Major 1 Data Structures Checklist

User Input Requirements

- The user will input an “infinite”(up to how much memory your system can hold) amount pairs of strings
    - A movie genre
    - A movie title
    - A movie rating
- You must stop the user from getting input when either the title or genre is a “.”
- These must be entered on separate lines
- Assume the input string is less than 30 characters

- Rating cannot be anything but 1-5 inclusive

Programming / Functionality Requirements

- Define a Struct it needs to be called MovieInfo ○ It has 2 pointers to a char, one for the genre and one for the title
- Both the pointers must be set up by using malloc
- Define the struct MovieInfo before any functions
- In main declare a struct
- Put input into a sorted doubly linked list
- Have 2 lists one that sorts by rating and another that sorts by genre
- After input is complete print both lists once
- Then obtain a genre and title pair to then change the rating
    - Change this rating in the genre list just by updating it
    - Change the rating in the rating list by deleting the node and inputting back a new set of data
- Print both lists again
- Then free both lists when finished

- Write a function called ‘getMovieInfo’ it takes 3 parameters, the first is a pointer to the structure list, the second is the movie genre obtained from the user and the third is the movie title also obtained from the user. This is where you allocate the blocks of data to store the genre and title, you need to flip through the list putting every pair of strings into a different list
- Write a function called ‘printMovieInfo’ this function takes the list struts as the only parameter and prints out all the movie genres and titles in a neat format, use width specifiers to split up the genre and title by 35 characters in length

  -Create the findMovie() function. It returns NULL if a movie is not found or it returns a pointer to the node containing a movie, if both the genre and title are matched (with a case-sensitive match, so you can use strcmp for this). It takes three parameters:
    - MovieNode *head: head of list
    - char *genre: pointer to null-terminated string containing genre
    - char *title: pointer to null-terminated string containing title
  - If only the genre or the title are found but both are not found in the same node, the movie is not found.

  -Create the deleteNode() function. It deletes a node, using three parameters:
    - MovieNode *node: node to delete
    - MovieNode **head: pointer to head of list
    - MovieNode **tail: pointer to tail of list
  - The key to this function is relinking pointers around the node before deleting.
  - If node is NULL, it returns immediately.
  - It returns nothing.
-
- Free the memory when done being used
- Do not use calloc()
- Do not have any other input or output

Commenting

- Comment where you think necessary
- Function headers
- Main Header

Style

- Make sure to go by my naming style
- Make sure to indent properly
- Make sure to space everything out properly and the same to make it look neat, I always do 4 spaces between functions and above main I usually use like 2

File Naming

- Project is called dsA1
- Source files are called dsA1.cpp for the main, dsA1LL.cpp for the functions and dsA1.h for the header files
- Checklist as well! Make sure the checklist is named “checklist.pdf” or check to see if it changed
- Zip file called mini1.zip

Things Required in all assignments

- Return 0 in main()
- Braces around single-line bodies for if, while and for… statements.
- No Global Variables
- Use Magic Numbers where applicable

Submitting

- Clean solution!!!!
- Delete vs.
- Zip the folder
- Make sure the solution is not bigger than 10kb