functions

1. getdata()- to enter the details of each movie
2. showdata()- To display the details of each movie
3. retname()- To retum the starting address of the name of the movie
4. retgenre()- To return the starting address of the genre of each movie
5. retid()- To return the ID of the movie
6. retdirector()- To return the starting address of the director’s name
7. retproducer()- To return the starting address of the producer’s name
8. retyear()- to return the release year of a movie

Coding

```
#include <iostream.h>
#include <conio.h>
#include <fstream.h>
#include <string.h>
#include <stdio.h>
class MovieData
{
char title[30];
int MovieID;
char director[30];
int year;
int runtime;
char genre[30];
char actor[30];
char producer[30];
char age_res;
float budget;
float boxoffice;
float Rating_IMDB;
float Rating_Rotten;
public:
void getdata() // Get movie data
{
cout << "\nEnter the title of the movie: ";
gets(title);
cout << "Enter movie ID: ";
cin >> MovieID;
cout << "Enter the Director's name of the movie: ";
gets(director);
cout << "Enter the year the movie was released: ";
cin >> year;
cout << "Enter runtime of the movie in minutes: ";
cin >> runtime;
cout << "Enter the producer of the movie: ";
gets(producer);
cout << "Enter the genre of the movie: ";
gets(genre);
cout << "Enter the lead actor of the movie: ";
gets(actor);
cout << "Movie Budget in crores: ";
cin >> budget;
cout << "Movie Boxoffice: ";
cin >> boxoffice;
cout << "IMDB rating: ";
cin >> Rating_IMDB;
cout << "Rotten Tomatoes Rating: ";
cin >> Rating_Rotten;
cout << "\n--------------------------------\n";
}
void showdata() // Display the movie information
{
cout << "\n--------------------------------\n";
cout << "\nBelow is the data of the movie:\n";
cout << "Movie Title " << title << endl;
cout << "Movie ID " << MovieID << endl;
cout << "Director " << director << endl;
cout << "Release Year " << year << endl;
cout << "Movie Runtime in minutes " << runtime << endl;
cout << "Producer " << producer << endl;
cout << "Genre " << genre << endl;
cout << "Lead actor " << actor << endl;
cout << "Budget " << budget << endl;
cout << "Boxoffice " << boxoffice << endl;
cout << "IMDB Rating " << Rating_IMDB << endl;
cout << "Rotten Tomatoes Rating " << Rating_Rotten << endl;
cout << "\n-------------------------------\n";
}
char* retname() // Return movie title
{
return (title);
}
char* retgenre() // Return genre
{
return genre;
}
int retid() // return movie id
{
return MovieID;
}
char* retproducer() // return name of the producer
{
return (producer);
}
```

```
char* retdirector() // return name of the director
{
return (director);
}
int retyear() // return year of release
{
return year;
}
};
char ans;
MovieData m, m2;
fstream file, temp;
void create() // function to create a file and get movie details from the user
{
file.open("data.dat", ios::out | ios::binary);
do
{
m.getdata();
file.write((char*)&m, sizeof(m));
cout << "do you want to enter another record?(y/n)";
cin >> ans;
} while ((ans == 'y') || (ans == 'Y'));
file.close();
}
void showfile() // function to show details of the file
{
file.open("data.dat", ios::in | ios::binary);
file.seekg(0);
while (file.read((char*)&m, sizeof(m)))
{
m.showdata();
}
file.close();
}
void search() // function to search for a movie
{
int choice;
cout << "Search on the basis of-\n 1. Name 2. Movie ID\n";
cin >> choice;
if (choice == 1) // search on the basis of name of the movie
{
file.open("data.dat", ios::in | ios::binary);
char mname[30];
```

```
int id;
cout << "Enter movie: ";
gets(mname);
while (file.read((char*)&m, sizeof(m)))
{
if (strcmp(mname, m.retname()) == 0)
{
m.showdata();
break;
}
}
file.close();
}
if (choice == 2) // search on the basis of movie ID
{
file.open("data.dat", ios::in | ios::binary);
int m_id;
cout << "Enter movie ID to be searched: ";
cin >> m_id;
while (file.read((char*)&m, sizeof(m)))
{
if (m.retid() == m_id)
{
m.showdata();
break;
}
}
file.close();
}
}
int count() // function to count the number of records
{
int choice, count = 0;
cout << "Count on the basis of- \n1. Producer 2. Director 3. Year 4. Count all
records\n";
cin >> choice;
if (choice == 1) // count the number of records on the basis of producer name
{
file.open("data.dat", ios::in | ios::binary);
char pname[30];
cout << "Enter name of the producer: ";
gets(pname);
while (file.read((char*)&m, sizeof(m)))
```

```
{
if (strcmp(pname, m.retproducer()) == 0)
{
count++;
}
}
file.close();
}
if (choice == 2) // count the number of records on the basis of director name
{
file.open("data.dat", ios::in | ios::binary);
char dname[30];
int id;
cout << "Enter name of the director: ";
gets(dname);
while (file.read((char*)&m, sizeof(m)))
{
if (strcmp(dname, m.retdirector()) == 0)
{
count++;
}
}
file.close();
}
if (choice == 3) // count the number of records on the basis of year of release
{
file.open("data.dat", ios::in | ios::binary);
int yr;
cout << "Enter release year: ";
cin >> yr;
while (file.read((char*)&m, sizeof(m)))
{
if (yr == m.retyear())
{
count++;
}
}
file.close();
}
if (choice == 4) // count total number of records
{
file.open("data.dat", ios::in | ios::binary);
while (file.read((char*)&m, sizeof(m)))
```

```
{
++count;
}
file.close();
}
return count;
}
void insert() // function to insert a new record in the file
{
int choice;
cout << endl << "Enter new record" << endl;
m.getdata();
cout << "Insert the record as :\n1.First record \n2.Last record \n3.At a specified ";
cout<<"record on the basis of position \n4.At a specified record on the basis of Movie name ";
cout<<"\n5.At a specified record on the basis of Movie ID\n";
cin >> choice;
if (choice == 1) // insert a new record file at the beginning of the file
{
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
file.seekg(0);
temp.write((char*)&m, sizeof(m));
while (file.read((char*)&m2, sizeof(m2)))
{
temp.write((char*)&m2, sizeof(m2));
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
if (choice == 2) // insert a new record at the end of the file
{
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
file.seekg(0);
while (file.read((char*)&m2, sizeof(m2)))
{
temp.write((char*)&m2, sizeof(m2));
}
temp.write((char*)&m, sizeof(m));
temp.close();
file.close();
```

```
remove("data.dat");
rename("new.dat", "data.dat");
}
if (choice == 3) // insert a new record in the file at a user given position
{
int n=1, n1;
cout << "Enter position: ";
cin >> n1;
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
file.seekg(0);
while (file.read((char*)&m2, sizeof(m2)))
{
n++;
if (n == n1)
{
temp.write((char*)&m, sizeof(m));
}
temp.write((char*)&m2, sizeof(m2));
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
if (choice == 4) // insert after a specific record on the basis of name of the movie
{
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
file.seekg(0);
char mname[30];
int id;
cout << "Enter movie name after which the new record is to added: ";
gets(mname);
while (file.read((char*)&m2, sizeof(m2)))
{
temp.write((char*)&m2, sizeof(m2));
if (strcmp(mname, m2.retname()) == 0)
{
temp.write((char*)&m, sizeof(m));
}
}
temp.close();
```

```
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
if (choice == 5) // insert after a specific record on the basis of movie ID
{
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
file.seekg(0);
int m_id;
cout << "Enter movie ID after which the new record is to be inserted after: ";
cin >> m_id;
while (file.read((char*)&m2, sizeof(m2)))
{
temp.write((char*)&m2, sizeof(m2));
if (m2.retid() == m_id)
temp.write((char*)&m, sizeof(m));
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
}
void del() // a function to delete a record from the file
{
int choice;
cout << "Delete record-\n 1.First \n 2.Last \n 3.nth \n 4.By MovieID \n 5.By name\n";
cin >> choice;
if (choice == 1) // delete first record from the file
{
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
file.read((char*)&m2, sizeof(m2));
while (file.read((char*)&m2, sizeof(m2)))
{
temp.write((char*)&m2, sizeof(m2));
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
```

```
if (choice == 2) // delete last record from the file
{
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
file.seekg(0, ios::end);
int l = file.tellg();
int s = sizeof(m);
int lr = l / s;
int r = 1;
file.seekg(0);
while (file.read((char*)&m2, sizeof(m2)))
{
if (r != lr)
temp.write((char*)&m2, sizeof(m2));
r++;
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
if (choice == 3) // delete nth record from the file
{
int r = 1, d;
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
cout << "Enter record to be deleted: ";
cin >> d;
file.seekg(0);
while (file.read((char*)&m2, sizeof(m2)))
{
if (r != d)
temp.write((char*)&m2, sizeof(m2));
r++;
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
if (choice == 4) // delete a record from the file on the basis of movie id
{
int n;
```

```
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
cout << "Enter record no. to be deleted: ";
cin >> n;
file.seekg(0);
while (file.read((char*)&m2, sizeof(m2)))
{
if (m2.retid() != n)
temp.write((char*)&m2, sizeof(m2));
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
if (choice == 5) // delete a record from the file on the basis of movie name
{
char n[30];
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
cout << "Enter Name of record to be deleted: ";
gets(n);
file.seekg(0);
while (file.read((char*)&m2, sizeof(m2)))
{
if (strcmpi(m2.retname(), n) != 0)
temp.write((char*)&m2, sizeof(m2));
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
}
void edit() // function to edit the records of a file
{
MovieData m3;
int choice;
// file.open("data.dat", ios::in|ios::binary);
cout << "Select file to be edited on the basis of- 1. Name 2. Movie ID\n";
cin >> choice;
if (choice == 1) // edit the records of a file on the basis of movie name
{
```

```
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
file.seekg(0);
char mname[30];
int id;
cout << "Enter Movie name: ";
gets(mname);
while (file.read((char*)&m2, sizeof(m2)))
{
if (strcmp(mname, m2.retname()) == 0)
{
cout << "Enter new data\n";
m3.getdata();
temp.write((char*)&m3, sizeof(m3));
}
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
if (choice == 2) // edit the records of a file on the basis of movie ID
{
file.open("data.dat", ios::in | ios::binary);
temp.open("new.dat", ios::out | ios::binary);
file.seekg(0);
int m_id;
cout << "Enter movie ID to be searched: ";
cin >> m_id;
while (file.read((char*)&m, sizeof(m)))
{
if (m.retid() == m_id)
{
cout << "Enter new data\n";
m3.getdata();
temp.write((char*)&m3, sizeof(m3));
}
}
temp.close();
file.close();
remove("data.dat");
rename("new.dat", "data.dat");
}
```

```
}
void main()
{
int ch;
clrscr();
char ch1;
do
{
clrscr();
cout << "\tMovie Database" << '\n';
cout << "1:Create new movie database" << '\n';
cout << "2:Read all records from the file" << '\n';
cout << "3:Search for records" << '\n';
cout << "4:Count the number of records" << '\n';
cout << "5:Insert record" << '\n';
cout << "6:Delete a record" << '\n';
cout << "7:Edit a record" << '\n';
cout << "\nEnter your choice\n";
cin >> ch;
switch (ch)
{
case 1:
create(); // calling create() function
break;
case 2:
showfile(); // caling showfile() function
break;
case 3:
search(); // calling search() function
break;
case 4:
cout << "Total number of records is: " << count(); // calling count() function
break;
case 5:
insert(); // calling insert() function
break;
case 6:
del(); // calling delete() function
break;
case 7:
edit(); // calling edit() function
break;
default:
```

```
cout << "Incorrect option";
}
cout << endl << "Do you want to go back to the menu?(y/n)\t" << endl;
cin >> ch1;
} while ((ch1 == 'y') || (ch1 == 'Y'));
```