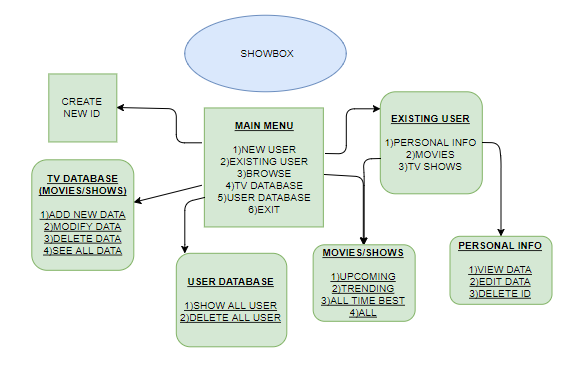
# INTRODUCTION

In todays world of Netflix, Hulu, Amazon prime, and other media consumption platform it is hard to keep track of not only the shows and movies that are launched recently but also the ones we have already watched or like to watch in future.

To solve this problem, we have devised an elegant program, SHOWBOX, which keep tracks of all the shows and movies which are upcoming, trending and all-time best as well as distinct personalized bookmark for our users with IDs.

This program can create, delete, edit a file and search various file. It has many other functions which prevent various errors as data redundancy and manage data deletion universally.

# MENU DIAGRAM



USER MANUAL

1. First the user has to insert the CD and execute the file name as “SHOWBOX.CPP” using Turbo C++. For doing so the user has to copy this file to C:\TC\BIN, then open DOSBox and open Turbo C++. Then the user can compile the program and run it.
2. After doing so, an animation will follow, welcoming the user, after which the main menu will appear which will include the following options to access application further
3. The Main Screen will show listing three options.

* **New User*:*** Press 1 if you want to create new ID.
* **Existing User**: Press 2 if you an existing user.
* **Browse**: Press 3 if you wish to browse without creating ID.
* **Access tv’s database**: Press 4 if you want to access tv’s database
* **Access user’s database**: Press 5 if you want to access user’s database
* **Exit:** Press 6 if you want to leave.

**New User:**

If you choose this option, a page will be displayed to create new ID.

**Existing User:**

If you choose EXISTING USER, the following menu will be displayed after entering your username:

**1)PERSONAL INFO:** The following menu will appear showing the content as per it’s option.

**2)MOVIES:** Let you browse and access your bookmarks.

**3)TV SHOWS:** Let you browse and access your bookmarks.

**Browse:**

If you choose this option, the browse menu will be displayed to give you choice to select if you want to see movies or series and choosing which category:

**1)UPCOMING**

**2)TRENDING**

**3)ALL TIME BEST**

**4)ALL**

**Access tv’s Database:**

If you choose this option, the following menu will be displayed and will be provided to enter either movies or series.

Then following options will be displayed

**1)ADD NEW DATA**

**2)MODIFY DATA**

**3)DELETE DATA**

**4)SEE ALL DATA**

**Access user’s Database:**

If you choose this option, the following menu will be displayed having two options,

**1)SHOW ALL DATA**

**2)DELETE ALL DATA**.

**Exit:**

If you choose this option, you will exit the program

SCOPE OF PROJECT

**New User**

User can create his/her own id which can be used to bookmark movies and shows to keep track of the same.

**Existing User**

After entering respective username, user gets options to browse and bookmark movies and shows after going through various categories. This option also lets user to access, delete, and modify his or her personal info.

**Browse**

This option lets a user who does not have an id simply browse movies and shows.

**Access tv’s Database**

It lets user to add, modify, and delete movies or shows from the database. It also lets user to access all movies or shows at once.

**Access user’s Database**

It allows user to navigate through all users. It also allows file to reset by deleting every existing user.

# ASSUMPTIONS MADE

* User knows basic turbo C++ terminology
* Any name is not exceeding the size of 20 bytes
* Username and email is not exceeding the size of 40 bytes
* User is not going to bookmark more than 20 bookmarks
* User checks the movie and series code before doing any kind of manipulation
* Email address entered is already verified and is in proper form

# DATA FILES USED

The following data files were used in the program:

1. **USER.dat**

This binary file contains data of all the users.

1. **MOVIES.dat**

This binary file contains data of all the movies.

1. **SHOWS.dat**

This binary file contains data of all the shows.

1. **TCONFIG.dat**

This binary is used for generating unique codes for Series and Movies.

# CLASSES AND STRUCTURES USED

**Class User**

Describes the interface regarding every user related menu as well as allows manipulation in user data.

**Class tv\_config**

Helps in unique id generation for movies and shows and saves data in file to prevent data loss after exiting from program.

**Structure TV**

Handles both movies and shows data file implementation by differentiating through opening different files depending on the use.

# FUNCTIONS USED

**The following functions are used in this program:**

* **int inputchoice():** Checks that input is single digit integer type character with no excess following data and helps cin to get out of failure state.
* **int Newuser():** Creates new user id.
* **int Search():** Runs a search in “User.dat” on the basis of username
* **int TSearch(int):** Runs a search in “Movies.dat” or “Shows.dat” depending on the parameter on the basis of code.
* **void Euser():** Provides menu and option after entering username for existing user.
* **int PersonalI(int):** Provides option to view and manipulate personal info.
* **void Movies(int):** Gives options regarding Movies menu
* **void Series(int):** Gives options regarding Series menu
* **void TModify(int,int):** Modifies certain movie or show
* **void TDelete(int,int):** Delete certain movie or show
* **void TAdd(int):** Add certain movie or show
* **void Show\_Bookmarks(int,int):** Shows bookmarks
* **int Browse(int,int):** Browse for existing user
* **void GBrowse():** Browse for guest
* **void Allshow(int):** Shows all movies or shows
* **int CheckName(char []):** checks proper formatting of entered name
* **void BookmarkDelete(int,int,int):** Deletes bookmark for user
* **void shift(int,int [],int):** Shifts bookmarks array after deletion
* **void BDeleteU(int,int):**  Deletes a bookmarks from every database
* **void Udatabase():** Give whole file-based option of “user.dat”
* **void Enter():** Function for Entering the Data of User.
* **void Show() :** Function for Showing the Data of User.
* **char \*ReturnUser():** Function for Returning the User ID of User.
* **int Write():** Function for Writing the Data of User on file.
* **void Read():** Function for Reading the Data of User from file.
* **void Modify():** Function for Modifying the Data of User.
* **int Delete(int):** Function for Deleting the Data of User.
* **void Add\_Bookmark(int, int, int):** Function for Bookmarking.
* **void main()**

# DATA STRUCTURES USED

**CIRCULAR QUEUE**

We have implemented circular queue algorithms to display single records from file using keyboard keys to help user navigate through the file in both left and right direction.

This implementation has been done in show all user sub menu

**ONE DIMENSIONAL ARRAY**

Tname[20]: Stores movies or series name

Tcategory[3]: Stores Boolean values depending on categories

Uname[20]: Stores User’s name

Uemail[40]: Stores User’s Email

UserID[40]: Stores Username

SBookmark[20]: Stores code of bookmarked series

MBookmark[20]: Stores code of bookmarked movies

# VERIFICATION & VALIDATIONS

Data checking functions are implemented at numerous places throughout the program.

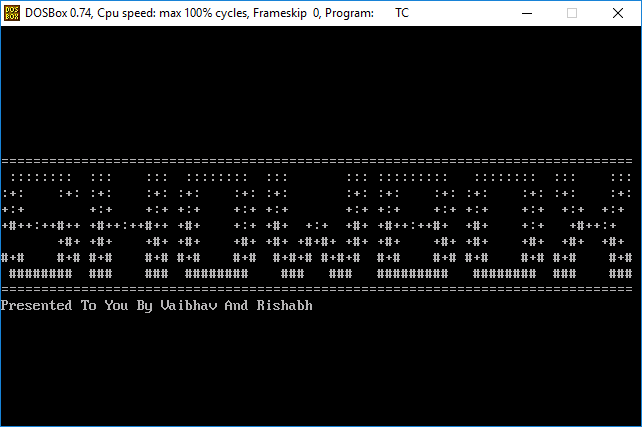
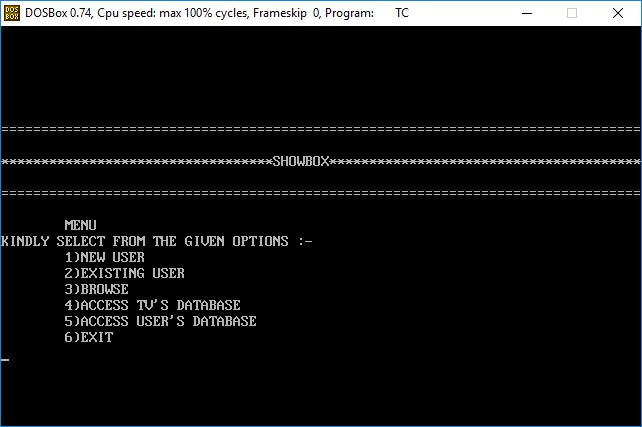
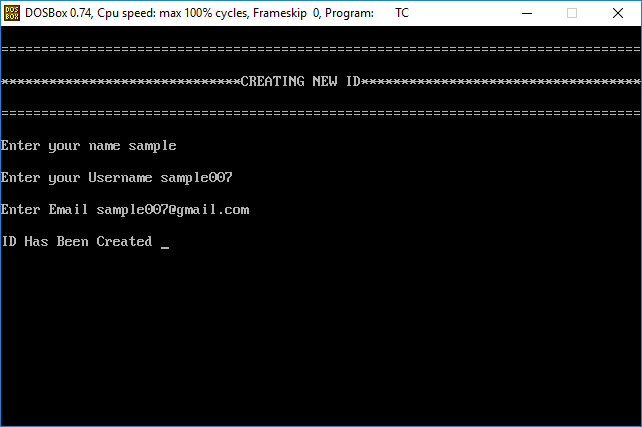
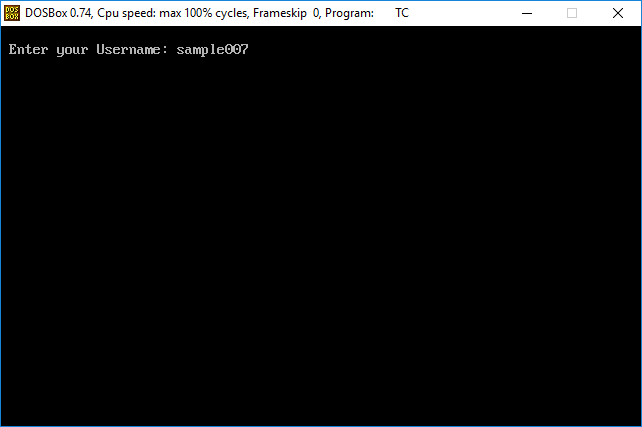
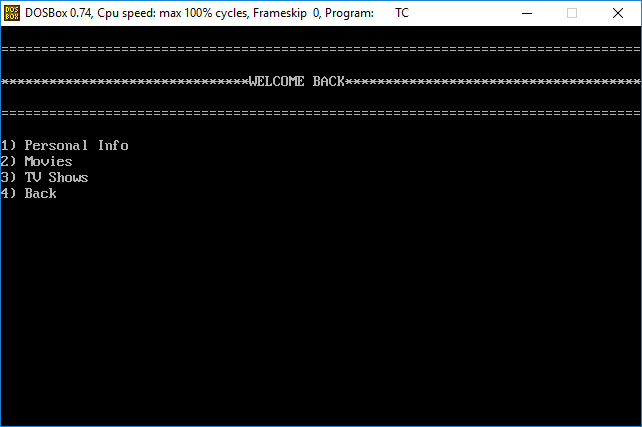
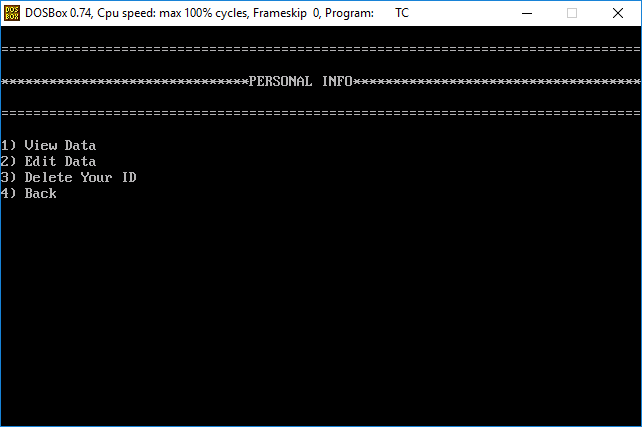
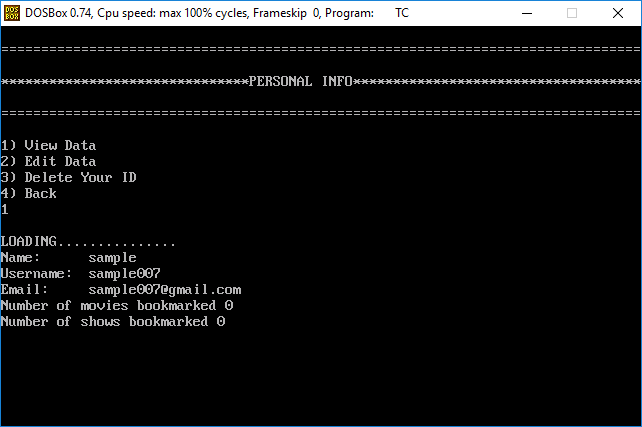
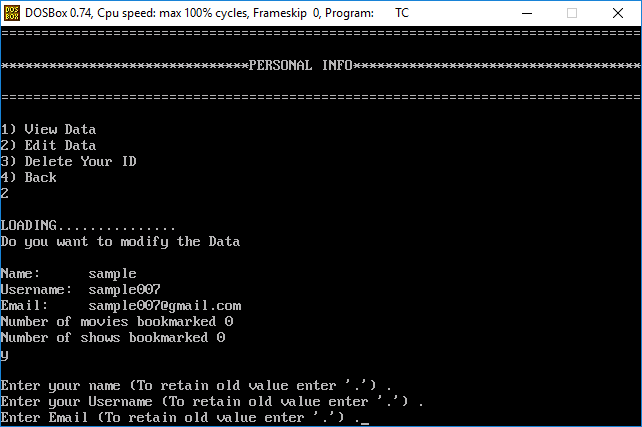
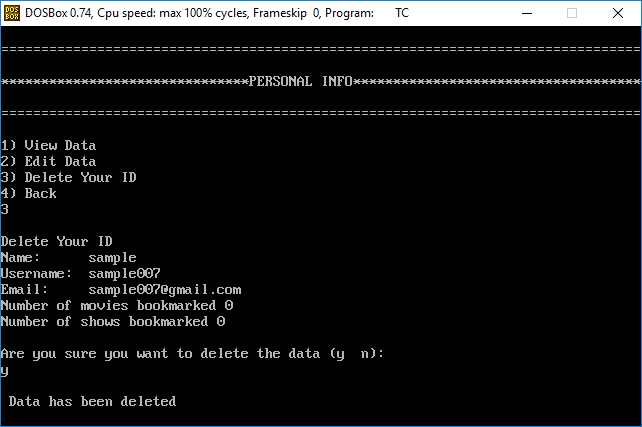
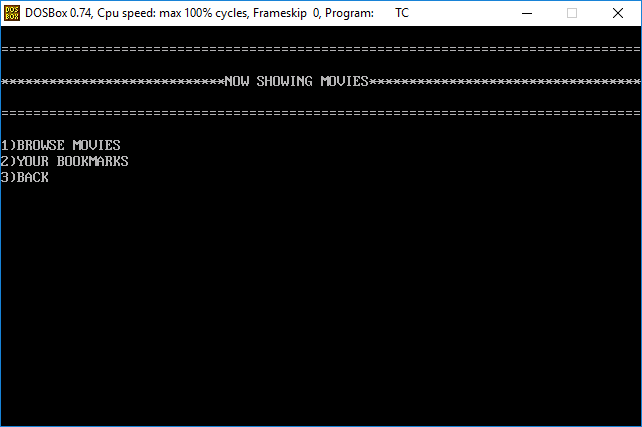
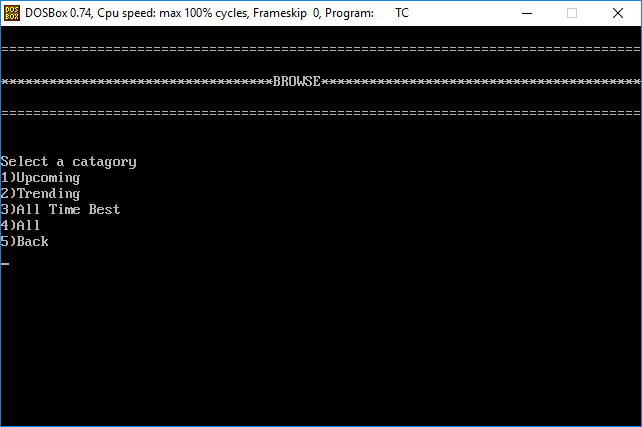
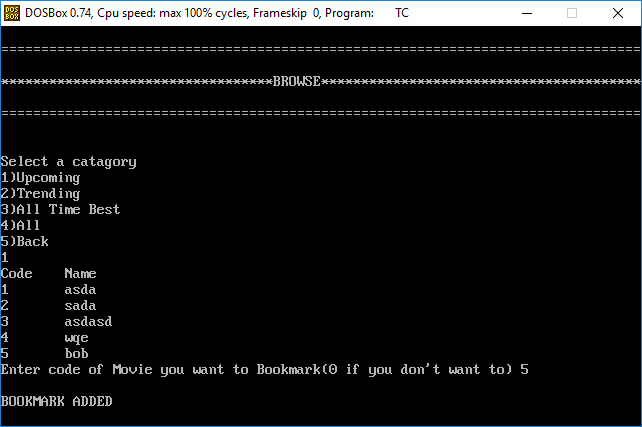
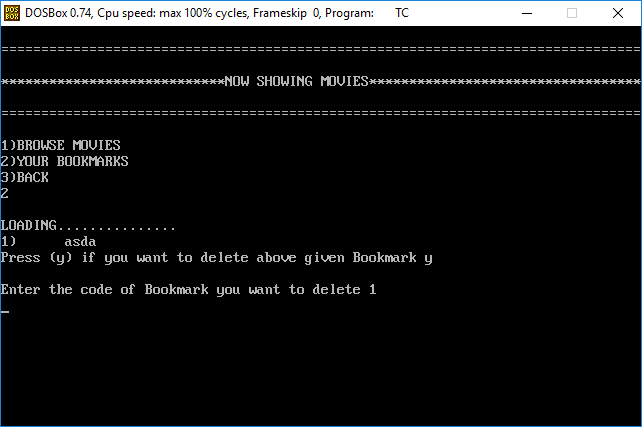
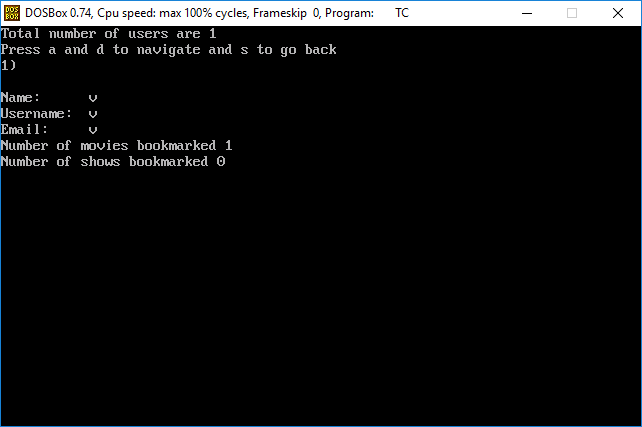
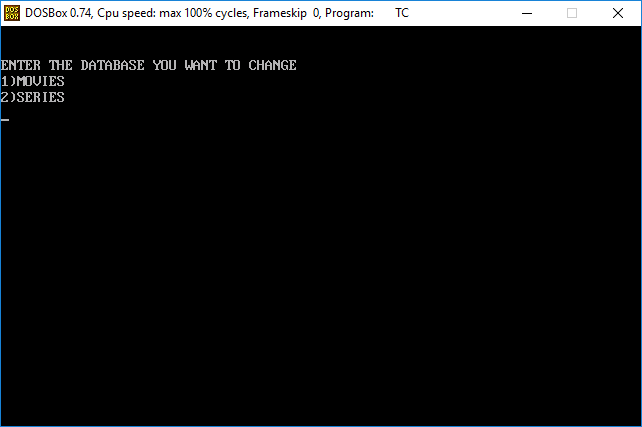
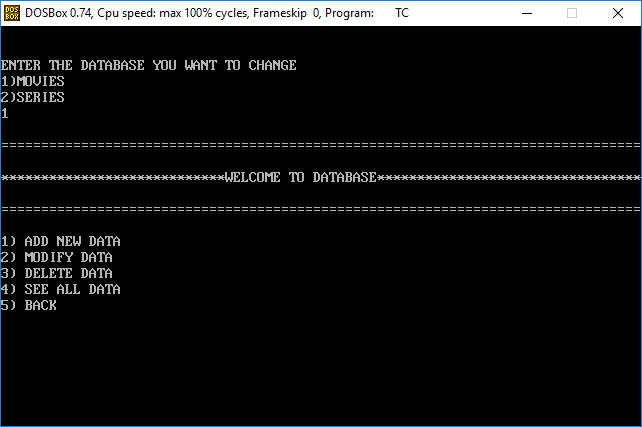
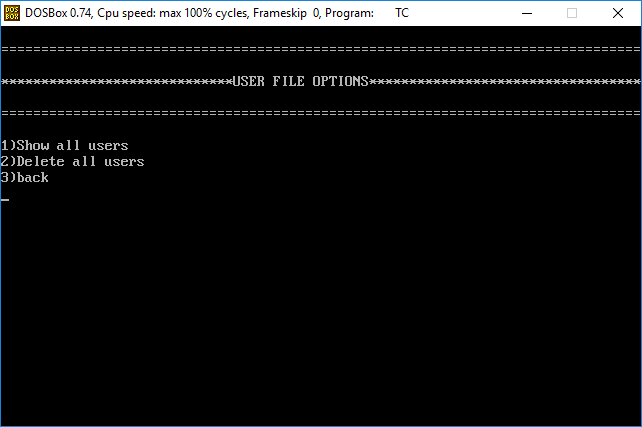
* Menus that depend on datafiles contains options which are dependent on the existence of the required datafile and may be vulnerable to crashes due to which all such functions are disabled unless the file is created on the system by the user.
* Measures have been taken to ensure that an integer field is only receiving integer type from the console no alphabets or follow up garbage values are taken. In the otherwise case, an error message is displayed, and input is re-prompted. Similar process also occurs for strings.

# Coding

|  |
| --- |
| */\* ----------------------------------------------------------------------------------------------------------- |==========================================================================================================| ||=========================================================================================================|| ||=========================================================================================================|| |==========================================================================================================| ----------------------------------------------------------------------------------------------------------- 01010011 01101000 01101111 01110111 01100010 01101111 01111000 \*/* */\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*  *//Header files*  #include<fstream.h>  #include<process.h>  #include<conio.h>  #include<string.h>  #include<stdio.h>  #include<ctype.h>  #include<dos.h> *//Function prototypes*  int  inputchoice();  int  Newuser();  int  Search();  int  TSearch(int);  void Euser();  int  PersonalI(int);  void Movies(int);  void Series(int);  void TModify(int,int);  void TDelete(int,int);  void TAdd(int);  void Show\_Bookmarks(int,int);  void Browse(int,int);  void GBrowse();  void Allshow(int);  int  CheckName(char []);  void BookmarkDelete(int,int,int);  void shift(int,int [],int);  void BDeleteU(int,int);  void Udatabase(); *//=============================================================================* *//Structure For TV  (Both movie and shows)*  struct TV  {  char Tname[20];  int  Tcode;  int  Tcategory[3];  }; *//=============================================================================* *//Class For User*  class User  {  char Uname[20];  char Uemail[40];  char UserID[40];    public:  int SBookmark[20];  int MBookmark[20];  int SNOB;   int MNOB;  void Enter();                *//Function For Entering the Data of User.*  void Show()                  *//Function For Showing the Data of User.*  {  cout<<"\nName:      ";  puts(Uname);  cout<<"Username:  "<<UserID;  cout<<"\nEmail:     "<<Uemail;      cout<<"\nNumber of movies bookmarked "<<MNOB;      cout<<"\nNumber of shows bookmarked "<<SNOB<<endl;   }  char \*ReturnUser()           *//Function For Returning the User ID of User.*  {  return UserID;  }  int  Write();                *//Function For Writing the Data of User on file.*  void Read();                 *//Function For Readinng the Data of User from file .*  void Modify();               *//Function For Modifing the Data of User.*  int  Delete(int);            *//Function For Deleting the Data of User.*  void Add\_Bookmark(int, int, int ); *//Function For Bookmarking .*  User()          *//Constructor*  {  strcpy(Uname,"Null");  strcpy(UserID,"Null");  strcpy(Uemail,"Null");  SNOB=0;  MNOB=0;  }  ~User()            *//Destructor*  {}  }; *//=============================================================================* *//Class For AutoGenerating Code*  class tv\_config {   public:  int Code\_Movies;  int Code\_Shows;  tv\_config()  {  Code\_Movies=1;  Code\_Shows=1;   }  ~tv\_config()  {   }  }; *//=============================================================================* *//Function For Checking Name*  int CheckName( char name[20])  {  int flag=0;  if(name[0]==' ')flag=1;  for(int i=0;i<strlen(name);i++)  {  if((name[i]!=' ') && !(isalpha(name[i])))  flag=1;  }    if(flag==1)    cout<<"\nINVALID NAME PLEASE TRY AGAIN ";     return flag;    } *//=============================================================================* *//Function For Entering the Data of User*  void User::Enter()  {  do  {  cout<<"\nEnter your name ";  gets(Uname);  } while(CheckName(Uname));  cout<<"\nEnter your Username ";  cin>>UserID;  cout<<"\nEnter Email ";  cin>>Uemail;  } *//=============================================================================* *//Function For Writing the Data of User from the file*  int User::Write()  {  User C,u;  ifstream fin;  int flag=0;  char ans;  fin.open("User.dat",ios::binary);  lb:   fin.seekg(0,ios::beg);  u.Enter();  while(fin.read((char\*)&C,sizeof(C)))  {  if (strcmp(C.ReturnUser(),u.ReturnUser())==0)  {  cout<<"\nEntered Username already exist";  cout<<"\nDo you want to try again(T) or open the Existing ID(E)";  cin>>ans;  if(ans=='T'||ans=='t')  goto lb;  else  { fin.close();  flag=1;  return flag;  }  }  }  fin.close();  if(flag!=1)  {  u.MNOB=0;  u.SNOB=0;  cout<<"\nID Has Been Created ";  ofstream fout;  fout.open("User.dat",ios::binary|ios::ate);  fout.write((char\*)&u,sizeof(u));  fout.close();  }  return -1;  } *//=============================================================================* *//Function For Reading the Data of User from the file*  void User::Read() { User n;   int c=0;   ifstream fin;   char Ch;           long pos;           fin.open("User.dat",ios::binary);           if(!fin)         {         cout<<"\n File not found";             getch();   }   else{   while(fin.read((char\*)&n,sizeof(n)))   c++;   cout<<fin.tellg()/sizeof(n)<<endl;   fin.seekg(0, ios::end); *// seek to the end of the file*   int eofOffset=fin.tellg(); *// store the offset of the end-of-file*    fin.seekg(0,ios::beg);     fin.clear();  do   {clrscr();       cout<<"Total number of users are "<<c<<endl;       cout<<"Press a and d to navigate and s to go back"<<endl;   fin.read((char\*)&n,sizeof(n));   pos=fin.tellg();   cout<<float(fin.tellg()/sizeof(n))<<")"<<endl;    n.Show();     Ch=getch();    if(Ch=='a'||Ch=='A')     {     if(pos==(sizeof(n)))       fin.seekg(eofOffset-(1)\*(sizeof(n)),ios::beg);   else   fin.seekg(pos-(2)\*(sizeof(n)),ios::beg);   }   else if(Ch=='d'||Ch=='D')   {   if(pos==eofOffset)   fin.seekg(0,ios::beg);   }   else   fin.seekg(pos-(1)\*(sizeof(n)),ios::beg);  }while(!(Ch=='s'||Ch=='S'));  fin.close();} } *//=============================================================================* *//Function For Modifying the Data of User*  void User::Modify()   {User u;  char ch='n';   char str[40];  ifstream fcheck;   fcheck.open("User.dat",ios::binary);   cout <<"\nDo you want to modify the Data " <<endl;  Show();  cin>>ch;  if(ch=='y'||ch=='Y')  {  do  {  cout<<"\nEnter your name (To retain old value enter '.') ";  gets(str);  if(strcmp(str,".")==0)  break;  }while(CheckName(str));  if(strcmp(str,".")!=0)  strcpy(Uname,str);       lb:   fcheck.seekg(0,ios::beg);        cout<<"Enter your Username (To retain old value enter '.') ";       cin>>str;           while(fcheck.read((char\*)&u,sizeof(u))){       if(strcmp(u.ReturnUser(),str)==0)       {  cout<<"This Username Already Exists Please Enter Again "<<endl;         goto lb;       }     }   if(strcmp(str,".")!=0)        strcpy(UserID,str);  cout<<"Enter Email (To retain old value enter '.') ";  cin>>str;  if(strcmp(str,".")!=0)        strcpy(Uemail,str);  }   } *//=============================================================================* *//Function For Deleting the Data of User*  int User::Delete(int pos)  {  User C;  int loc,flag=0;  ifstream fin;  fin.open("User.dat",ios::binary|ios::beg);  ofstream fout;  fout.open("Temp.dat",ios::binary);  while(fin.read((char\*)&C,sizeof(C)))  {  loc=fin.tellg()-(sizeof(C));  if (pos==loc)  {  char ans;  C.Show();  lb:  cout<<"\nAre you sure you want to delete the data (y \ n):"<<endl;  cin>>ans;   if (ans=='n')  fout.write((char\*)&C,sizeof(C));  else if (ans=='y')  flag=1;  else  {  cout<<"\nInvalid Answer ";  goto lb;  }  }  else  fout.write((char\*)&C,sizeof(C));  }  fin.close();  fout.close();  remove("User.dat");  rename("Temp.dat","User.dat");  if (flag==1)  {  cout<<"\n Data has been deleted ";  getch();  return 4;  }  return 1;   } *//=============================================================================* *//Function For Creating new ID* int Newuser()  {  clrscr();  User C;  int flag;    cout<<"\n================================================================================";   cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CREATING NEW ID\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";   cout<<"\n================================================================================";  flag=C.Write();  getch();  if(flag==1);  {  return 2;  }   } *//=============================================================================* *//Function To search for a record using Username*  int Search()  {  int pos;  ifstream fin;  fin.open("User.dat",ios::binary);  if(!fin)  {  cout<<"\n File not found";      getch();      return -1;  }  User u1;  char uname[40];   cout<<"\n Enter your Username: ";  gets(uname);  while(fin.read((char\*)&u1,sizeof(u1)))  {  if(strcmpi(u1.ReturnUser(),uname)==0)  {  pos=fin.tellg()-(sizeof(u1));  return pos;  }  }  fin.close();  cout<<"\n";   return -1;   } *//=============================================================================* *//Function Of Personal ID access*  int PersonalI(int pos)  {  User U;  int ch;  fstream fio;  fio.open("User.dat",ios::binary|ios::out|ios::in);  do  {  clrscr();  fio.seekg(pos);  fio.read((char\*)&U,sizeof(U));      cout<<"\n================================================================================";      cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PERSONAL INFO\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";      cout<<"\n================================================================================";   cout<<"\n1) View Data ";  cout<<"\n2) Edit Data ";  cout<<"\n3) Delete Your ID ";  cout<<"\n4) Back "<<endl;  ch=inputchoice();  switch(ch)  {  case 1:cout<<"\nLOADING...............";        delay(500);         U.Show();              getch();        break;  case 2:cout<<"\nLOADING...............";        delay(500);              U.Modify();        fio.seekp(pos);        fio.write((char\*)&U,sizeof(U));        cout<<"\nData modified";        break;  case 3:cout<<"\nDelete Your ID ";        ch=U.Delete(pos);      if(ch==4)      return ch;      break;  case 4:cout<<"\nBack ";        break;  default :cout<<"\nWrong Choice";  };   }while(ch!=4);  fio.close();  return 0;  } *//=============================================================================* *//Function of Existing User*  void Euser()  {  clrscr();  int pos,ch;  char uid[40];  pos=Search();  if(pos==-1)  {  cout<<"\The Username Doesn't Exist ";  getch();  }  else  {  do  {  clrscr();       cout<<"\n================================================================================";  cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*WELCOME BACK\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";  cout<<"\n================================================================================";   cout<<"\n1) Personal Info" ;  cout<<"\n2) Movies ";  cout<<"\n3) TV Shows";  cout<<"\n4) Back"<<endl;  ch=inputchoice();;  switch(ch)  {  case 1 :cout<<"\nLOADING...............";  delay(500);  ch=PersonalI(pos);  break;  case 2 :cout<<"\nLOADING...............";  delay(500);  Movies(pos);  break;  case 3 :cout<<"\nLOADING...............";  delay(500);  Series(pos);  break;  case 4 :break;  default :cout<<"\nWrong Choice";  getch();  };  }while(ch!=4);  }  } *//=============================================================================* *//Function for Adding Movies Or TV Series*  void Tadd(int a)  {  ifstream fin;  fin.open("Tconfig.dat",ios::binary|ios::beg);     TV T;  tv\_config c;  fin.read((char\*)&c,sizeof(c));  fin.close();  char y;  ofstream fout;  if(a==2)  {  fout.open("Shows.dat",ios::binary|ios::ate);  c.Code\_Shows++;  T.Tcode=c.Code\_Shows;  }  else if(a==1)  {  fout.open("Movies.dat",ios::binary|ios::ate);  c.Code\_Movies++;  T.Tcode=c.Code\_Movies;  }   cout<<"\n================================================================================";  cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ENTERING DATA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";  cout<<"\n================================================================================";    lb:  cout<<"\nName ";  gets(T.Tname);   if(T.Tname[0]==' ')   { clrscr();     cout<<"Entered name is invalid ";     goto lb;    }   cout<<"Do you want to add it in the category Upcoming " ;  cin>>y;  if(y=='y'||y=='Y')  T.Tcategory[0]=1;  else  T.Tcategory[0]=0;  cout<<"Do you want to add it in the category Trending " ;  cin>>y;  if(y=='y'||y=='Y')  T.Tcategory[1]=1;  else  T.Tcategory[1]=0;  cout<<"Do you want to add it in the category All time best " ;  cin>>y;  if(y=='y'||y=='Y')  T.Tcategory[2]=1;  else  T.Tcategory[2]=0;  fout.write((char\*)&T,sizeof(T));  fout.close();  fout.open("Tconfig.dat",ios::binary|ios::trunc|ios::ate);  fout.write((char\*)&c,sizeof(c));  fout.close();  } *//=============================================================================* *//Function To search for a TV record*  int TSearch(int a)  {  int pos;   ifstream fin;  if(a==2)  fin.open("Shows.dat",ios::binary|ios::beg);  else if(a==1)  fin.open("Movies.dat",ios::binary);   if(!fin)   {     cout<<"\n File not found";     getch();     return -1;   }   TV T;  int flag=0,code;  cout<<"\nEnter Movie/Series Code ";  code=inputchoice();  while(fin.read((char\*)&T,sizeof(T)))  {  if(T.Tcode==code)  {  flag=1;  pos=fin.tellg()-(sizeof(T));  return pos;  }  }  fin.close();  cout<<"\n";   return -1;  } *//=============================================================================* *//Function for Modifying TV's Data*  void TModify(int pos,int a)  {  TV T;  char ch='n',y;  fstream fio;  if(a==2)  fio.open("Shows.dat",ios::binary|ios::in|ios::out|ios::ate);  else if(a==1)  fio.open("Movies.dat",ios::binary|ios::in|ios::out|ios::ate);  fio.seekg(pos);  fio.read((char\*)&T,sizeof(T));   cout<<"\n================================================================================";  cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*MODIFY\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";  cout<<"\n================================================================================";   cout<<"\nName "<<T.Tname<<"\nCode "<<T.Tcode;  for(int i=0;i<3;i++)  cout<<"\nCategory "<<i+1<<" "<<T.Tcategory[i];  cout << "\nDo you want to modify the Data ";  cin>>ch;  if (ch=='y'||ch=='Y')  {  cout<<"\nModify The Data: ";     lb:     cout<<"\nName ";     gets(T.Tname);     if(T.Tname[0]==' ')     { clrscr();       cout<<"Entered name is invalid ";       goto lb;      }   cout<<"Do you want to add it in the category Upcoming " ;  cin>>y;  if(y=='y')  T.Tcategory[0]=1;  else  T.Tcategory[0]=0;  cout<<"Do you want to add it in the category Trending " ;  cin>>y;  if(y=='y')  T.Tcategory[1]=1;  else  T.Tcategory[1]=0;  cout<<"Do you want to add it in the category All time best " ;  cin>>y;  if(y=='y')  T.Tcategory[2]=1;  else  T.Tcategory[2]=0;  fio.seekp(pos);  fio.write((char\*)&T,sizeof(T));  fio.close();  cout<<"\nData modified ";  }   } *//=============================================================================* *//Function for Deleting TV's Data*  void TDelete(int pos,int a)  {  TV T;  int loc,flag=0,code;  ifstream fin;  if(a==2)  fin.open("Shows.dat",ios::binary|ios::beg);  else if(a==1)  fin.open("Movies.dat",ios::binary|ios::beg);  ofstream fout;  fout.open("Temp.dat",ios::binary);   cout<<"\n================================================================================";  cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DELETE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";  cout<<"\n================================================================================";   while(fin.read((char\*)&T,sizeof(T)))  {  loc=fin.tellg()-(sizeof(T));  if (pos==loc)  {code=T.Tcode;  char ans;  cout<<"\nName "<<T.Tname<<"\nCode "<<T.Tcode;  cout<<"\nEnter (y) if you want to delete the data: "<<endl;  cin>>ans;  if(ans=='y'||ans=='Y')  flag=1;  else  fout.write((char\*)&T,sizeof(T));  }  else  fout.write((char\*)&T,sizeof(T));  }  fin.close();  fout.close();  if (flag==1)  {     cout<<"\n Data has been deleted ";     BDeleteU(code,(a-1));  }   if(a==2)  {  remove("Shows.dat");  rename("temp.dat","Shows.dat");  }  if(a==1)  {  remove("Movies.dat");  rename("Temp.dat","Movies.dat");  }   }  *//=============================================================================* *//Function for Acessing  TV's Database*   void Talter()  {  TV T;  int ch,a;  int pos;  do  {    lb:    clrscr();    cout<<"\n\nENTER THE DATABASE YOU WANT TO CHANGE ";    cout<<"\n1)MOVIES";    cout<<"\n2)SERIES\n";    a=inputchoice();    if(a!=1 && a!=2)    {      cout<<"\nWRONG INPUT PLEASE ENTER AGAIN ";      getch();      goto lb;    }    cout<<"\n================================================================================";    cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*WELCOME TO DATABASE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";    cout<<"\n================================================================================";    cout<<"\n1) ADD NEW DATA" ;    cout<<"\n2) MODIFY DATA ";    cout<<"\n3) DELETE DATA ";    cout<<"\n4) SEE ALL DATA ";    cout<<"\n5) BACK"<<endl;    ch=inputchoice();;    switch(ch)    {      case 1:cout<<"\nLOADING...............";             delay(500);             clrscr();             Tadd(a);             break;      case 2:cout<<"\nLOADING...............";             delay(500);             clrscr();             pos=TSearch(a);             if(pos== -1)             {          cout<<"\nNO DATA WAS FOUND ";          ch=5;          cout<<"\nGOING BACK TO MAIN MENU ";          getch();             }             else             {          clrscr();          TModify(pos,a);             }             break;      case 3:cout<<"\nLOADING...............";             delay(500);             clrscr();             pos=TSearch(a);             if(pos== -1)             {          cout<<"\nNO DATA WAS FOUND ";          ch=5;          cout<<"\nGOING BACK TO MAIN MENU ";          getch();             }             else             {          clrscr();          TDelete(pos,a);             }             break;      case 4:cout<<"\nLOADING...............";             delay(500);             clrscr();             Allshow(a);             break;      case 5:break;      default :cout<<"\nWRONG CHOICE";    };  }while(ch!=5);  } *//=============================================================================* *//Function for Movies* void Movies(int pos) {  int ch;  User u1;  do  {    clrscr();    cout<<"\n================================================================================";    cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*NOW SHOWING MOVIES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";    cout<<"\n================================================================================";    cout<<"\n1)BROWSE MOVIES ";    cout<<"\n2)YOUR BOOKMARKS ";    cout<<"\n3)BACK "<<endl;    ch=inputchoice();    switch (ch)    {      case 1:cout<<"\nLOADING...............";             delay(500);             Browse(1,pos);             break;      case 2:cout<<"\nLOADING...............";             delay(500);             Show\_Bookmarks(1,pos);             break;      case 3:break;      default:cout<<"Invalid choice";    };  }while(ch!=3); }*//=============================================================================* *//Function for Series* void Series(int pos) {  int ch;  User u1;  do  {    clrscr();    cout<<"\n================================================================================";    cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*NOW SHOWING SERIES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";    cout<<"\n================================================================================";    cout<<"\n1)BROWSE SERIES ";    cout<<"\n2)YOUR BOOKMARKS ";    cout<<"\n3)BACK "<<endl;    ch=inputchoice();;    switch (ch)    {      case 1:cout<<"\nLOADING...............";             delay(500);             Browse(0,pos);             break;      case 2:cout<<"\nLOADING...............";             delay(500);             Show\_Bookmarks(0,pos);             break;      case 3:break;      default:cout<<"Invalid choice";    };  }while(ch!=3); } *//=============================================================================* *//Function for Browsing*  void Browse(int tv,int pos)  {  int ch,code,i,j,m;  TV T;  User u;  char ans;  do{  i=1;j=0;  clrscr();   cout<<"\n================================================================================";  cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*BROWSE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";  cout<<"\n================================================================================";   cout<<"\n\nSelect a catagory ";  cout<<"\n1)Upcoming ";  cout<<"\n2)Trending ";  cout<<"\n3)All Time Best ";   cout<<"\n4)All ";   cout<<"\n5)Back "<<endl;  ch=inputchoice();;    if(ch>5||ch<1)   {     cout<<"Invalid choice";     getch();     continue;   }  ifstream fin;  if (tv==1)  fin.open("Movies.dat",ios::binary|ios::beg);  else if (tv==0)  fin.open("Shows.dat",ios::binary|ios::beg);     if(!fin)  {  cout<<"\n File not found";       getch();       break;  } cout<<"Code\tName ";  while (fin.read((char\*)&T,sizeof(T)))  {  if (T.Tcategory[ch-1]==1)  {  cout<<"\n"<<(i++)<<"\t"<<T.Tname;  }     if(ch==4&&ch!=5)     {       cout<<"\n"<<(i++)<<"\t"<<T.Tname;     }   } if(ch!=5){  if (tv==1)  cout<<"\nEnter code of Movie you want to Bookmark(0 if you don't want to) ";  else if (tv==0)  cout<<"\nEnter code of Show you want to Bookmark(0 if you don't want to) ";  code=inputchoice();}  fin.close();   if(code>=i||code<0)     cout<<"Invalid choice";   else if(code!=0)  {  if (tv==1)  fin.open("Movies.dat",ios::binary|ios::beg);  else if (tv==0)  fin.open("Shows.dat",ios::binary|ios::beg);  while (fin.read((char\*)&T,sizeof(T)))  { if(ch==4)       {         j++;         if(j==(code))           m=T.Tcode;           break;         }      else if (T.Tcategory[ch-1]==1)        {  j++;  if(j==(code))  m=T.Tcode;           break;            }    }  u.Add\_Bookmark(m,pos,tv);     fin.close();     ch=5;  }   getch();  }while(ch!=5); }  *//=============================================================================* *//Function for Adding Bookmarks*  void User::Add\_Bookmark(int code,int pos,int tv)  {  User u1;  int flag=1;  fstream fio;  fio.open("User.dat",ios::binary|ios::in|ios::out|ios::ate);  fio.seekg(pos);  fio.read((char\*)&u1,sizeof(u1));  if (tv==0)  {  for(int i=0;i<u1.SNOB;i++)  if(u1.SBookmark[i]==code)  flag=0;  if(flag==0)  cout<<"\nAlready Bookmarked";  else  {  u1.SBookmark[u1.SNOB]=code;  u1.SNOB++;       cout<<"\nBOOKMARK ADDED ";  getch();   }  }  else if (tv==1)  {  for(int i=0;i<u1.MNOB;i++)  if(u1.MBookmark[i]==code)  flag=0;  if(flag==0)  cout<<"\nAlready Bookmarked";  else  {  u1.MBookmark[u1.MNOB]=code;  u1.MNOB++;       cout<<"\nBOOKMARK ADDED ";  getch();   }   }  fio.seekp(pos);  fio.write((char\*)&u1,sizeof(u1));  fio.close();  } *//=============================================================================* *//Function for Showing Added Bookmarks*  void Show\_Bookmarks(int tv,int pos) {  int NOB,j=1,code;  char ans;  User u1;  TV T;  ifstream fin,f;  fin.open("User.dat",ios::binary);  fin.seekg(pos);  fin.read((char\*)&u1,sizeof(u1));  if (tv==0)  {  NOB=u1.SNOB;  f.open("Shows.dat",ios::binary);   }  else if (tv==1)  {  NOB=u1.MNOB;  f.open("Movies.dat",ios::binary);  }  while (f.read((char\*)&T,sizeof(T)))  {  for (int i = 0; i<=NOB ; i++)  {  if (tv==0)  {  if(T.Tcode==u1.SBookmark[i])  cout<<"\n"<<(j++)<<")\t"<<T.Tname;   }  else if (tv==1)  {  if(T.Tcode==u1.MBookmark[i])  cout<<"\n"<<(j++)<<")\t"<<T.Tname;  }   }  }  f.close();  fin.close();  getch();  if(NOB>0)  {  cout<<"\nPress (y) if you want to delete above given Bookmark ";  cin>>ans;  if(ans=='y'||ans=='Y')  {  cout<<"\nEnter the code of Bookmark you want to delete";  code=inputchoice();  if(code>0 && code<j)  {  if (tv==0)  {  BookmarkDelete(u1.SBookmark[code-1],pos,1);   }  else if (tv==1)  {  BookmarkDelete(u1.MBookmark[code-1],pos,0);  }  }  else  cout<<"Invalid choice";  }  }   else if(NOB==0)   cout<<"\nNo bookmarks yet"<<endl;  getch();  } *//=============================================================================* *//Function for Deleting Bookmark for a single user*  void BookmarkDelete(int code,int userpos,int tv) {  User u;  ifstream fin;  fin.open("User.dat",ios::binary);  fin.seekg(userpos);  fin.read((char\*)&u,sizeof(u));  fin.close();  if(tv==0)  {  for(int i=0;i<=u.MNOB;i++)  if(u.MBookmark[i]==code)  shift(i,u.MBookmark,u.MNOB);  u.MNOB--;  }  else if(tv==1)  {  for(int i=0;i<=u.SNOB;i++)  if(u.SBookmark[i]==code)  shift(i,u.SBookmark,u.SNOB);  u.SNOB--;  }  ofstream fout;  fout.open("User.dat",ios::binary);  fout.seekp(userpos);  fout.write((char\*)&u,sizeof(u));  fout.close();  } *//=============================================================================* *//Function For overwriting deleted bookmark*  void shift(int index,int arr[],int size) {   for(int j=index;j<size;j++)   arr[j]=arr[j+1]; }  *//=============================================================================* *//Function For Deleting Bookmarks for all user*  void BDeleteU(int code,int tv )  {  int pos;  User u;  ifstream fin;  fin.open("User.dat",ios::binary);  while (fin.read((char\*)&u,sizeof(u)))  {  pos=(fin.tellg())-(sizeof(u));  BookmarkDelete(code,pos,tv);  }  fin.close();  } *//=============================================================================* *//Function For Guest Browse*  void GBrowse()  {  int ch,code,i;  TV T;  char ans,tv;  ifstream fin;    clrscr();     lb:  cout<<"\nDo you want to See Movies or Series (m/s) ";  cin>>tv;     if(tv!='m'&&tv!='s'&&tv!='M'&&tv!='S')     {clrscr();       cout<<"Wrong choice"<<endl;     goto lb;}     do    {       i=1;  clrscr();     lb2:     cout<<"Select a catagory";  cout<<"\n1)Upcoming";  cout<<"\n2)Trending";  cout<<"\n3)All Time Best";     cout<<"\n4)All ";     cout<<"\n5)Back";  ch=inputchoice();;  if(ch<1 || ch>5)   {clrscr();           cout<<"Wrong choice"<<endl;         goto lb2;}  if (tv=='m'||tv=='M')  {  fin.open("Movies.dat",ios::binary|ios::beg);  }  else if (tv=='s'||tv=='S')  {  fin.open("Shows.dat",ios::binary|ios::beg);  }     if(!fin)  {  cout<<"\n File not found";       getch();       ch=5;       break;  }    if(ch!=5)    {      cout<<"\nNAME\tCODE";    }   while (fin.read((char\*)&T,sizeof(T)))  {  if (T.Tcategory[ch-1]==1)  {  cout<<"\n"<<(i++)<<"\t"<<T.Tname;  }       if(ch==4)       {         cout<<"\n"<<(i++)<<"\t"<<T.Tname;       }     }   getch();     fin.close();   }while(ch!=5);  } *//=============================================================================* *//Function For Showing All TV's Data*  void Allshow(int a)  {  TV T;  ifstream fin;   cout<<"\n================================================================================";  cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ALL DATA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";  cout<<"\n================================================================================";   if(a==2)  fin.open("Shows.dat",ios::binary|ios::beg);  else if(a==1)  fin.open("Movies.dat",ios::binary|ios::beg);     if(!fin)     {       cout<<"\n File not found";       getch();      }      else{   while(fin.read((char\*)&T,sizeof(T)))  {  cout<<"\nName "<<T.Tname;  cout<<"\nCode "<<T.Tcode;  for(int i=0;i<3;i++)  cout<<"\nCategory "<<i+1<<" "<<T.Tcategory[i];  }  fin.close();  getch(); }  } *//=============================================================================* *//Function For Checking Input*  int inputchoice()  {  int c;char excess;  cin>>c;  cin.get(excess);  if(cin.fail()||excess!='\n')  {  cin.clear();  cin.ignore(15000,'\n');  cout<<"\nInvalid input Please try again"<<endl;   c=inputchoice();  }  else  return c;  return -1;  }  *//=============================================================================*  *//Function for Acessing  User's Database* void Udatabase() {   int ch;  User u;   char ans;  do  {  clrscr();     cout<<"\n================================================================================";     cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*USER FILE OPTIONS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";     cout<<"\n================================================================================";   cout<<"\n1)Show all users ";  cout<<"\n2)Delete all users ";  cout<<"\n3)back "<<endl;  ch=inputchoice();;  switch (ch)  {  case 1:u.Read();        break;  case 2:cout<<"\nAre you sure you want to delete all users(y/n)";              cin>>ans;              if(ans=='y'||ans=='Y')              {                remove("User.dat");                cout<<"Data deleted";                fstream f;                f.open("User.dat",ios::binary);                f.close();              }              break;  case 3:break;  default:cout<<"Invalid choice";  };  }while(ch!=3);  } *//=============================================================================* *//Void Main* void main() { clrscr(); int pos,choice,go; char ans; for(int j=0;j<7;j++)cout<<endl; cout<<"\n==============================================================================="     <<"\n ::::::::  :::    :::  ::::::::  :::       ::: :::::::::   ::::::::  :::    :::"     <<"\n:+:    :+: :+:    :+: :+:    :+: :+:       :+: :+:    :+: :+:    :+: :+:    :+:"     <<"\n+:+        +:+    +:+ +:+    +:+ +:+       +:+ +:+    +:+ +:+    +:+  +:+  +:+"     <<"\n+#++:++#++ +#++:++#++ +#+    +:+ +#+  +:+  +#+ +#++:++#+  +#+    +:+   +#++:+"     <<"\n       +#+ +#+    +#+ +#+    +#+ +#+ +#+#+ +#+ +#+    +#+ +#+    +#+  +#+  +#+"     <<"\n#+#    #+# #+#    #+# #+#    #+#  #+#+# #+#+#  #+#    #+# #+#    #+# #+#    #+#"     <<"\n ########  ###    ###  ########    ###   ###   #########   ########  ###    ###"     <<"\n==============================================================================="; cout<<"\nPresented To You By Vaibhav And Rishabh "; getch(); do {   clrscr();   User U;   for(int i=0;i<5;i++)cout<<endl;   cout<<"\n================================================================================";   cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SHOWBOX\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";   cout<<"\n================================================================================";   cout<<"\n\tMENU ";   cout<<"\nKINDLY SELECT FROM THE GIVEN OPTIONS :-";   cout<<"\n\t1)NEW USER";   cout<<"\n\t2)EXISTING USER";   cout<<"\n\t3)BROWSE";   cout<<"\n\t4)ACCESS TV'S DATABASE";   cout<<"\n\t5)ACCESS USER'S DATABASE ";   cout<<"\n\t6)EXIT\n";   choice=inputchoice();   switch(choice)   {     case 1: cout<<"\nLOADING...............";       delay(500);       go=Newuser();       if(go!=2)           break;     case 2: cout<<"\nLOADING...............";       delay(500);       Euser();       break;     case 3: cout<<"\nLOADING...............";       delay(500);       cout<<"\nGuest..............";       GBrowse();       break;     case 4: cout<<"\nLOADING...............";       delay(500);       cout<<"\nModify.............";       Talter();       break;     case 5: cout<<"\nLOADING...............";       delay(500);       clrscr();       cout<<"\nALL USERS.......... ";       Udatabase();       break;     case 6:       cout<<"\n\*\*\*\*\*\*THANKYOU FOR USING OUR PROGRAM\*\*\*\*\*\*";       delay(500);       exit(1);       break;     default:cout<<"\nInvalid Option";       getch();   }; }while(choice!=6); getch(); }  *//=============================================================================* *//=============================THE END========x=================================* |

End of code

# Output

# Problems Encountered

We encountered several problems throughout the development of the program most of which were fixed. We faced problems such as:

* Several runtime errors while developing algorithms for proper data integrity maintenance functions for date and integer datatypes.
* Effectively maintaining a unique ID for each structures’ object.
* Effectively linking class and structure through a meaningful relationship.
* Developing data file handling functions powerful enough to be able to completely utilize the classes.
* Data redundancy regarding username.
* Universal movie and series delete from database and bookmarks.

# Limitations

While the program takes care of most logical errors while providing full functionality with the ability to arrange data as per the users’ need through the grouping mechanism of media , it still has some limitations. A few of which are:

* The files used to perform operations can’t be changed.
* The program can’t perform any data integrity tests on the “E-mail address” string from the “User” class.
* Since the user is allowed to delete certain files, issues regarding compatibility with new data arise as one part of the system may not be updated accordingly on deletion of a file, which basically resets the unique Code system.
* There is no option in *MENU* where the user can view the number of users bookmarked a particular movie or series as and when they are updated.
* If user delete’s his ID then there is no way he can restore it
* There’s no way to check integrity or to authenticate movies and shows

# Extension Possible

* Better graphics could’ve been used to improve the user interface of the program.
* A master search mechanism could’ve been added.
* Linked lists could have been used to provide seemingly no limitations on writing limit.
* Circular queue mechanism could have bee used in different fields for better interface such as in menus and other display functions.

**Bibliography**

This program file has been completed using the help of the following sources:

* Computer Science with C++ (Class 11) By SUMITA ARORA
* Computer Science with C++ (Class 12) By SUMITA ARORA
* Stacks overrflow
* Google

**CD**