```
/*******************************************************************************
 * PROGRAMMER : Ali Eshghi
 * STUDENT ID : 1112261
 * CLASS      : CS1B
 * SECTION    : MW 7:30pm
 * Assign #3  : Searching linked list
 * DUE DATE   : 6 November 2019
 ******************************************************************************/

#include "Header.h"

/*******************************************************************************
 * Function - SearchTitle
 * ---------------------------------------------------------------------------
 * This function will get a title from the user and search the title content of
 * each node to see if there is a matching title in the nodes with the title
 * searched by the user. if found, the function prints out every content of the
 * node for the user, if not found, the function outputs that the title searched
 * by the user was not found.
 *
 * return type - nothing
 *               the function is void type
 ******************************************************************************/

void SearchTitle(MovieNode *head, ofstream &outFile)
{
    /*************
     * VARIABLES *
     ************/

    //NODES

    MovieNode *movPtr;      //PROCESS - stores the target name

    MovieNode node;             //PROCESS & IN - passes the info into persPtr


    //BOOLS

    bool found;             //PROCESS - condition value for the searched name


    //INTEGERS

    int stringLength;

    int maxLength;

    int lineLen;

    int wordLen;

    int i;


    //STRINGS

    string word;

    string line;

    string plot;

    /****************
     * INITIALIZING *
     ****************/

    word.clear();
    line.clear();

    maxLength = 75;

    /**************************************************************************
     * OUTPUT - outputs the title of the movie, the lead actor and the support
     *          actor, genre and the alternative genre and the plot of the movie
     **************************************************************************/


    //CHECKS FOR EMPTY LIST

    found = false;

    if (head == NULL)
    {
        cout << endl;

        cout << "Can not search for the title in an empty list.";

        cout << endl;
    }

    else
    {
        //creating new node

        movPtr = new MovieNode;

        //INPUT

        cout << endl;

        cout << "Which title are you looking for? ";
        getline(cin, node.title);

        *movPtr = node;

        cout << endl << endl;

        cout << "Searching for the title " << movPtr -> title;

        cout << endl;

        //GOES ATHROUGH THE LIST

        while ((head -> next != NULL) && !found)
        {
            //IF FOUND

            if (head -> title == movPtr -> title)
            {
                //OUTPUT

                cout << "Found the movie " << head -> title << "!" << endl;

                outFile << endl;

                outFile << left;
                outFile << "*****************************************************"
                      << "****************************************************"
                      << endl;

                outFile << "Title: " << head -> title << endl;

                outFile << "-----------------------------------------------------"
                      << "----------------------------------------------------"
                      << endl;

                outFile << setw(6)   << "Year:"   << setw (12) << head -> year;
                outFile << setw(8)   << "Rating:" << head -> rate << endl;

                outFile << "-----------------------------------------------------"
                      << "----------------------------------------------------"
                      << endl;

                outFile << setw(18) << "Leading Actor:"
                      << setw(25) << head -> leadActor;

                outFile << setw(9)  << "Genre 1:" << head -> genre << endl;


                outFile << setw(18) << "Supporting Actor:"
                      << setw(25) << head -> supportActor;

                outFile << setw(9)  << "Genre 2:" << head -> altGenre << endl;

                outFile << "-----------------------------------------------------"
                      << "----------------------------------------------------"
                      << endl;

                outFile << "PLOT:" << endl;

                stringLength  = head -> synopsis.length();

                cout << head -> synopsis;

```

```
       plot = head -> synopsis;

       for(i = 0; i <= stringLength; i++)
       {
            if(plot[i] != ' ')
            {
                word = word + head -> synopsis[i];
            }

            else
            {
                lineLen = line.length();
                wordLen = word.length();

                if((lineLen + wordLen) > maxLength)
                {
                    outFile << plot;

                    line.clear();

                }

                line = word + " ";

                word.clear();
            }

       } //END - FOR

       outFile << "*****************************************************"
             << "****************************************************"
             << endl << endl;

       found = true;

   } //END - IF

   else
   {

       head  = head -> next;

       found = false;

   }


} //END - WHILE

if (head -> next == NULL && head -> title == movPtr -> title)
{
   //OUTPUT

   outFile << endl;

   outFile << left;
   outFile << "*****************************************************"
         << "****************************************************"
         << endl;

   outFile << "Title: " << head -> title << endl;

   outFile << "-----------------------------------------------------"
         << "----------------------------------------------------"
         << endl;

   outFile << setw(6)   << "Year:"   << setw (12) << head -> year;
   outFile << setw(8)   << "Rating:" << head -> rate << endl;

   outFile << "-----------------------------------------------------"
         << "----------------------------------------------------"
         << endl;

   outFile << setw(18) << "Leading Actor:"
         << setw(25) << head -> leadActor;

   outFile << setw(9)  << "Genre 1:" << head -> genre << endl;


   outFile << setw(18) << "Supporting Actor:"
         << setw(25) << head -> supportActor;

   outFile << setw(9)  << "Genre 2:" << head -> altGenre << endl;

   outFile << "-----------------------------------------------------"
         << "----------------------------------------------------"
         << endl;

   outFile << "PLOT:" << endl;

   stringLength  = head -> synopsis.length();

   plot = head -> synopsis;

   for(i = 0; i <= stringLength; i++)
   {
        if(plot[i] != ' ')
        {

            word = word + head -> synopsis[i];

        }

        else
        {
            lineLen = line.length();
            wordLen = word.length();

            if((lineLen + wordLen) > maxLength)
            {
```

```
                         outFile << line;
                         line.clear();

                     }

                     line = word + " ";
                     word.clear();
                 }

            } // END - FOR

            outFile << "*****************************************************"
                  << "****************************************************"
                  << endl << endl;

            found = true;

         } //END - IF

         //IF NOT FOUND IN THE LIST

         else if (!found)
         {
             cout << "Sorry, the movie \"" << movPtr -> title;
             cout << "\" was not found.";

             cout << endl;
         }

     }

    //JUST TOBE SAFE
     movPtr = NULL;
}

```