NAME: - JITENDRA SAH

USN: -1NH17CS059

SEMESTER: - 4 “A”

MINI PROJECT: - MOVIE REVIEW

CHAPTER 1

INTRODUCTION

1.1 COURSE OBJECTIVES

Movie review is the program designed to know the gross production of the movie on the basis of the rating of the actor, actress, director, producer etc. It is designed to give the detail information of the movie. On the other hand, it will also predict the future possible outcomes of the movie.

The method to solve this problem is to create a software where details of each and every Actors, Directors and Producers will be there. It will be containing the ratings of each of them and we can find out about the upcoming movie whether it will be a Flop or Hit or Super Hit.

This software will contain the rating of each of them according to their performance of previous movies i.e. how their previous movie performed in the industry. It will be mostly guessing of how the upcoming movie is going to perform.

Now with the help of this software the producers or the film makers could know the outcome that is to come from the movie before its release because this software contains all the ratings and the past performance of the actors and directors. This software will make the work of producers easy and not let them think that they are risking their money on the movie and be assured that the movie will do better on the industry.

1.2 PROBLEM DEFINITION

In previous days film industry was not able to find out whether the movie that they are making will be hit or flop. Previously the movie industry used to make this record manually in registers and diaries. It was difficult for them to search those documents and find out about how their movie is going to perform on the box office. The producers sometime had to risk their money on the movie production because they couldn't figure out how the movie will perform in the market with the actors and directors that are casted in the movie. They also had to face the loss many times.

CHAPTER 2

OBJECT ORIENTED FEATURES

2.1 POLYMORPHISM

The ability to use an operator or function in different ways, in other words giving different meaning or functions to the operators or functions is called polymorphism. Poly refers to 'many'. That is a single function or an operator functioning in many ways different upon the usage is called polymorphism

Function overloading and Operator overloading are examples of polymorphism. Polymorphism is a feature using which an object behaves differently in different situation.

In function overloading we can have more than one function with same name but different numbers, type or sequence of arguments.

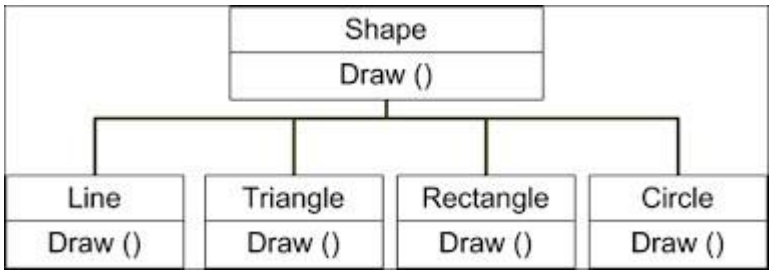


Fig. 2.1: Polymorphism

2.2 DATA ABSTRACTION

Data abstraction refers to, providing only essential information to the outside world and hiding their background details, i.e., to represent the needed information in program without presenting the details.

For example, when you send a SMS you just type the message, select the contact and click send, the phone shows you that the message has been sent, what actually happens in background when you click send is hidden from you as it is not relevant to you.

2.3 CLASS

Class in OOPs is a user-defined type. It is declared by using the keyword 'class'. It has two compartments namely 'Private' and 'Public'. It includes data members which are called class data member which comes under the private section of class and the functions which are called as class member functions that comes under the public section of the class. The data member in private section of class are protected and are not easily accessible by any other functions except the functions that are declared with in the public section of class. It is also called as the collection of objects.

Its syntax is,

```
Class (classname){
Private:
        (data members);
Public:
        (member functions);
};
```

2.4 OBJECT

An object is instance of a class. We can create as my object as we want. Each object created will possess the values of all the data members of the corresponding class. We create an object and access the private members of the class with the help of object dot functions. All the member functions of a

class are called using the object. Each object shares the same member functions to access the data members of the class. A constructor is also called while creating an object.

This is the basic unit of object-oriented programming. That is both data and function that operate on data are bundled as a unit called as object. Object are the basic run time entities in an object-oriented system. It can be a person, place, bank account, table of data or any item that the program has to handle. Each object contains data and functions to manipulate data. Object is inheritance of class. The syntax to create an object is;

<class_ name> <object _name>;

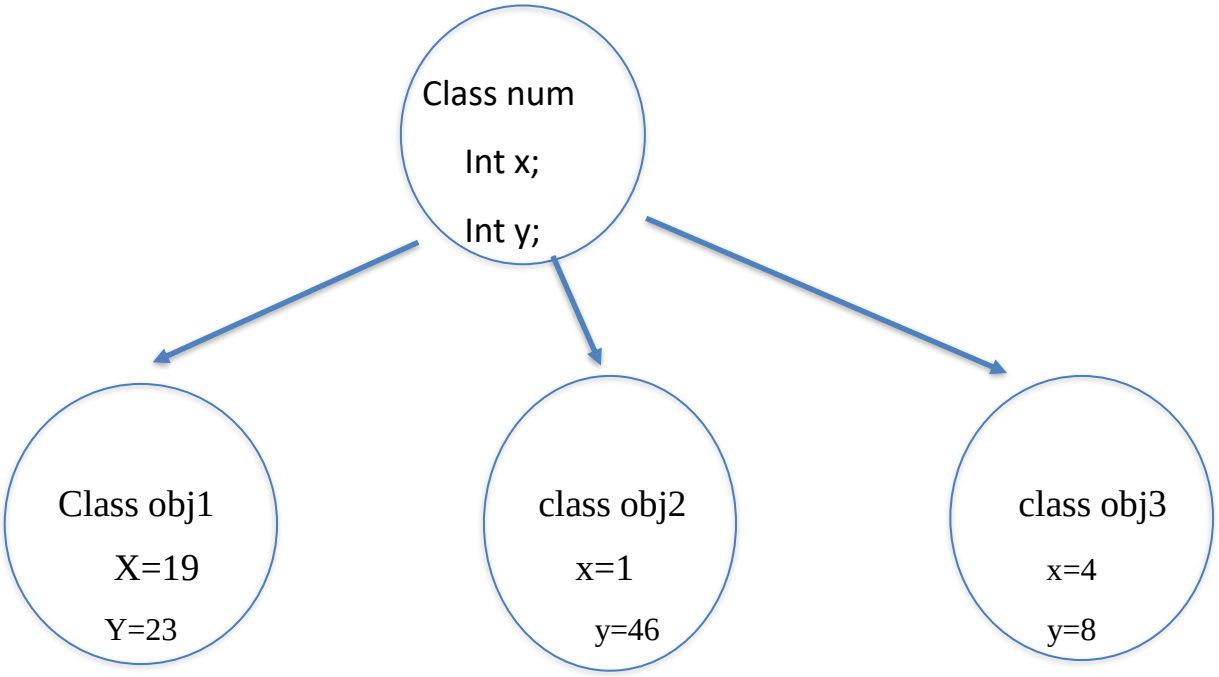


Fig. 2.2: class and object

2.5 ENCAPSULATION

The wrapping up of data and functions into a single unit (called Class) is known as Encapsulation. Encapsulation is placing the data and the functions that work on that data in the same place. While working with procedural languages, it is not always clear which functions work on which variables but object-oriented programming provides you framework to place the data and the relevant functions together in the same object. It is a process of combining data and function into a single unit like capsule. This is to avoid the access of private data members from outside the class. To achieve encapsulation, we make all data members of class private and create public functions, using them we can get the values from these data members or set the value to these data members.

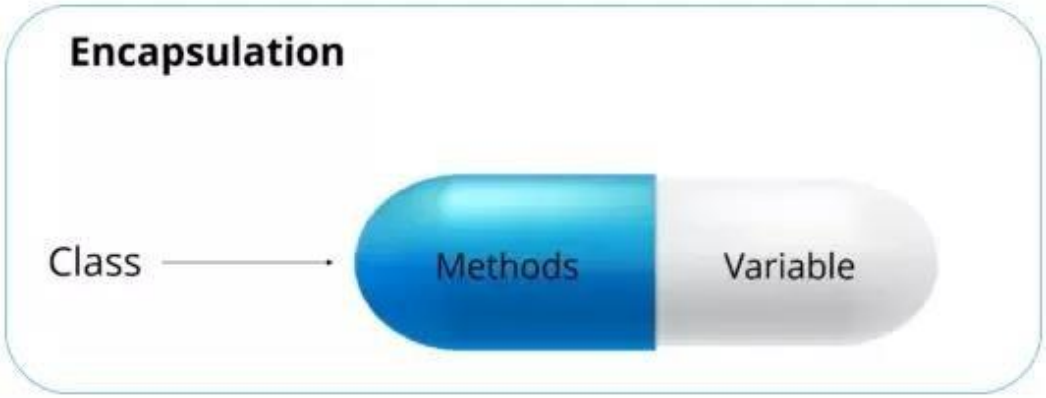


Fig. 2.3: Encapsulation

2.6 Inheritance

One of the most useful aspects of object-oriented programming is code reusability. Inheritance is the process in which object of child class will acquire/inherit the properties of parent class. Through this each derived class will shares the properties of base class. As the name suggests Inheritance is the process of forming a new class from an existing class that is from the existing class called as base class, new class is formed called as derived class. Through this each derived class shares the properties of base class.

This is a very important concept of object-oriented programming since this feature helps to reduce the code size.

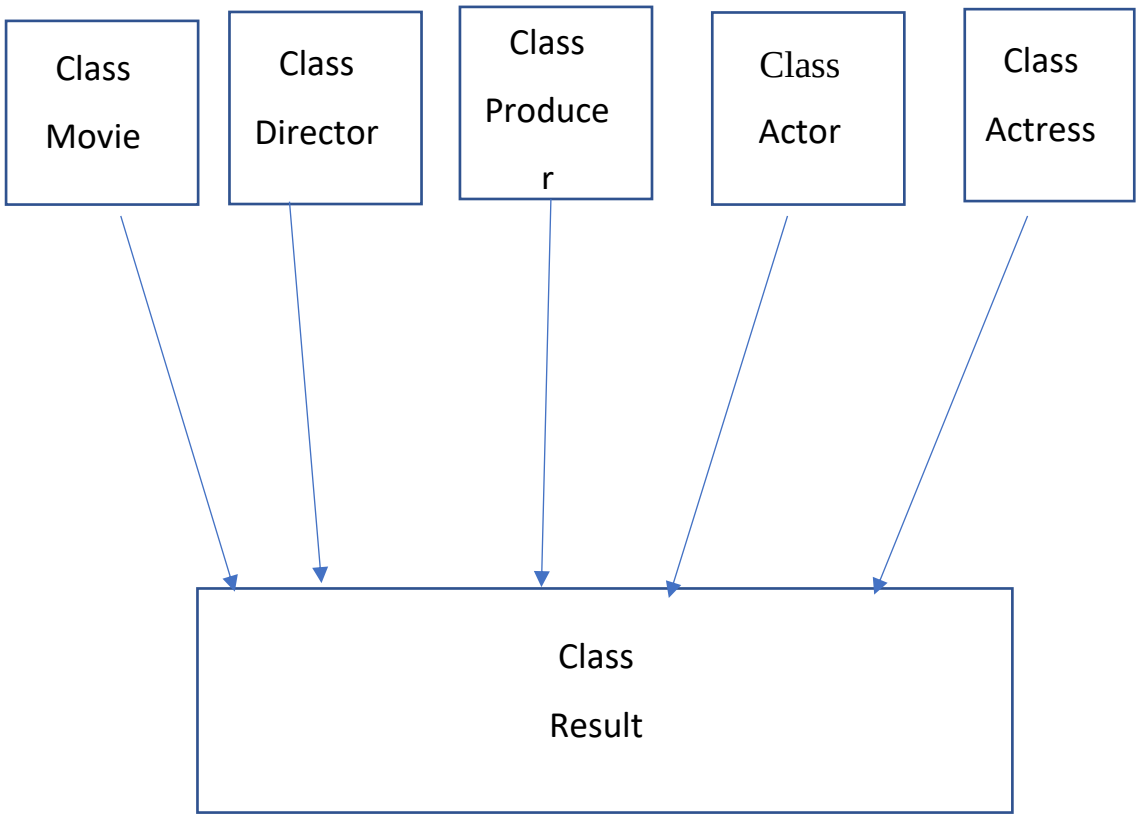


Fig 2.4: Inheritance

CHAPTER 3

REQUIREMENTS AND DESIGN

3.1 HARDWARE CONFIGURATION

Processor : Intel Core i3 or above

Hard Disk : 5 GB or above

RAM : 512MB or more.

3.2 SOFTWARE SPECIFICATION

Operation System : WINDOWS XP or above.

Compiler : Turbo C++ ,Code blocks, dev C++

3.3 ALGORITHM

Step 1.

Start

Step 2.

A class Movie will be created with objects of actor, actress, director, producer.

Step 3.

The name and details of all the celebrities are stored in private section so that they can only be used by the member function.

Step 4.

Using the concept of friend function and friend class in order to link all these objects

Step 5.

Getting the inputs from the users. Like the tittle of movie & name of the actor, actress, director, producer.

Step 6.

On the basis of the names we calculate the rating of the movie.

Step 7.

The rating is calculated on the basis of the information stored in the file.

Step 8.

The calculated rating is used to know the movie will be hit, super hit of flop.

Step 9.

If the rating of the movie is less than 5 then the movie will be flop.

Else if the rating of the movie is more than 4and less than 7 than the Movies is hit.

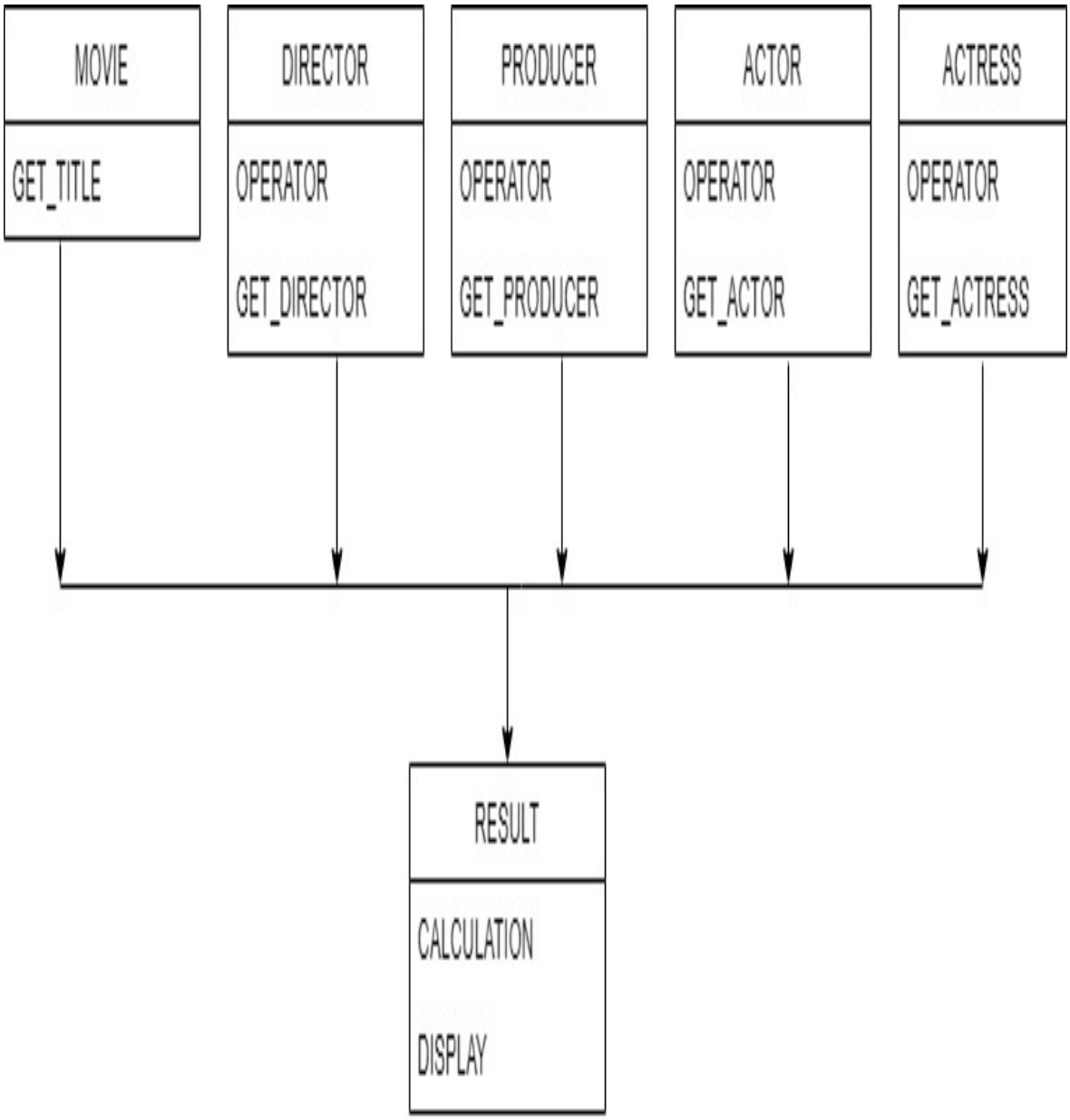
Else the rating will be super hit.

Step 10.

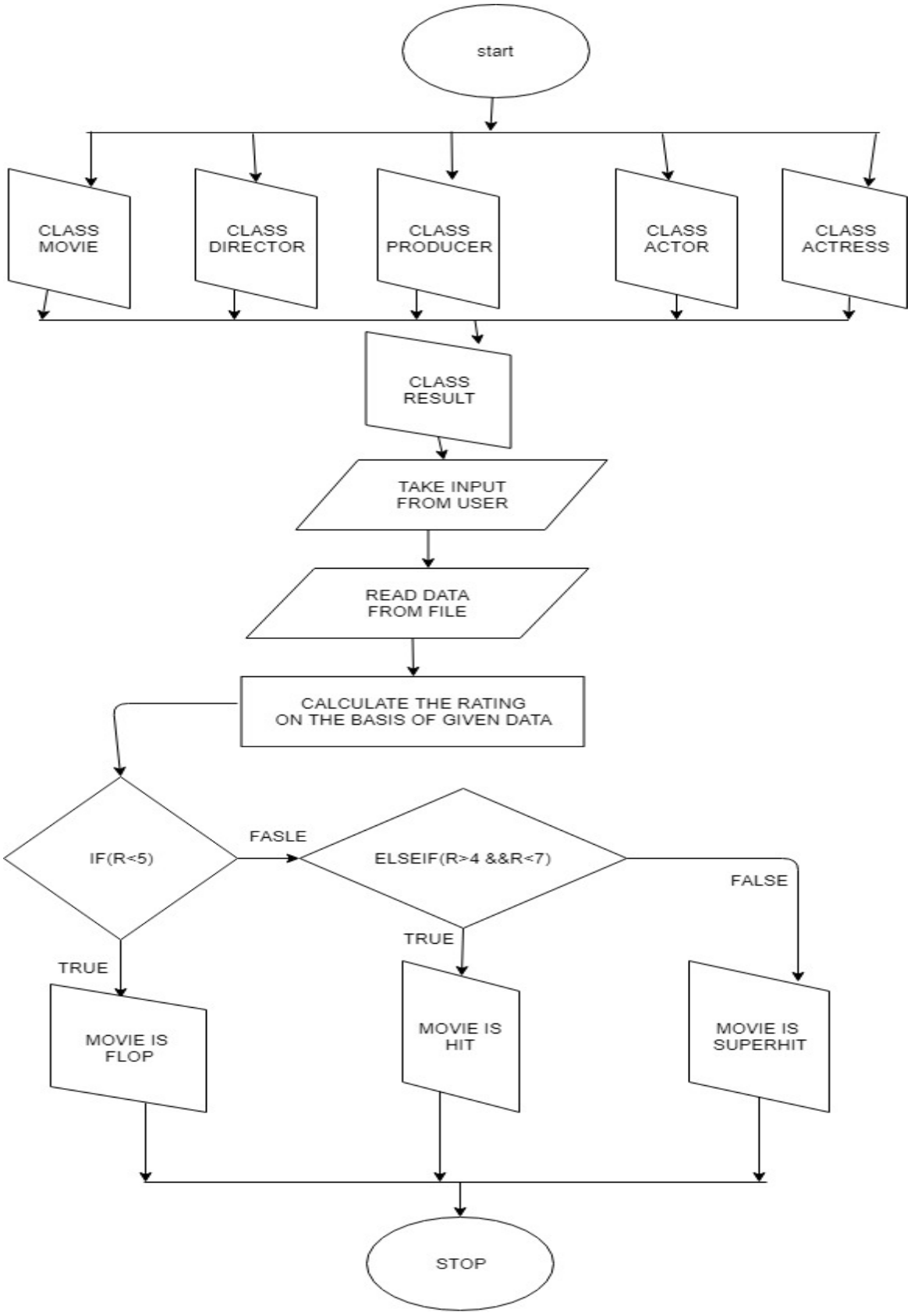
The rating is saved in the different class.

Step 11. End

3.4 CLASS DIAGRAM



3.5 DATA FLOW DIAGRAM



CHAPTER 4

IMPLEMENTATION

This code has been implemented using C++ coding. The code for this project is given below:

```
#include<iostream>
#include<conio.h>
#include<iomanip>
#include<fstream>
using namespace std;
class movie
{
public:
char title[20];
void get_title()
{
 cout<<"enter the title of movie:";
 cin.getline(title,20);
}
};
class director
{
public:
     int n1;
     float arr1[26];
     std::size_t pos;
 std::string str3;
   string data;
    string str1;
    string num;
    string ans1;
    char op;
```

```
director()
{
  arr1[1]=3.5;
  arr1[2]=4.5;
  arr1[3]=7;
  arr1[4]=9;
  arr1[5]=6.7;
  arr1[6]=3.5;
  arr1[7]=4.5;
  arr1[8]=7;
  arr1[9]=9;
  arr1[10]=6.7;
  arr1[11]=3.5;
  arr1[12]=4.5;
  arr1[13]=7;
  arr1[14]=9;
  arr1[15]=6.7;
  arr1[16]=3.5;
  arr1[17]=4.5;
  arr1[18]=7;
  arr1[19]=9;
  arr1[20]=6.7;
  arr1[21]=3.5;
  arr1[22]=4.5;
  arr1[23]=7;
  arr1[24]=9;
  arr1[25]=6.7;
}
```

```
}
void operator =(director r1)
{
  if(num=="01") n1=1;
  else if(num=="02") n1=2;
  else if(num=="03") n1=3;
  else if(num=="04") n1=4;
  else if(num=="05") n1=5;
  else if(num=="06") n1=6;
  else if(num=="07") n1=7;
  else if(num=="08") n1=8;
  else if(num=="09") n1=9;
  else if(num=="10") n1=10;
  else if(num=="11") n1=11;
  else if(num=="12") n1=12;
  else if(num=="13") n1=13;
  else if(num=="14") n1=14;
  else if(num=="15") n1=15;
  else if(num=="16") n1=16;
  else if(num=="17") n1=17;
  else if(num=="18") n1=18;
  else if(num=="19") n1=19;
  else if(num=="20") n1=20;
  else if(num=="21") n1=21;
  else if(num=="22") n1=22;
  else if(num=="23") n1=23;
  else if(num=="24") n1=24;
  else if(num=="25") n1=25;
```

```
86      void get_director()
87      {
88
89        ifstream infille;
90       infille.open("director.dat");
91       cout<<"\nthe name of the directors are:\n";
92    while(infille.eof()!=1)
93     {
94    getline(infille,data);
95    size_t found = data.find(".");
96    if(found!=string::npos)
97    cout<<data<<endl;
98     }
99     infille.close();
100      ifstream infile;
101      infile.open("director.dat");
102       cout<<"\nEnter director no.";
103       getline(cin,num);
104      // open a file in read mode.
105      cout << "Reading from the file" << endl;
106   cout<<"\n";
107
108   while(infile.eof()!=1)
109    {
110   getline(infile,data);
111   size_t found = data.find(num);
112   if(found!=string::npos)
113  {
```

```
114   cout<<data<<endl;       //final output to be stored in string
115   ans1=data;
116    }
117    }
118   infile.close();
119  }
120  };
121    class producer
122    {
123
124   public:
125       int n2;
126       float arr2[26];
127        std::size_t pos;
128    std::string str3;
129      string data;
130       string str1;
131       string num;
132       string ans2;
133       char op;
134       producer()
135       {
136         arr2[1]=3.5;
137         arr2[2]=4.5;
138         arr2[3]=7;
139         arr2[4]=9;
140         arr2[5]=6.7;
141         arr2[6]=3.5;
```

```
148       arr2[13]=7;
149       arr2[14]=9;
150       arr2[15]=6.7;
151       arr2[16]=3.5;
152       arr2[17]=4.5;
153       arr2[18]=7;
154       arr2[19]=9;
155       arr2[20]=6.7;
156       arr2[21]=3.5;
157       arr2[22]=4.5;
158       arr2[23]=7;
159       arr2[24]=9;
160       arr2[25]=6.7;
161     }
162     void operator =(producer r1)
163     {
164       if(num=="01") n2=1;
165       else if(num=="02") n2=2;
166       else if(num=="03") n2=3;
167       else if(num=="04") n2=4;
168       else if(num=="05") n2=5;
169       else if(num=="06") n2=6;
170       else if(num=="07") n2=7;
171       else if(num=="08") n2=8;
172       else if(num=="09") n2=9;
173       else if(num=="10") n2=10;
174       else if(num=="11") n2=11;
175       else if(num=="12") n2=12;
```

```
171       else if(num=="08") n2=8;
172       else if(num=="09") n2=9;
173       else if(num=="10") n2=10;
174       else if(num=="11") n2=11;
175       else if(num=="12") n2=12;
176       else if(num=="13") n2=13;
177       else if(num=="14") n2=14;
178       else if(num=="15") n2=15;
179       else if(num=="16") n2=16;
180       else if(num=="17") n2=17;
181       else if(num=="18") n2=18;
182       else if(num=="19") n2=19;
183       else if(num=="20") n2=20;
184       else if(num=="21") n2=21;
185       else if(num=="22") n2=22;
186       else if(num=="23") n2=23;
187       else if(num=="24") n2=24;
188       else if(num=="25") n2=25;
189     }
190
191
192     void get_producer()
193     {
194         ifstream infille;
195    infille.open("producer.dat");
196    cout<<"\nthe name of the producers are:\n";
197 while(infille.eof()!=1)
198  {
```

```
198  {
199  getline(infille,data);
200  size_t found = data.find(".");
201  if(found!=string::npos)
202  cout<<data<<endl;
203  }
204   infille.close();
205     ifstream infile;
206     infile.open("producer.dat");
207      cout<<"\nEnter producer no.";
208      getline(cin,num);
209
210     // open a file in read mode.
211     cout << "Reading from the file" << endl;
212  cout<<"\n";
213
214  while(infile.eof()!=1)
215  {
216  getline(infile,data);
217  size_t found = data.find(num);
218  if(found!=string::npos)
219 {
220  cout<<data<<endl;       //final output to be stored in string
221  ans2=data;
222  }
223  }
224  infile.close();
225
```

```
228  };
229   class actor
230   {
231
232  public:
233       int n3;
234       std::size_t pos;
235   std::string str3;
236     string data;
237      string str1;
238      string num;
239      string ans3;
240      float arr3[26];
241      char op;
242
243
244      actor()
245      {
246        arr3[1]=3.5;
247        arr3[2]=4.5;
248        arr3[3]=7;
249        arr3[4]=9;
250        arr3[5]=6.7;
251        arr3[6]=3.5;
252        arr3[7]=4.5;
253        arr3[8]=7;
254        arr3[9]=9;
255        arr3[10]=6.7;
```

```
258         arr3[13]=7;
259         arr3[14]=9;
260         arr3[15]=6.7;
261         arr3[16]=3.5;
262         arr3[17]=4.5;
263         arr3[18]=7;
264         arr3[19]=9;
265         arr3[20]=6.7;
266         arr3[21]=3.5;
267         arr3[22]=4.5;
268         arr3[23]=7;
269         arr3[24]=9;
270         arr3[25]=6.7;
271       }
272       void operator =(actor r1)
273       {
274         if(num=="01") n3=1;
275         else if(num=="02") n3=2;
276         else if(num=="03") n3=3;
277         else if(num=="04") n3=4;
278         else if(num=="05") n3=5;
279         else if(num=="06") n3=6;
280         else if(num=="07") n3=7;
281         else if(num=="08") n3=8;
282         else if(num=="09") n3=9;
283         else if(num=="10") n3=10;
284         else if(num=="11") n3=11;
285         else if(num=="12") n3=12;
```

```
286         else if(num=="13") n3=13;
287         else if(num=="14") n3=14;
288         else if(num=="15") n3=15;
289         else if(num=="16") n3=16;
290         else if(num=="17") n3=17;
291         else if(num=="18") n3=18;
292         else if(num=="19") n3=19;
293         else if(num=="20") n3=20;
294         else if(num=="21") n3=21;
295         else if(num=="22") n3=22;
296         else if(num=="23") n3=23;
297         else if(num=="24") n3=24;
298         else if(num=="25") n3=25;
299       }
300
301       void get_actor()
302       {
303             ifstream infille;
304      infille.open("actor.dat");
305      cout<<"\nthe name of the actors are:\n";
306    while(infille.eof()!=1)
307    {
308    getline(infille,data);
309    size_t found = data.find(".");
310    if(found!=string::npos)
311    cout<<data<<endl;
312    }
313    infille.close();
```

```
313   infille.close();
314     ifstream infile;
315     infile.open("actor.dat");
316      cout<<"\nEnter actor no.";
317      getline(cin,num);
318     // open a file in read mode.
319     cout << "Reading from the file" << endl;
320   cout<<"\n";
321
322   while(infile.eof()!=1)
323    {
324   getline(infile,data);
325   size_t found = data.find(num);
326   if(found!=string::npos)
327   {
328   cout<<data<<endl;       //final output to be stored in string
329   ans3=data;
330    }
331    }
332   infile.close();
333
334   }
335   };
336    class actress
337    {
338
339   public:
340        int n;
```

```
343     string data;
344      string str1;
345      string num;
346      string ans4;
347      float arr[26];
348      char op;
349      actress()
350      {
351        arr[1]=3.5;
352        arr[2]=4.5;
353        arr[3]=7;
354        arr[4]=9;
355        arr[5]=6.7;
356        arr[6]=3.5;
357        arr[7]=4.5;
358        arr[8]=7;
359        arr[9]=9;
360        arr[10]=6.7;
361        arr[11]=3.5;
362        arr[12]=4.5;
363        arr[13]=7;
364        arr[14]=9;
365        arr[15]=6.7;
366        arr[16]=3.5;
367        arr[17]=4.5;
368        arr[18]=7;
369        arr[19]=9;
370        arr[20]=6.7;
```

```
371         arr[21]=3.5;
372         arr[22]=4.5;
373         arr[23]=7;
374         arr[24]=9;
375         arr[25]=6.7;
376       }
377       void operator =(actress r1)
378       {
379         if(num=="01") n=1;
380         else if(num=="02") n=2;
381         else if(num=="03") n=3;
382         else if(num=="04") n=4;
383         else if(num=="05") n=5;
384         else if(num=="06") n=6;
385         else if(num=="07") n=7;
386         else if(num=="08") n=8;
387         else if(num=="09") n=9;
388         else if(num=="10") n=10;
389         else if(num=="11") n=11;
390         else if(num=="12") n=12;
391         else if(num=="13") n=13;
392         else if(num=="14") n=14;
393         else if(num=="15") n=15;
394         else if(num=="16") n=16;
395         else if(num=="17") n=17;
396         else if(num=="18") n=18;
397         else if(num=="19") n=19;
398         else if(num=="20") n=20;
```

```
399         else if(num=="21") n=21;
400         else if(num=="22") n=22;
401         else if(num=="23") n=23;
402         else if(num=="24") n=24;
403         else if(num=="25") n=25;
404       }
405
406       void get_actress()
407       {
408        ifstream infille;
409      infille.open("actress.dat");
410      cout<<"\nthe name of the actress are:\n";
411    while(infille.eof()!=1)
412     {
413    getline(infille,data);
414    size_t found = data.find(".");
415    if(found!=string::npos)
416    cout<<data<<endl;
417     }
418     infille.close();
419       ifstream infile;
420       infile.open("actress.dat");
421        cout<<"\nEnter actress no.";
422        getline(cin,num);
423
424
425       // open a file in read mode.
426       cout << "Reading from the file" << endl;
```

```
427  cout<<"\n";
428
429  while(infile.eof()!=1)
430   {
431  getline(infile,data);
432  size_t found = data.find(num);
433  if(found!=string::npos)
434  {
435  cout<<data<<endl;       //final output to be stored in string
436  ans4=data;
437  }
438   }
439  infile.close();
440
441  }
442   };
443   class result:public movie,public director,public producer,public actor,public actress
444   {
445  public:
446     double d,p,a,ac,res;
447      float calculation()
448      {
449          res=((arr1[n1]+arr2[n2]+arr3[n3]+arr[n])/4);
450         return res;
451  }
452
453      void display()
454      {
```

```
453      void display()
454      {
455
456         cout<<endl<<"The Title of the movie = "<<title;
457         cout<<endl<<"The director of the movie = "<<ans1<<" having rating ="<<arr1[n1];
458         cout<<endl<<"The producer of the movie = "<<ans2<<" having rating ="<<arr2[n2];
459         cout<<endl<<"The actor of the movie = "<<ans3<<" having rating ="<<arr3[n3];
460         cout<<endl<<"The actress of the movie = "<<ans4<<" having rating ="<<arr[n];
461
462       if(res<5)
463       {
464       cout<<"\nREVIEW="<<res;
465           cout<<"\nMovie is flop !!!!!!!";}
466     else if(res>4 && res<7)
467       {   cout<<"\nREVIEW="<<res;
468        cout<<"\nMovie is hit !!!!!!!";}
469     else
470     {
471     cout<<"\nREVIEW="<<res;
472        cout<<"\nMovie is SuperHit!!!";}
473
474
475      }
476   };
477  int main()
478  {
479  result r1;
480  r1.get_title();
```

```
464         cout<<"\nREVIEW="<<res;
465             cout<<"\nMovie is flop !!!!!!!";}
466       else if(res>4 && res<7)
467         {   cout<<"\nREVIEW="<<res;
468          cout<<"\nMovie is hit !!!!!!!";}
469       else
470       {
471       cout<<"\nREVIEW="<<res;
472          cout<<"\nMovie is SuperHit!!!";}
473
474
475       }
476     };
477     int main()
478     {
479     result r1;
480     r1.get_title();
481     r1.get_director();
482     r1.get_producer();
483     r1.get_actor();
484     r1.get_actress();
485     r1=r1;
486     r1.calculation();
487     r1.display();
488     getch();
489     return 0;
490     }
491
```

CHAPTER 5

OUTPUT SNAPSHOTS

Fig.5.1 Main page

Fig.5.2 List of Directors

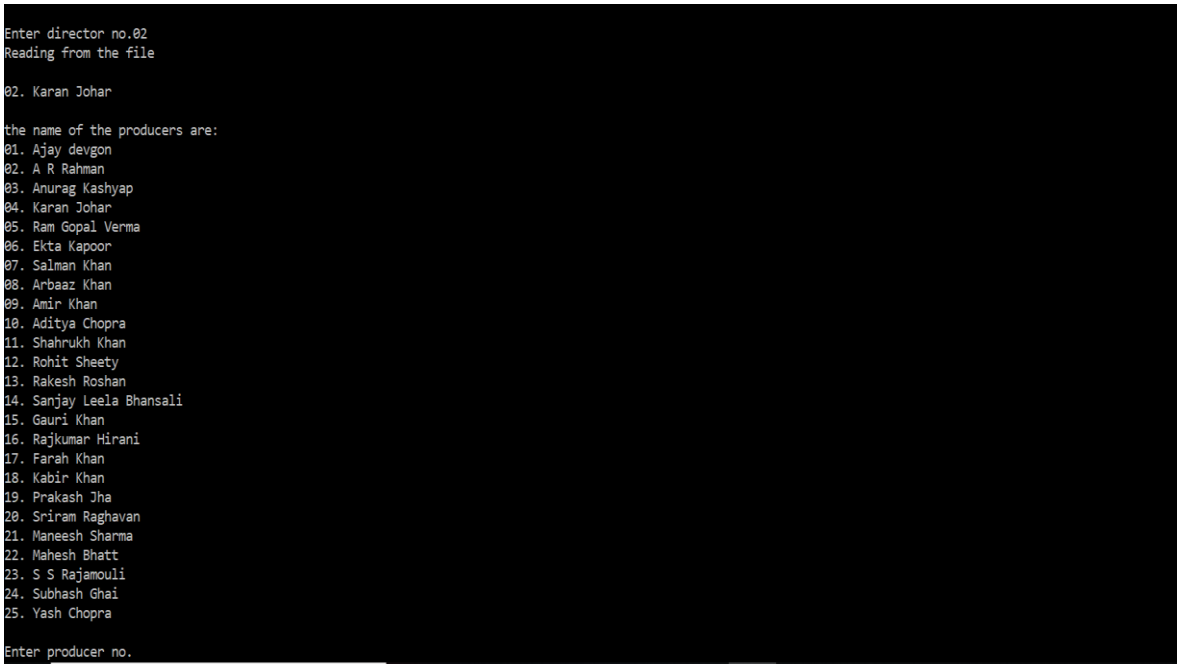


Fig.5.3 List of Producers

Fig.5.4 List of Actors

```
Enter actor no.21
Reading from the file

21. Varun Dhawan

the name of the actress are:
01. Deepika Padukone
02. Kajal Agarwal
03. Priyanka Chopra
04. Kareena Kapoor
05. Kangana Ranaut
06. Katrina Kaif
07. Alia Bhatt
08. Sara Ali Khan
09. Shraddha Kapoor
10. Kriti Sanon
11. Jacqueline Fernandez
12. Sonakshi Sinha
13. Sonam Kapoor
14. Madhuri Dixit
15. Anushka Sharma
16. Aishwarya Rai
17. Yami Gautam
18. Kajol
19. Shilpa Shetty
20. Rakul preet singh
21. Tamannaah Bhatia
22. Shruti Haasan
23. Adah Sharma
24. Amy Jackson
25. Samantha Ruth Prabhu

Enter actress no.
```

Fig.5.5 List of Actress

```
Enter actress no.07
Reading from the file

07. Alia Bhatt

The Title of the movie = KALANK
The director of the movie = 02. Karan Johar having rating =4.5
The producer of the movie = 05. Ram Gopal Verma having rating =6.7
The actor of the movie = 21. Varun Dhawan having rating =3.5
The actress of the movie = 07. Alia Bhatt having rating =4.5
REVIEW=4.8
Movie is flop !!!!!!!
```

Fig.5.6 Result

CHAPTER 6

CONCLUSION

Now with the help of this software the producers or the film makers could know the outcome that is to come from the movie before its release because this software contains all the ratings and the past performance of the actors and directors. This software will make the work of producers easy and not let them think that they are risking their money on the movie and be assured that the movie will do better on the industry.

CHAPTER 7

REFERENCES

1. www.scribd.com/cpp
2. www.geeksforgeeks.org/c-plus-plus/
3. www.slideshare.net/cplusplus
4. www.topcoder.com/community/competitive-programming/tutorials/
5. “C++: The Complete Reference”, Fourth Edition by Herbert Schilhdt, McGraw Hill Education (India).