CPT-287

Team 2

Movie Management System

Chris **Anderson**

Luke **Hunt**

Steven **Shackleford**

System Design

The movie management system is designed to use doubly-linked lists to interact with movies and track which movies are coming, and which are currently showing. Iterators were used to traverse the linked lists. A text file is used for initial input, output, and long term storage. A Scanner object is used to loop through and parse the text file during load. A PrintWriter is used to output and write the contents of the lists to the same file used in the initial load. A custom Movie object, a menu system, and methods contained in a custom MovieListMethods class were also created.

Movie Object

The relevant movie data is stored in a custom object defined in Movie.java. The Movie class contains data fields for the movie title, received date, released date, description, and whether it is received (in the coming list) or released (in the showing list). Methods for getting, setting, printing to string, and date formatting aref also included in the Movie class.

Linked Lists

Two doubly-linked lists, received (AKA coming) and released (AKA showing), are created using the standard Java LinkedList class. This enables us to use predefined nodes that store the movie objects and traverse these lists using ListIterator objects. The received list must be kept in non-decreasing order by released dates. These lists contained nodes that each stored a single Movie object.

Menu System

Users can interact with the program using a menu system through the system console. The menu operates using a while loop that checks for an exit key after every pass and a case/switch system for evaluating input. Integers of 1-7 are used to access commands in the menu. These commands are as follows:

1. Display all movies in each list

a. Movies are displayed with headers separating the two lists
b. Iterators are used to loop through each list and use the overridden toString method in the Movie class to display data fields

2. Edit movie in the coming list

a. Prompts user for movie name
b. If movie exists with name given
    i. Asks user for which information should be edited
    ii. Allows user to keep data by leaving input empty and hitting enter

        iii. Removes movie with old data using iterator
        iv. Adds movie with new data in the correct location using custom add method
    c. If movie does not exist, or dates are invalid
        i. Error message is given and no movie is accessed

3. Add movie to the coming list

a. Prompts user for title, release date, description and received date
b. Assumes movie is of status received since it is added to the coming list
c. Error message is given if dates are invalid or if movie with the same name exists in either coming or showing lists
d. If dates are valid, loops through the coming list and checks the release date of each movie in the list
    i. If list is empty, add to the beginning/end of list
    ii. If date of current movie iterated over is greater than the movie being added, go back to previous movie in list and use iterator add method to insert new movie
    iii. If made to the end of the list without finding a greater release date, add to the end of list

4. Start showing movies with a given date

a. Prompts user for date
b. Uses an iterator to loop through each movie in the coming list
c. If release date matches given date, change status to released in the coming list and then add to showing list
d. Loop through coming list a second time and remove all movies that now have a status of released

5. Display number of movies before a given release date

a. Prompts user for date
b. Uses an iterator to loop through each movie in the coming list
c. If release date is less than (before) given date add to counter
d. Display counter

6. Save to text file

a. Uses iterator to loop through both lists
b. Uses print writer to write to text file in specified format

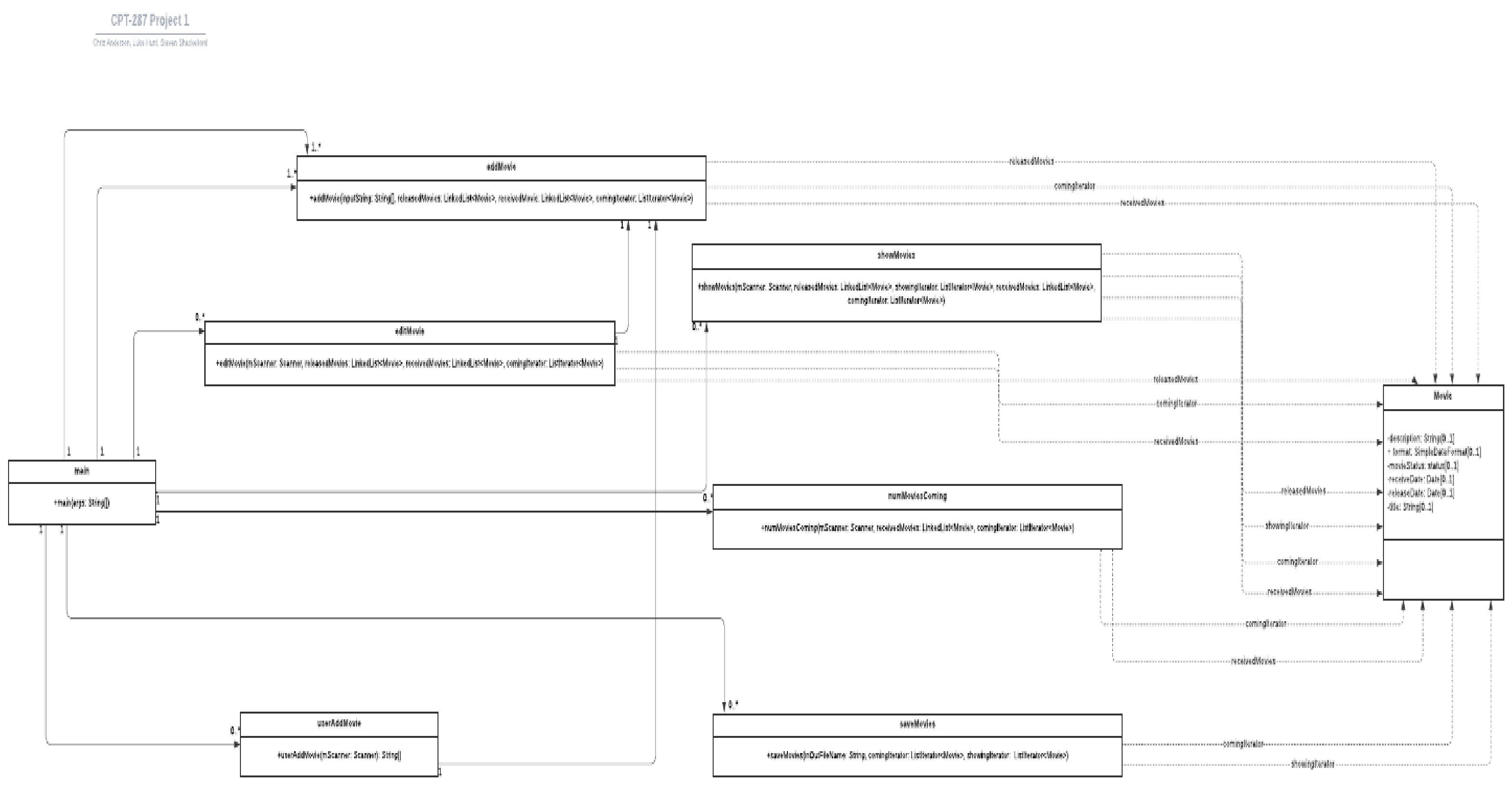
7. Exit
    a. Terminates program

Movie List Methods

Methods that needed to be defined outside of the standard LinkedList or ListIterator methods were created within a MovieListMethods class. These methods include:

- addMovie
    - Parameters: string array, released movie list object, released movie iterator, received movie list object, received movie iterator
    - Returns void
    - Splits string array into data fields required to construct a new movie object
    - Date objects are created by converting strings
    - Checks for invalid dates
        - Incorrect format
        - Release date less than or equal to received date
    - Checks for an existing movie in the coming or showing lists with the same name
    - If released status
        - Add to the end of the showing list
    - If received status
        - Iterate through list and check released date
        - If list is empty, add to list
        - If date is greater than movie to be added, add before using iterator
        - Else, add to end of list
- userAddMovie
    - Parameters: scanner object for input
    - Returns string array to be used in addMovie method
    - Prompts user for all data fields
- editMovie
    - Parameters: scanner object for input, released movie list object, released movie iterator, received movie list object, received movie iterator
    - Returns void
    - Prompts user for title of movie to be edited
    - Checks if coming list is empty
    - Uses iterator to check title of all movies in coming list
    - If a match is found
        - Prompts user for new values for fields
        - If input is blank, assumes existing values
        - Removes movie object from the coming list
        - Uses the addMovie method to insert a new movie with the new values into the coming list

- showMovies
  - Parameters: scanner object for input, released movie list object, released movie iterator, received movie list object, received movie iterator
  - Returns void
  - Prompts user for release date
  - Uses iterator to check release date for all movies in the coming list
  - If match is found
    - Status is changed to released
    - Movie is added the the showing list
  - Iterator checks status of all movies in the coming list
  - If status is released
    - Remove movie from the coming list
- numMoviesComing
  - Parameters: scanner object for input, received movie list object, received movie iterator
  - Returns void
  - Prompts user for release date
  - Uses iterator to check release date for all movies in the coming list
  - If match is found
    - Counter is incremented
  - Counter is printed to the console
- saveMovies
  - Parameters: string for filename of output, released movie list object, released movie iterator, received movie list object, received movie iterator
  - Returns void
  - Iterators are used to loop through each list and use the overridden toString method in the Movie class to display data fields

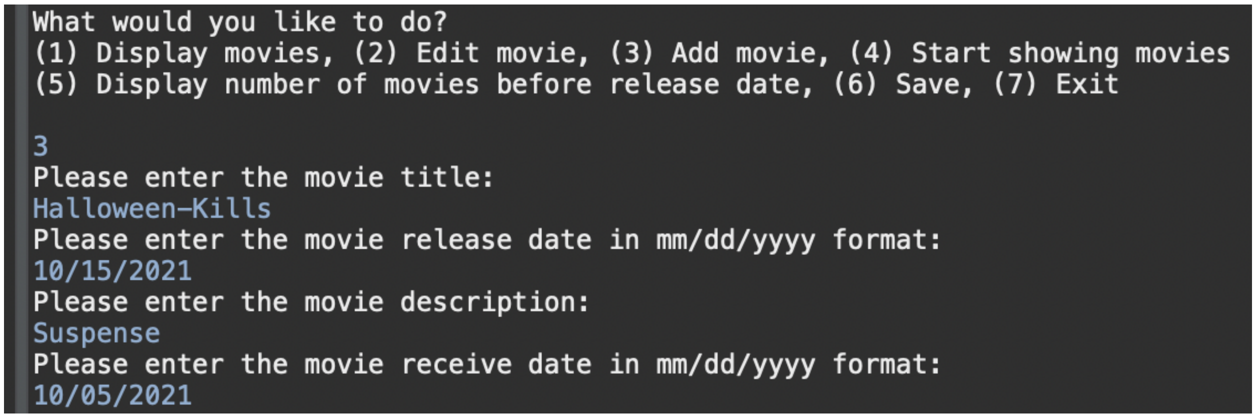

CPT-287 Project 1
Chris Anderson, Luke Hunt, Steven Shackelford
main
+main(args: String[])
addMovie
+addMovie(inputString: String[], releasedMovies: LinkedList<Movie>, receivedMovie: LinkedList<Movie>, comingIterator: ListIterator<Movie>)
editMovie
+editMovie(mScanner: Scanner, releasedMovies: LinkedList<Movie>, receivedMovies: LinkedList<Movie>, comingIterator: ListIterator<Movie>)
showMovies
+showMovies(mScanner: Scanner, releasedMovies: LinkedList<Movie>, showingIterator: ListIterator<Movie>, receivedMovies: LinkedList<Movie>, comingIterator: ListIterator<Movie>)
numMoviesComing
+numMoviesComing(mScanner: Scanner, receivedMovies: LinkedList<Movie>, comingIterator: ListIterator<Movie>)
userAddMovie
+userAddMovie(mScanner: Scanner): String[]
saveMovies
+saveMovies(inOutFileName: String, comingIterator: ListIterator<Movie>, showingIterator: ListIterator<Movie>)
Movie
-description: String[0..1]
+ format: SimpleDateFormat[0..1]
-movieStatus: status[0..1]
-receiveDate: Date[0..1]
-releaseDate: Date[0..1]
-title: String[0..1]
releasedMovies
comingIterator
receivedMovies
showingIterator
1
1..*
0..*

//At least 2 test cases. Give input data and give expected outcome vs actual outcome

Test case 1:

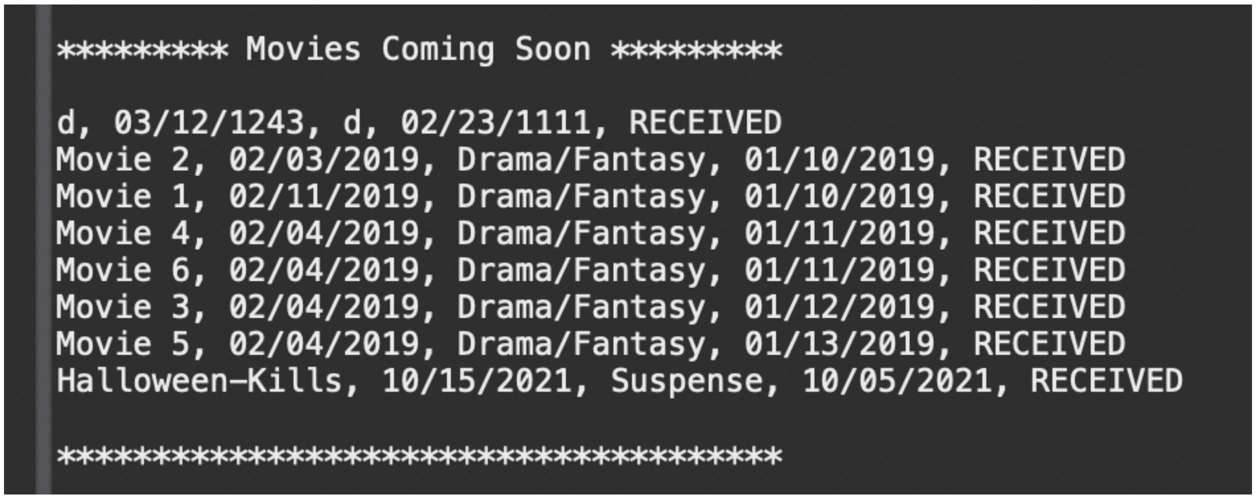
Creating a new movie and modifying its description and title

For creating a brand new movie and then modifying its description and title, you will need to call the "edit movie" method only once. The movie that will be added will be "Halloween-Kills", with a release date of 10/15/2021, a description of "Suspense", a receive date of 10/05/2021, and a status of received. The method to edit a movie from the menu will then be called, and the movie will be searched for inside the "coming" movies list by it's name and will be changed accordingly.



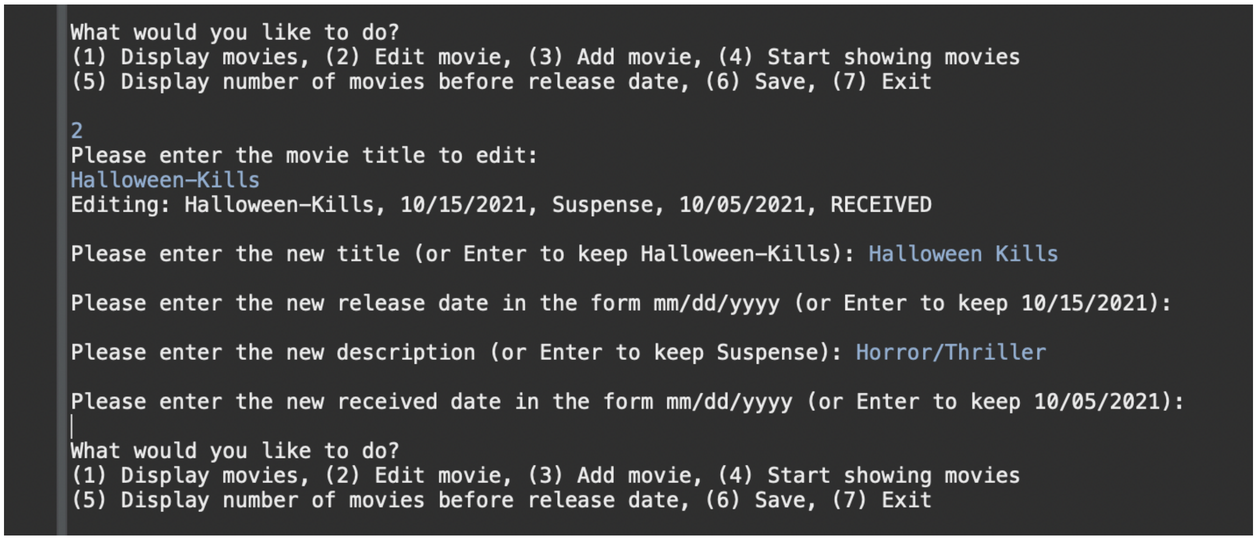
```
What would you like to do?
(1) Display movies, (2) Edit movie, (3) Add movie, (4) Start showing movies
(5) Display number of movies before release date, (6) Save, (7) Exit

3
Please enter the movie title:
Halloween-Kills
Please enter the movie release date in mm/dd/yyyy format:
10/15/2021
Please enter the movie description:
Suspense
Please enter the movie receive date in mm/dd/yyyy format:
10/05/2021
```

Movie created



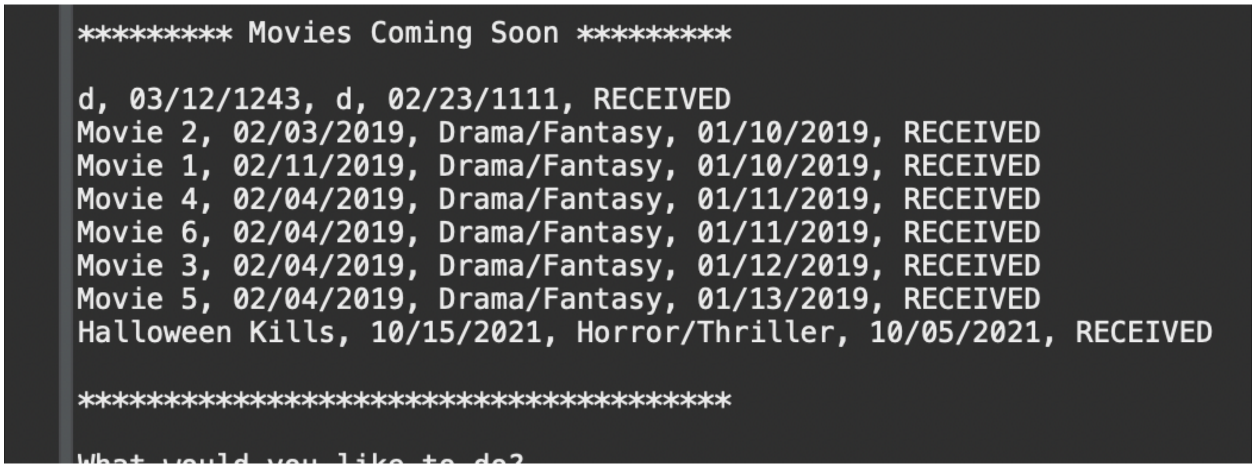
```
********* Movies Coming Soon *********

d, 03/12/1243, d, 02/23/1111, RECEIVED
Movie 2, 02/03/2019, Drama/Fantasy, 01/10/2019, RECEIVED
Movie 1, 02/11/2019, Drama/Fantasy, 01/10/2019, RECEIVED
Movie 4, 02/04/2019, Drama/Fantasy, 01/11/2019, RECEIVED
Movie 6, 02/04/2019, Drama/Fantasy, 01/11/2019, RECEIVED
Movie 3, 02/04/2019, Drama/Fantasy, 01/12/2019, RECEIVED
Movie 5, 02/04/2019, Drama/Fantasy, 01/13/2019, RECEIVED
Halloween-Kills, 10/15/2021, Suspense, 10/05/2021, RECEIVED

*************************************
```

Movie automatically added to the “coming” list, sorted by release date



```
What would you like to do?
(1) Display movies, (2) Edit movie, (3) Add movie, (4) Start showing movies
(5) Display number of movies before release date, (6) Save, (7) Exit

2
Please enter the movie title to edit:
Halloween-Kills
Editing: Halloween-Kills, 10/15/2021, Suspense, 10/05/2021, RECEIVED

Please enter the new title (or Enter to keep Halloween-Kills): Halloween Kills

Please enter the new release date in the form mm/dd/yyyy (or Enter to keep 10/15/2021):

Please enter the new description (or Enter to keep Suspense): Horror/Thriller

Please enter the new received date in the form mm/dd/yyyy (or Enter to keep 10/05/2021):

What would you like to do?
(1) Display movies, (2) Edit movie, (3) Add movie, (4) Start showing movies
(5) Display number of movies before release date, (6) Save, (7) Exit
```

Movie title and description edited



```
********* Movies Coming Soon *********

d, 03/12/1243, d, 02/23/1111, RECEIVED
Movie 2, 02/03/2019, Drama/Fantasy, 01/10/2019, RECEIVED
Movie 1, 02/11/2019, Drama/Fantasy, 01/10/2019, RECEIVED
Movie 4, 02/04/2019, Drama/Fantasy, 01/11/2019, RECEIVED
Movie 6, 02/04/2019, Drama/Fantasy, 01/11/2019, RECEIVED
Movie 3, 02/04/2019, Drama/Fantasy, 01/12/2019, RECEIVED
Movie 5, 02/04/2019, Drama/Fantasy, 01/13/2019, RECEIVED
Halloween Kills, 10/15/2021, Horror/Thriller, 10/05/2021, RECEIVED

*************************************

What would you like to do?
```

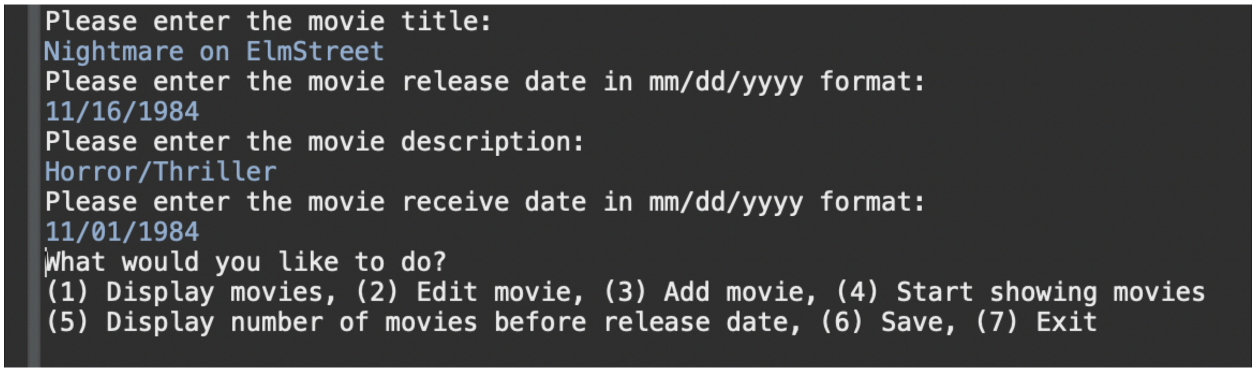
Movie title and description updated inside “coming” list

The movie was successfully created and added to the “coming” movies list, and sorted by release date. The movie was then correctly edited through the edit movie method and correctly displays the updated information. This was the expected outcome as the movie still retained it’s correct details and nothing was changed that could mark the movie as “invalid”.

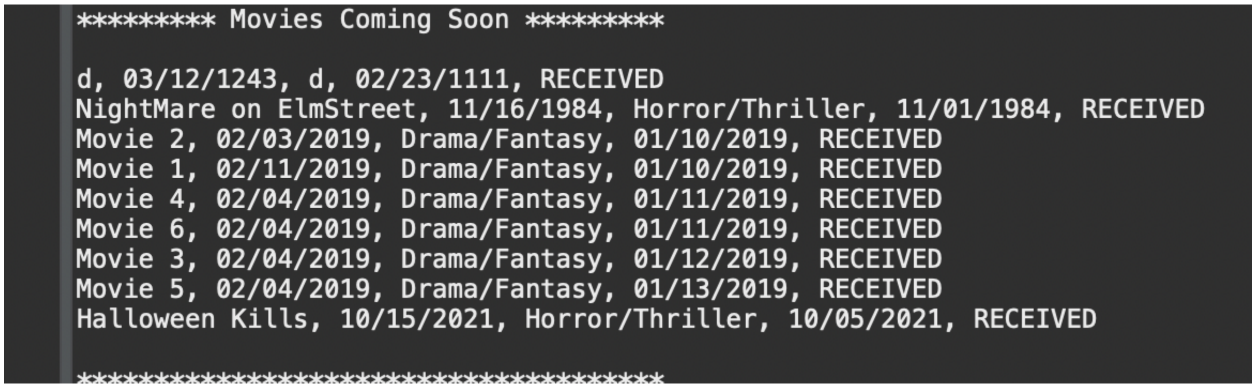
Test case 2:

Creating a new movie and attempting to change it’s release date to before the receive date

Creating a new movie successfully and then attempting to change the movie's release date to before the movie was released should return an error and cancel the method for editing the movie. It should not save any other attempted edits to the movie should the release date be invalid and remove the movie from the list as it is no longer a valid movie.

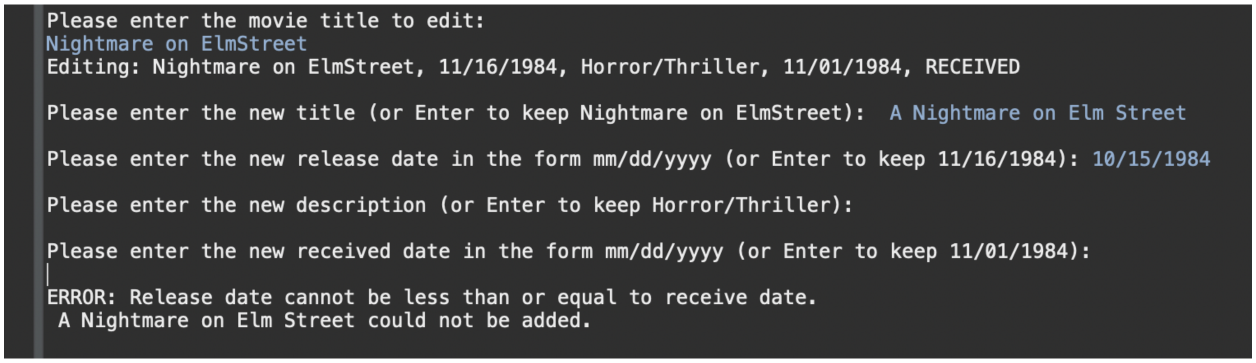


```
Please enter the movie title:
Nightmare on ElmStreet
Please enter the movie release date in mm/dd/yyyy format:
11/16/1984
Please enter the movie description:
Horror/Thriller
Please enter the movie receive date in mm/dd/yyyy format:
11/01/1984
What would you like to do?
(1) Display movies, (2) Edit movie, (3) Add movie, (4) Start showing movies
(5) Display number of movies before release date, (6) Save, (7) Exit
```



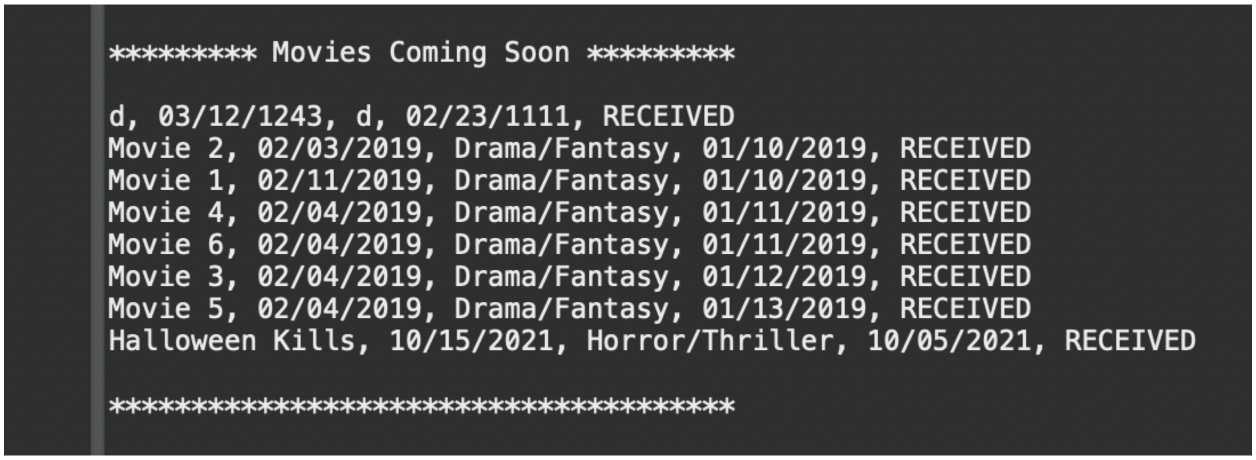
```
********* Movies Coming Soon *********

d, 03/12/1243, d, 02/23/1111, RECEIVED
NightMare on ElmStreet, 11/16/1984, Horror/Thriller, 11/01/1984, RECEIVED
Movie 2, 02/03/2019, Drama/Fantasy, 01/10/2019, RECEIVED
Movie 1, 02/11/2019, Drama/Fantasy, 01/10/2019, RECEIVED
Movie 4, 02/04/2019, Drama/Fantasy, 01/11/2019, RECEIVED
Movie 6, 02/04/2019, Drama/Fantasy, 01/11/2019, RECEIVED
Movie 3, 02/04/2019, Drama/Fantasy, 01/12/2019, RECEIVED
Movie 5, 02/04/2019, Drama/Fantasy, 01/13/2019, RECEIVED
Halloween Kills, 10/15/2021, Horror/Thriller, 10/05/2021, RECEIVED

**************************************
```

Movie successfully created and stored inside the “coming” list



```
Please enter the movie title to edit:
Nightmare on ElmStreet
Editing: Nightmare on ElmStreet, 11/16/1984, Horror/Thriller, 11/01/1984, RECEIVED

Please enter the new title (or Enter to keep Nightmare on ElmStreet):  A Nightmare on Elm Street

Please enter the new release date in the form mm/dd/yyyy (or Enter to keep 11/16/1984): 10/15/1984

Please enter the new description (or Enter to keep Horror/Thriller):

Please enter the new received date in the form mm/dd/yyyy (or Enter to keep 11/01/1984):

ERROR: Release date cannot be less than or equal to receive date.
 A Nightmare on Elm Street could not be added.
```

Failed attempt at editing a movies release date to be before the receive date



```
********* Movies Coming Soon *********

d, 03/12/1243, d, 02/23/1111, RECEIVED
Movie 2, 02/03/2019, Drama/Fantasy, 01/10/2019, RECEIVED
Movie 1, 02/11/2019, Drama/Fantasy, 01/10/2019, RECEIVED
Movie 4, 02/04/2019, Drama/Fantasy, 01/11/2019, RECEIVED
Movie 6, 02/04/2019, Drama/Fantasy, 01/11/2019, RECEIVED
Movie 3, 02/04/2019, Drama/Fantasy, 01/12/2019, RECEIVED
Movie 5, 02/04/2019, Drama/Fantasy, 01/13/2019, RECEIVED
Halloween Kills, 10/15/2021, Horror/Thriller, 10/05/2021, RECEIVED

*************************************
```

Movie has been removed from list due to being an invalid movie

The movie was Successfully added to the list, but due to the failed attempt at editing the movie's details, it was no longer a valid movie and was removed from the "coming" movies list. This is the expected outcome as the movie's release date was edited to be before the receive date, which is an invalid date. Movies are only stored into the coming list after being edited should all the details of the movie be correct.//Each team members contribution
Chris: input loop, saveMovies method, UML, & comments

Luke: addMovie, editMovie, showMovies, userAddMovie, numMoviesComing methods, design explanation

Steven: menu, movie class, date formatting, test cases, & future improvements

//Discuss improvements to the system that could be done in the future

System could print the movie after it is edited to the user and then prompt if they would like to save the movie with the new details, or cancel the edit.
System could add all movies in the “coming” list to the showing list by inputting a given date instead of only movies showing on that day.
System could save information when an error occurs and prompt the user for corrections rather than deleting movies with incorrect information. This could improve user experience and reduce duplicate data entry.