```cpp
#ifndef HEADER_H_
#define HEADER_H_
#include <iostream>
#include <string>
#include <fstream>
#include <iomanip>
#include <unistd.h>

struct DVD {
   std::string title;
   std::string leadActor;
   std::string subActor;
   std::string genre;
   std::string altGenre;
   int year;
   int rating;
   std::string synopsis;
   DVD *nextNode;
};
enum MenuOptions { Exit = 0, OutputEntireList, TitleSearch, GenreSearch, ActorSearch, YearSearch, RatingSearch};
void heading();//outputs heading
DVD *readInput();//reads data from file and stores it in heap using linked list
void dispMenu();//displays menu
MenuOptions switchValidation();//validates switch input and returns enum MenuOptions
void searchKeyLogic(
   DVD *head,
   std::fstream &oFile,
   const int &key,
   int &index);//searches for year or rating

void searchKeyLogic(
   DVD *head,
   std::fstream &oFile,
   const std::string &key,
   int &index);//searches for genre or actor

void outputList(
   DVD *head,
   std::fstream &oFile,
   int &index);//outputs entire list to outfile

//testing which format of protype is more used if you can give me some help

void multiMoviePrint(std::fstream &oFile, DVD* node, int &index);//prints multi movies
void switchOption(const MenuOptions &option, DVD* head, std::fstream &oFile);//directs user on there choice
void subString(DVD *head);//formats data
void genreSearch(DVD* head, std::fstream &oFile, int &index);//genre search output
void actorSearch(DVD* head, std::fstream &oFile, int &index);//actor search output
void yearSearch(DVD* head, std::fstream &oFile, int &index);//year search output
void ratingSearch(DVD* head, std::fstream &oFile, int &index);//rating search output
void yearValidation(int &keyInt);//validates year using try catch
void ratingValidation(int &keyInt);//validates rating using try catch
void printSingleMovie(DVD* node, std::fstream &oFile);//prints single movie
void titleSearch(DVD* head, std::fstream &oFile);//title search output
void titleSearchLogic(DVD* head, std::fstream &oFile, std::string
strKey);//searches linked list for title
void deallocate(DVD* head);//dallocates memory

#endif
```

```cpp
/****************************************************************************
 * AUTHOR       : Carlos Aguilera
 * STUDENT ID   : 1152562
 * LAB #3       : Searching Linked Lists
 * CLASS        : CS1B
 * SECTION      : M-W
 * DUE DATE     : 04.05.22
 ***************************************************************************/

#include "../include/header.h"

/***************************************************************************
 * Title: Searching Linked Lists
 * ---------------------------------------------------------------------------
 * PROGRAM:
 *    This Program pulls data from an input file that is specified or it will
 * resort to a default, it is using linked lists to collect that data and
 * store it in the heap. The functionality of the program should be as follows
 * user starts application and selects file to read from then that file if
 * valid gets collected as a linked list. Then you can obtain that data and
 * search through it using sequential search.
 * ---------------------------------------------------------------------------
 * Data Table
 * ----------
 * std::string outputFile IN - used to create or open output file
 * DVD* head IN, CALC & OUT - creates pointer that points to DVD struct
 * MenuOptions option IN & CALC - enum that holds what option they choose
 * std::fstream oFile OUT - used for outputting to a file
 ***************************************************************************/

int main() {
   heading();
   std::string outputFile;
   DVD *head = NULL;
   head = readInput();//reads input file
   subString(head);

   MenuOptions option;
   std::fstream oFile;
   std::cout << "Which output file would you like to use(type d for default
file)? ";
   std::getline(std::cin, outputFile);
   if (outputFile == "d")
      outputFile = "output.txt";
   oFile.open(outputFile, std::ios::out);//opens file to write

   do {
      dispMenu();//displays menu
      option = switchValidation();//validates switch choice before choosing
      switchOption(option, head, oFile);
   } while(option != Exit);

   oFile.close();
   deallocate(head);
   head = NULL;
   return 0;
}
```

```cpp
#include "../include/header.h"
/****************************************************************************
 * Title: dispMenu
 * --------------------------------------------------------------------------
 * FUNCTION:
 *    This function displays the menu
 * --------------------------------------------------------------------------
 * NO Data Table
 * ----------
 ***************************************************************************/

void dispMenu() {
   std::cout << "\nDVD MENU OPTIONS\n\n";
   std::cout << "1 - Output Entire List\n";
   std::cout << "2 - Title Search\n";
   std::cout << "3 - Genre search\n";
   std::cout << "4 - Actor Search\n";
   std::cout << "5 - Year Search\n";
   std::cout << "6 - Rating Search (0 - 9)\n";
   std::cout << "0 - EXIT\n";
}
```

```cpp
#include "../include/header.h"
/*****************************************************************************
 * Title: readInput
 * ---------------------------------------------------------------------------
 * FUNCTION:
 *    This function reads from the input file specified, how it goes about it
 * is as so creates head ptr to DVD and creates node ptr to DVD but allocates
 * a new DVD on the heap. What happens is node = new DVD and head = NULL
 * and first iteration of the while loop the next node equals head which is
 * NULL what that is doing is setting the node to be NULL and last in the
 * linked list. head then has the address of orignal node and node = new DVD
 * ---------------------------------------------------------------------------
 * Data Table
 * ----------
 * DVD* head = NULL IN & CALC - used in linked list
 * DVD* node = new DVD IN & CALC - used in linked list
 * std::fstream inFile IN - used to grab movie data
 * std::string temp IN - used to grab \n
 * std::string inputFile IN & CALC - used to identify input file
 *****************************************************************************/


DVD *readInput() {
   DVD *head = NULL;
   DVD *node = new DVD;
   std::fstream inFile;
   std::string temp;
   std::string inputFile;
   do {
      std::cout << "Which input file would you like to use(type d for default
file)? ";
      std::getline(std::cin, inputFile);
      if (inputFile != "d")
         inputFile = "../build/" + inputFile;//adding folder structure to file
      else
         inputFile = "../build/AS5-BigInFile.txt";//default

      inFile.open(inputFile, std::ios::in);

      if (inFile.is_open()) {
         while (!inFile.eof()) {//while not end of file
            std::getline(inFile, node -> title);
            std::getline(inFile, node -> leadActor);
            std::getline(inFile, node -> subActor);
            std::getline(inFile, node -> genre);
            std::getline(inFile, node -> altGenre);
            inFile >> node -> year >> node -> rating;
            inFile.ignore(10000, '\n');
            std::getline(inFile, node -> synopsis);
            std::getline(inFile, temp);

            node->nextNode = head;
            head = node;
            node = new DVD;
         }
         inFile.close();
      }else
         std::cout << "Enter Valid File name\n";// if no input file exist
   } while (head == NULL);//run while havent been initialized
```

```
    delete node;
    return head;
}
```

```cpp
#include "../include/header.h"
/*****************************************************************************
 * Title: subString
 * ---------------------------------------------------------------------------
 * FUNCTION:
 *    This function formats the data to match all calculations made to it
 * ---------------------------------------------------------------------------
 * NO Data Table
 * ----------
 ****************************************************************************/

void subString(DVD* head) {
   DVD* node = head;
   while (node->nextNode != NULL) {
      //cuts one character off the ends of each string
      node->title = node->title.substr(0, node->title.size() - 1);
      node->leadActor = node->leadActor.substr(0, node->leadActor.size() - 1);
      node->subActor = node->subActor.substr(0, node->subActor.size() - 1);
      node->genre= node->genre.substr(0, node->genre.size() - 1);
      node->altGenre= node->altGenre.substr(0, node->altGenre.size() - 1);
      node->synopsis= node->synopsis.substr(0, node->synopsis.size() - 1);
      node = node->nextNode;
   }
}
```

```cpp
#include "../include/header.h"
/****************************************************************************
 * Title: validateInput
 * --------------------------------------------------------------------------
 * FUNCTION:
 *    This is an input validation function it handles all major validations
 * from the switch statement, year, and rating validation
 * --------------------------------------------------------------------------
 * Data Table
 * ----------
 * bool inputValidated = false; CALC - to determine if input was validated
 * int choice; IN & CALC - input choice and verify if its valid
 ***************************************************************************/

MenuOptions switchValidation() {
   int choice;
   bool inputValidated = false;
   //try catch wrapped in a do while until input is validated
   do {
      try {
         std::cout << "Enter an option (0 to exit): ";
         std::cin >> choice;
         if (std::cin.fail()) {//if cin fails throw expection
            std::cin.clear();//clears buffer for next cin
            std::cin.ignore(10, '\n');
            throw(true);
         }
         inputValidated = true;
      }
      catch(bool invalid) {
         std::cout << "Please enter a number!\n";
      }
   } while (!inputValidated);

   return static_cast <MenuOptions> (choice);//returns int but static cast to enum MenuOptions
}

void yearValidation(int &keyInt) {
   bool inputValidated = false;
   do {
      try {
         std::cout << "\nWhich year are you looking for? ";
         std::cin >> keyInt;
         if (std::cin.fail()) {
            std::cin.clear();
            std::cin.ignore(1000, '\n');
            throw(true);
         }else if (keyInt >= 1878 && keyInt <= 3000)//if key is valid and its betweens years then
            inputValidated = true;
         else {
            std::cout << "The number " << keyInt << " is an invalid entry\n";
            std::cout << "**** Please input a number between 1878 and 3000 ****\n";
         }
      }
      catch(bool invalid) {
         std::cout << "**** Please input a NUMBER between 1878 and 3000 ****\n";
      }
   } while (!inputValidated);
}

void ratingValidation(int &keyInt) {
   bool inputValidated = false;
   do {
      try {
         std::cout << "\nWhich rating are you looking for? ";
         std::cin >> keyInt;
         if (std::cin.fail()) {
            std::cin.clear();
            std::cin.ignore(1000, '\n');
            throw(true);
         }else if (keyInt >= 0 && keyInt <= 9)
            inputValidated = true;
         else {
            std::cout << "The number " << keyInt << " is an invalid entry\n";
            std::cout << "**** Please input a number between 0 and 9 ****\n";
         }
      }
      catch(bool invalid) {
         std::cout << "**** Please input a NUMBER between 0 and 9 ****\n";
      }
   } while (!inputValidated);
}
```

```cpp
#include "../include/header.h"
/*****************************************************************************
 * Title: switchOption
 * ---------------------------------------------------------------------------
 * FUNCTION:
 *    Switch function handles the choice of user
 * ---------------------------------------------------------------------------
 * Data Table
 * ----------
 * int index CALC - used to determine index of linked list
 ****************************************************************************/

void switchOption(const MenuOptions &option, DVD* head, std::fstream &oFile)
{
   int index = 0;
   switch (option) {
      case OutputEntireList:
         std::cout << "\nCOMPLETE MOVIE LISTING!\n\n";
         oFile << "All Movies Found:\n";
         outputList(head, oFile, index);
         oFile << "\n";
         break;
      case TitleSearch:
         titleSearch(head, oFile);
         break;
      case GenreSearch:
         genreSearch(head, oFile, index);
         break;
      case ActorSearch:
         actorSearch(head, oFile, index);
         break;
      case YearSearch:
         yearSearch(head, oFile, index);
         break;
      case RatingSearch:
         ratingSearch(head, oFile, index);
         break;
      case Exit:
         std::cout << "\nThank You!!\n";
         break;
      default:
         std::cout << "Enter A Valid Option\n";
         break;
   }

}
/*****************************************************************************
 * Title: titleSearch
 * ---------------------------------------------------------------------------
 * FUNCTION:
 *    handles the output for title search
 * ---------------------------------------------------------------------------
 * Data Table
 * ----------
 * std::strKey IN & CALC input key and searches linked list for it
 ****************************************************************************/

void titleSearch(DVD* head, std::fstream &oFile) {
   std::string strKey;

   std::cout << "Which Title are you looking for? ";
   std::cin.ignore(1000, '\n');
   std::getline(std::cin, strKey);

   titleSearchLogic(head, oFile, strKey);
}
/************************************************************************
 * Title: genreSearch
 * ----------------------------------------------------------------------
 * FUNCTION:
 *    Handles the output for genreSearch
 * ----------------------------------------------------------------------
 * Data Table
 * ----------
 * std::string strKey IN & CALC - holds user genre input
 ***********************************************************************/

void genreSearch(DVD* head, std::fstream &oFile, int &index) {
   std::string strKey;

   std::cout << "\nWhich Genre are you looking for? ";
   std::cin.ignore(1000, '\n');
   std::getline(std::cin, strKey);
   std::cout << "\nSearching for the genre " << strKey << "\n";

   searchKeyLogic(head, oFile, strKey, index);

   if (index == 0)//if after going through the linked list the index still is
0 then movie was not found
      std::cout << "Sorry, no movies for the genre " << strKey << " were
found.\n";
   else {
      std::cout << "Found " << index << " movies for the genre " << strKey <<
"!\n";
      oFile << "\n";
   }
}
/************************************************************************
 * Title: actorSearch
 * ----------------------------------------------------------------------
 * FUNCTION:
 *    handles actor search output
 * ----------------------------------------------------------------------
 * Data Table
 * ----------
 * std::string strKey IN & CALC - holds actor name
 ***********************************************************************/

void actorSearch(DVD* head, std::fstream &oFile, int &index) {
   std::string strKey;

   std::cout << "\nWhich Actor are you looking for? ";
   std::cin.ignore(1000, '\n');
   std::getline(std::cin, strKey);
   std::cout << "\nSearching for the actor " << strKey << "\n";

   searchKeyLogic(head, oFile, strKey, index);

   if (index == 0)
```

```
      std::cout << "Sorry, no movies for the actor " << strKey << " were found.\n";
   else {
      std::cout << "Found " << index << " movies for the actor " << strKey << "!\n";
      oFile << "\n";
   }
}
/**************************************************************************
 * Title: yearSearch
 * ------------------------------------------------------------------------
 * FUNCTION:
 *    handles year search output
 * ------------------------------------------------------------------------
 * Data Table
 * ----------
 * int keyInt IN & CALC - user year input and then we validate it
 *************************************************************************/

void yearSearch(DVD* head, std::fstream &oFile, int &index) {
   int keyInt;
   yearValidation(keyInt);

   std::cout << "\nSearching for the year " << keyInt << "\n";
   searchKeyLogic(head, oFile, keyInt, index);

   if (index == 0)
      std::cout << "Sorry, no movies for the year " << keyInt << " were found.\n";
   else {
      std::cout << "Found " << index << " movies for the year " << keyInt << "!\n";
      oFile << "\n";
   }
}
/**************************************************************************
 * Title: ratingSearch
 * ------------------------------------------------------------------------
 * FUNCTION:
 *    handles rating search output
 * ------------------------------------------------------------------------
 * Data Table
 * ----------
 * int keyInt IN & CALC - user rating input and then we validate it
 *************************************************************************/

void ratingSearch(DVD* head, std::fstream &oFile, int &index) {
   int keyInt;
   ratingValidation(keyInt);

   std::cout << "\nSearching for the rating " << keyInt << "\n";
   searchKeyLogic(head, oFile, keyInt, index);

   if (index == 0)
      std::cout << "Sorry, no movies for the rating " << keyInt << " were found.\n";
   else {
      std::cout << "Found " << index << " movies for the rating " << keyInt << "!\n";
      oFile << "\n";
```

```cpp
   }
}
/*****************************************************************************
 * Title: outputList
 * ---------------------------------------------------------------------------
 * FUNCTION:
 *    handles entire output of movies
 * ---------------------------------------------------------------------------
 * NO Data Table
 * ----------
 *****************************************************************************/

void outputList(DVD *head, std::fstream &oFile, int &index) {
   DVD *node = head;
   while (node->nextNode != NULL) {

      multiMoviePrint(oFile, node, index);
      node = node->nextNode;
   }
}
```

```cpp
#include "../include/header.h"
/****************************************************************************
 * Title: titleSearchLogic
 * --------------------------------------------------------------------------
 * FUNCTION:
 *    handles the title search logic has a while loop the iterates until
 * end of linked list or until found
 * --------------------------------------------------------------------------
 * Data Table
 * ----------
 * DVD* node = head CALC - use it not to mess with head
 * bool found CALC - false until found
 ***************************************************************************/

void titleSearchLogic(DVD* head, std::fstream &oFile, std::string strKey) {
   DVD *node = head;
   bool found = false;
   while (node->nextNode != NULL && !found) {

      if (node->title == strKey) {//current strKey being a title of course
         std::cout << "Found the movie " << strKey << "!\n";
         printSingleMovie(node, oFile);
         found = true;
      }

      node = node->nextNode;
      if (node->nextNode == NULL)//output statement if not found
         std::cout << "Sorry, the movie \" " << strKey <<  " \" was not 
found.\n";
   }
}
/****************************************************************************
 * Title: searchKeyLogic
 * --------------------------------------------------------------------------
 * FUNCTION:
 *    These are 2 overloaded functions that handle searches for year, rating
 * genre, and lead actor
 * --------------------------------------------------------------------------
 * NO Data Table
 * ----------
 ***************************************************************************/

void searchKeyLogic(DVD *head, std::fstream &oFile, const int &key, int 
&index) {
   DVD *node = head;
   while (node->nextNode != NULL) {

      if (node->year == key) {
         if (index == 0)
            oFile << "Search by year for " << key << " found:\n";
         multiMoviePrint(oFile, node, index);
      } else if (node->rating == key) {
         if (index == 0)
            oFile << "Search by rating for " << key << " found:\n";
         multiMoviePrint(oFile, node, index);
      }

      node = node->nextNode;
   }
}
void searchKeyLogic(DVD *head, std::fstream &oFile, const std::string &key,
int &index) {
   DVD *node = head;
   while (node->nextNode != NULL) {

      if (node->genre == key || node->altGenre == key) {
         if (index == 0)
            oFile << "Search by genre for " << key << " found:\n";
         multiMoviePrint(oFile, node, index);
      } else if (node->leadActor == key || node->subActor == key) {
         if (index == 0)
            oFile << "Search by actor for " << key << " found:\n";
         multiMoviePrint(oFile, node, index);
      }

      node = node->nextNode;
   }
}
```

```cpp
#include "../include/header.h"
/****************************************************************************
 * Title: printSingleMovie
 * --------------------------------------------------------------------------
 * FUNCTION:
 *    Handles word wrap functionality and output to file a single movie
 * --------------------------------------------------------------------------
 * Data Table
 * ----------
 * std::string line OUT - used to output line of text
 * std::string word CALC - used to hold word and add to line
 * const int maxLineLength CALC max length a line can be
 ***************************************************************************/

void printSingleMovie(DVD* node, std::fstream &oFile) {
   std::string line;
   std::string word;
   const int maxLineLength = 75;
   oFile << std::left;
   oFile << "***************************************************************************\n";
   oFile << "Title: " << node->title << "\n";
   oFile << "---------------------------------------------------------------------------\n";
   oFile << "Year: " << node->year << "        " << "Rating: " << node->rating << "\n";
   oFile << "---------------------------------------------------------------------------\n";
   oFile << std::setw(18) << "Leading Actor:" << std::setw(25) << node->leadActor << "Genre 1: " << node->genre << "\n";
   oFile << "Supporting Actor: " << std::setw(25) << node->subActor << "Genre 2: " << node->altGenre << "\n";
   oFile << "---------------------------------------------------------------------------\n";
   oFile << "PLOT:\n";
   for (int i = 0; i < node->synopsis.length(); i++) {//logic for word wrap
      //runs character by character if its a space then its ignored if not its added to word once word is completed it adds it to line
      //if line length is greater than max then we output line
      if (node->synopsis.at(i) != ' ')
         word.push_back(node->synopsis.at(i));
      else if (word.length() + line.length() > maxLineLength) {
         oFile << line << "\n";
         line.clear();
         line += word;
         word.clear();
      }else {
         line += word + ' ';
         word.clear();
      }
      if (i + 1 == node->synopsis.length()) {
         line += word;
         oFile << line << "\n";
      }
   }
   oFile << "***************************************************************************\n\n";
```

```
    oFile << std::right;
}
```

```cpp
#include "../include/header.h"
/*****************************************************************************
 * Title: multiMoviePrint
 * ---------------------------------------------------------------------------
 * FUNCTION:
 *    handles output of multiple movies also handles greater than format
 * amount feature and cuts it accordingly
 * ---------------------------------------------------------------------------
 * NO Data Table
 * ----------
 *****************************************************************************/

 void multiMoviePrint(std::fstream &oFile, DVD* node, int &index) {
   ++index;
   oFile << std::left;
   if (index <= 10) {
      if(index == 1) {
         oFile << "MOVIE #"  << std::setw(50) << "  TITLE" << "YEAR " <<
"RATING  "  << std::setw(18) << "GENRE" << std::setw(18) << "ALT GENRE" <<
std::setw(20) << "LEAD ACTOR" << "SUPPORTING ACTOR\n";
         oFile << "------- ------------------------------------------------ --
-- ------ ----------------- ----------------- ------------------- ------------
-------\n";
      }
      oFile << "   " << std::setw(6) << index  << std::setw(48);

      if (node->title.size() > 47) {//if title is greater than format space
         oFile << (node->title.substr(0, 44)) + "..."; //we cut and add ...
      }else
         oFile << node->title;

      oFile << std::setw(8) << node->year << std::setw(5) << node->rating <<
std::setw(18) << node->genre << std::setw(18) << node->altGenre <<
std::setw(20);

      if (node->leadActor.size() > 18) {
         oFile << (node->leadActor.substr(0, 15)) + "...";
      }else
         oFile << node->leadActor;

      if (node->subActor.size() > 18) {
         oFile << (node->subActor.substr(0, 15)) + "...";
      }else
         oFile << node->subActor;
      oFile << "\n";
      if (index == 10)
         oFile << "...\n";
   }

   oFile << std::right;
}
```

```
#include "../include/header.h"

void deallocate(DVD* head) {
   DVD* node = head;
   while(node != NULL) {
      head = node->nextNode;
      delete node;
      node = head;
   }
}
```

```cpp
#include "../include/header.h"

void heading() {

   /***********************************************************************
    * CONSTANTS
    * ---------------------------------------------------------------------
    * OUTPUT - USED FOR CLASS HEADING
    * ---------------------------------------------------------------------
    * PROGRAMMER : Programmer's Name
    * CLASS      : Student's Course
    * SECTION    : Class Days and Times
    * LAB_NUM    : Lab Number (specific to this lab)
    * LAB_NAME   : Title of the Lab
    **********************************************************************/
   const char PROGRAMMER[] = "Carlos Aguilera";
   const char CLASS[]      = "CS1B";
   const char SECTION[]    = "MW: 7:30p - 9:50p";
   const int LAB_NUM       = 3;
   const char LAB_NAME[]   = "Searching Linked Lists";

   std::cout << std::left;
   std::cout << "****************************************************\n";
   std::cout << "*   PROGRAMMED BY : " << PROGRAMMER << std::endl;
   std::cout << "*   " << std::setw(14) <<"CLASS" << ": " << CLASS << std::endl;
   std::cout << "*   " << std::setw(14) <<"SECTION" << ": " << SECTION << std::endl;
   std::cout << "*   LAB #" << std::setw(9) << LAB_NUM << ": " << LAB_NAME << std::endl;
   std::cout << "****************************************************\n\n";
   std::cout << std::right;
}
```

```
All Movies Found:
MOVIE #  TITLE                                           YEAR RATING  GENRE
          ALT GENRE         LEAD ACTOR          SUPPORTING ACTOR
------- ------------------------------------------------ ---- ------ ---------
-------- ----------------- ------------------- -------------------
   1     007 - Casino Royale                             2006    8    Action
         Adventure         Daniel Craig        Eva Green
   2     007 - Quantum of Solace                         2008    7    Action
         Adventure         Daniel Craig        Olga Kurylenko
   3     10 Items or Less                                2006    7    Comedy
         Drama             Morgan Freeman      Paz Vega
   4     15 minutes                                      2001    6    Action
         Crime             Robert De Niro      Edward Burns
   5     17 Again                                        2009    7    Comedy
         Comedy            Zac Effron          Leslie Mann
   6     21                                              2008    7    Drama
          Drama             Jim Sturgess        Kevin Spacey
   7     25th Hour                                       2002    8    Drama
          Drama             Edward Norton       Philip Seymour ...
   8     3:10 to Yuma                                    2007    8    Drama
          Crime             Russell Crowe       Christian Bale
   9     50 First Dates                                  2004    7    Romantic
Comedy   Comedy            Adam Sandler        Drew Barrymore
   10    88 Minutes                                      2007    6    Action
         Crime             Al Pacino           Alicia Witt
...

***************************************************************************
Title: Men in Black
---------------------------------------------------------------------------
Year: 1997        Rating: 7
---------------------------------------------------------------------------
Leading Actor:    Tommy Lee Jones          Genre 1: Comedy
Supporting Actor: Will Smith               Genre 2: Comedy
---------------------------------------------------------------------------
PLOT:
Two men who keep an eye on aliens in New York City must try to save the
worldafter the aliens threaten to blow it up.
***************************************************************************

Search by genre for Romantic Comedy found:
MOVIE #  TITLE                                           YEAR RATING  GENRE
          ALT GENRE         LEAD ACTOR          SUPPORTING ACTOR
------- ------------------------------------------------ ---- ------ ---------
-------- ----------------- ------------------- -------------------
   1     50 First Dates                                  2004    7    Romantic
Comedy   Comedy            Adam Sandler        Drew Barrymore
   2     Alfie                                           2004    6    Romantic
Comedy   Comedy            Jude Law            Renee Taylor
   3     Always                                          1989    6    Romantic
Comedy   Romance           Richard Dryfus      Holly Hunter
   4     American President, The                         1995    7    Romantic
Comedy   Comedy            Michael Douglas     Annette Bening
   5     Benny & Joon                                    1993    7    Romantic
Comedy   Comedy            Mary Stuart Mas...  Johnny Depp
   6     Break-up, The                                   2006    6    Romantic
Comedy   Romantic Comedy   Jennifer Aniston    Vince Vaughn
   7     Bridget Jones 2: The Edge of Reason             2004    6    Romantic
Comedy   Drama             Renee Zellweger     Colin Firth
```

```
   8     Bridget Jones's Diary w. BJ2                    2001    7    Romantic
Comedy   Drama             Renee Zellweger     Hugh Grant
   9     Casanova                                        2005    7    Romantic
Comedy   Comedy            Heath Ledger        Oliver Platt
   10    Couples Retreat                                 2009    6    Romantic
Comedy   Comedy            Vince Vaughn        Jon Favreau
...

Search by actor for Anthony Hopkins found:
MOVIE #  TITLE                                           YEAR RATING  GENRE
          ALT GENRE         LEAD ACTOR          SUPPORTING ACTOR
------- ------------------------------------------------ ---- ------ ---------
-------- ----------------- ------------------- -------------------
   1     Bobby                                           2006    7
Biography         Drama             Anthony Hopkins     Harry Belafonte
   2     Legends of the Fall                             1995    7    Drama
          Drama             Brad Pitt           Anthony Hopkins
   3     Meet Joe Black                                  1998    7    Drama
          Drama             Brad Pitt           Anthony Hopkins
   4     Proof                                           2005    7    Drama
          Drama             Gweneth Paltrow     Anthony Hopkins
   5     Shadowlands                                     1994    7
Biography         Drama             Anthony Hopkins     Debra Winger
   6     World's Fastest Indian, The                     2005    8
Biography         Drama             Anthony Hopkins     Iain Rea

Search by year for 2007 found:
MOVIE #  TITLE                                           YEAR RATING  GENRE
          ALT GENRE         LEAD ACTOR          SUPPORTING ACTOR
------- ------------------------------------------------ ---- ------ ---------
-------- ----------------- ------------------- -------------------
   1     3:10 to Yuma                                    2007    8    Drama
          Crime             Russell Crowe       Christian Bale
   2     88 Minutes                                      2007    6    Action
         Crime             Al Pacino           Alicia Witt
   3     A Mighty Heart                                  2007    7
Biography         Drama             Dan Futterman       Angelina Joile
   4     Across the Universe                             2007    8    Musical
          Drama             Evan Rachel Wood    Jim Sturgess
   5     August Rush                                     2007    8    Drama
          Music             Freddie  Highmore   Keri Russell
   6     Before the Devil Knows You're Dead              2007    7    Drama
          Crime             Philip Seymore ...  Ethan Hawke
   7     Bourne Ultimatum                                2007    8    Action
         Adventure         Matt Damon          Joan Allen
   8     Breach                                          2007    7
Biography         Crime             Chris Cooper        Ryan Phillippe
   9     Breakfast with Scot                             2007    7    Comedy
         Drama             Cameron Anzel       Benz Antoine
   10    Charlie Wilson's War                            2007    7
Biography         Drama             Tom Hanks           Julia Roberts
...

Search by rating for 8 found:
MOVIE #  TITLE                                           YEAR RATING  GENRE
          ALT GENRE         LEAD ACTOR          SUPPORTING ACTOR
------- ------------------------------------------------ ---- ------ ---------
-------- ----------------- ------------------- -------------------
   1     007 - Casino Royale                             2006    8    Action
         Adventure         Daniel Craig        Eva Green
   2     25th Hour                                              2002    8    Drama
          Drama             Edward Norton       Philip Seymour ...
   3     3:10 to Yuma                                           2007    8    Drama
          Crime             Russell Crowe       Christian Bale
   4     A Beautiful Mind                                       2001    8
Biography         Drama             Russell Crowe       Ed Harris
   5     Across the Universe                                    2007    8    Musical
          Drama             Evan Rachel Wood    Jim Sturgess
   6     August Rush                                            2007    8    Drama
          Music             Freddie  Highmore   Keri Russell
   7     Back to the Future (1)                                 1985    8    Action
         Comedy            Michael J. Fox      Christopher Lloyd
   8     Bank Job, The                                          2008    8    Drama
          Crime             Jason Straham       Saffron Burrows
   9     Batman Begins                                          2005    8    Action
         Action            Christian Bale      Michael Caine
   10    Beauty and the Beast                                   1991    8
Family/Animation  Animation         Paige O'Hara        Robby Benson
...
```

```
Which input file would you like to use(type d for default file)? d
Which output file would you like to use(type d for default file)? d

DVD MENU OPTIONS

1 – Output Entire List
2 – Title Search
3 – Genre search
4 – Actor Search
5 – Year Search
6 – Rating Search (0 – 9)
0 – EXIT
Enter an option (0 to exit): 1

COMPLETE MOVIE LISTING!


DVD MENU OPTIONS

1 – Output Entire List
2 – Title Search
3 – Genre search
4 – Actor Search
5 – Year Search
6 – Rating Search (0 – 9)
0 – EXIT
Enter an option (0 to exit): 0

Thank You!!
❯ ./main
Which input file would you like to use(type d for default file)? d
Which output file would you like to use(type d for default file)? d

DVD MENU OPTIONS

1 – Output Entire List
2 – Title Search
3 – Genre search
4 – Actor Search
5 – Year Search
6 – Rating Search (0 – 9)
0 – EXIT
Enter an option (0 to exit): 1

COMPLETE MOVIE LISTING!


DVD MENU OPTIONS

1 – Output Entire List
2 – Title Search
3 – Genre search
4 – Actor Search
5 – Year Search
6 – Rating Search (0 – 9)
0 – EXIT
Enter an option (0 to exit): 2
Which Title are you looking for? Men in Black
Found the movie Men in Black!
```

```

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): 2
Which Title are you looking for? Shawshank Redemption
Sorry, the movie " Shawshank Redemption " was not found.

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): 3

Which Genre are you looking for? Romantic Comedy

Searching for the genre Romantic Comedy
Found 45 movies for the genre Romantic Comedy!

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): 3

Which Genre are you looking for? HORROR

Searching for the genre HORROR
Sorry, no movies for the genre HORROR were found.

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): 4

Which Actor are you looking for? Anthony Hopkins

Searching for the actor Anthony Hopkins
```

```
Found 6 movies for the actor Anthony Hopkins!

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): 4

Which Actor are you looking for? Don Johnson

Searching for the actor Don Johnson
Sorry, no movies for the actor Don Johnson were found.

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): 5

Which year are you looking for? 2007

Searching for the year 2007
Found 52 movies for the year 2007!

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): 5

Which year are you looking for? 1800
The number 1800 is an invalid entry
**** Please input a number between 1878 and 3000 ****

Which year are you looking for? 5
The number 5 is an invalid entry
**** Please input a number between 1878 and 3000 ****

Which year are you looking for? 3001
The number 3001 is an invalid entry
**** Please input a number between 1878 and 3000 ****

Which year are you looking for? a
**** Please input a NUMBER between 1878 and 3000 ****

```

```
Which year are you looking for? 1900

Searching for the year 1900
Sorry, no movies for the year 1900 were found.

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): 6

Which rating are you looking for? -1
The number -1 is an invalid entry
**** Please input a number between 0 and 9 ****

Which rating are you looking for? 10
The number 10 is an invalid entry
**** Please input a number between 0 and 9 ****

Which rating are you looking for? z
**** Please input a NUMBER between 0 and 9 ****

Which rating are you looking for? 8

Searching for the rating 8
Found 136 movies for the rating 8!

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): -1
Enter A Valid Option

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): 7
Enter A Valid Option

DVD MENU OPTIONS

1 - Output Entire List
2 - Title Search
```

```
3 - Genre search
4 - Actor Search
5 - Year Search
6 - Rating Search (0 - 9)
0 - EXIT
Enter an option (0 to exit): a
Please enter a number!
Enter an option (0 to exit): 0

Thank You!!
```