CSCI 41 Project 1 Report

By: Brandon Hang

Instructor: Professor Belman

Class: CSCI 41

Date: November 23, 2021

# **Main Function**

The purpose of the main function is to be able to call all the functions and performs their actions. The main creates both list (The category list and the movies list) objects and uses those objects to call the other functions within their classes.

Text

Description automatically generated

As shown above, you can see that each category is being added to the double linked list and the movies are being added to a circular linked list.

Text

Description automatically generated

Then the rest of the main function can be a way of performing each functions that are within objects classes.

# A picture containing text, wall, blackboard, tiled Description automatically generated**Category Nodes Structure**

The **CategoryNodes** structure has:

* The name of the category
* The pointer to the next node
* The pointer to the previous node
* The pointer to the Movie nodes (linkC)
* Size of the movie list

The thought process can be shown in the picture above where linkC points to the head movie node and links that way.

Text

Description automatically generated

The functions are constructors that are called to create that object (node).

# **Categories Double linked list class**

The **CategoriesDLL** class has:

* Inserting a category into the double linked list
* Searching the category within the double linked list
* Searching a movie (checking each category)
* Removing a category along with the movies in it
* Delete all of the movies that are before a certain date
* Prints all of the categories and the movies along with it

Adding Category Function

So, what this function does is that it creates temp node as the category node that is to be added into the list. It first checks if the head is null and if so, it places that node at the beginning of the list. Otherwise, it would loop through the list until it the last node next points to null, then places the temp node into the list. (Inserting at the rear of the list.)

Text

Description automatically generated

Searching for the category with the given category name function

This function searches for the category within the double linked list and returns the pointer to the category node.

Text

Description automatically generated

Searching for the movie with the given movie name function

The function iterates through each category and in those category, they loop through the circular linked list so that the movie can meet its condition of being found. It breaks the entire loop of going through each category when it is found, and if it isn’t go to the next category. Prints not found once the categories list has been looped.

Text

Description automatically generated

Removing a category by its name function

The function first loops through the double linked list class and checks if the given name matches with where the temp position is at. If not it goes onto the next and loops. If it is found, it relinks the nodes before and after the temp node that is being deleted.

Text

Description automatically generated

Printing all the categories and the movies that are in them function

This function goes through the entire double linked list category and the circular linked list and prints each of their names out. It finishes until the double linked list is at the end (next points to NULL).

Text

Description automatically generated

Deleting all of the movies that are before a certain date function

So this function uses many of other class functions, but those will be explained later or have already been explained. It first sets node as the head of the double linked list. With that, loops through the entire list, but finds the category of the list, that way it can make the head and tail of the circular linked list. It then loops through the entire circular and checks if current positions date is less than the date that is given. If it is, it goes through each case and check if 1.) there is only one element in the list, 2.) if the node that is at the front of the list needs to be deleted, 3.) deletes the node that is at the end of the list, 4.) or if it is somewhere in the middle. Relinks them if one of those cases exist, otherwise it goes to the next movie. When the circular list hits the head position again, we go onto the next category and repeat the process.

Text

Description automatically generatedText

Description automatically generated

# A screenshot of a computer Description automatically generated with low confidence**Movie Node Structure**

The **MovieNodes** structure has:

* The movie name
* The movie category
* The date
* The pointer that points to next
* The pointer that points to previous

Thought process can be represented with the picture above as each of those yellow boxes are considered a movie node.

Text

Description automatically generated

The constructors are to create the objects(nodes).

# **Movies Circular Linked List Class**

The **MoviCLL** class has the functions:

* Adding a movie to the category in alphabetical order
* Searching for a movie within the circular list using binary search
* Getting the index of all of the movies that are in the category
* Prints all of the movies with the given category

Adding a movie into its category alphabetically

This function adds a movie into the circular linked list by it category. It sets the temp as a node to be inserted and then goes to find the category that was given so that it could be inserted within that category. There are many cases in this function and it is inserted if 1.) there is nothing in the list (NULL), it inserts it there, 2.) it checks if the list only has one node in it and needs to be inserted at the beginning, 3.) checks if the list needs to be inserted at the beginning and has more than one node in the list, 4.) Checks if it belongs somewhere in the middle of the list, 5.) checks if the list already contains it (does nothing with the node if that is the case), 6.) and finally if the node needs to be inserted at the end of the list. Loops through the whole list until it follows with one of the conditions.

Text

Description automatically generated

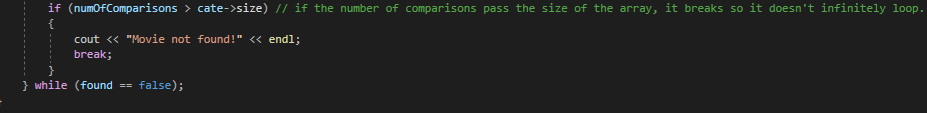
Text

Description automatically generated

Searches for the movie using a binary search function

This function uses binary search to find the movie name in its category. It first needs to find the category, then sets the start at the head of the circular list and end at the head->previous(because it is circular). With the size recorded when adding and deleting the nodes, it finds the middle that was dividing it by 2. Using a for loop, it finds the middle node “temp”, which is used to determine if it matches the movie name. If it doesn’t match and is less than the given movie name, it sets the start as the next node goes through the whole process. Same goes with greater than, but instead it sets the end as the previous node. If it isn’t found it breaks out of the loop.

Text

Description automatically generated 

Prints the all of the movies index with its category function

The function first finds the category that was given and with that, it goes through each movie and when each movie is passed, it adds 1 to the counter so that we can keep track of the index. Stops the loop when the iterator goes back to the head.

Text

Description automatically generated

Prints all of the movies within its category function

This function prints all the movies with the given category. First finds the category, and loops through it printing each of the nodes movie name.

Text

Description automatically generated

# **Date Class**

The **Date** class has:

* The month
* The day
* The year
* A operator overload class to compare two dates

Text

Description automatically generatedThe operator overload < function

This function compares two dates and checks if the upload date of the current node is less than the date that is given. (This function is used for when we were deleting the nodes that are before a certain date.)