Prathyoosha Chaya, CS161

ALGORITHM 4:

**Header file:**

The header file will include the struct definition, with the function prototypes of the following public member functions in the class interface:

Constructor function: The purpose of the constructor function is to initialize the data members. Here, the pointer will be set to the structure will be set to null, the subject counter is set to 0, and the size of the array is set to 0.

Read function: The purpose of this function is to read new subjects from the user and add them into the dynamically allocated array. A new object is created using the pointer, then the function will prompt the user to enter the name of the new subject, and will store this into the first member. Next, the user will be able to enter all fields, and all structure members will be filled out, using the same delimiting process as the previous program. If the user would like to add another subject, this will loop again, but the loop will not enter if the array counter is greater than 20. For the subject name, learned info, and improvement info fields, the user input is read into a temporary array of characters, then a new dynamically allocated array is created based on the length of the temporary array. The temporary value is copied into the new array.

Load function (optional): The purpose of the load function is to take the information in the text file and store it into a dynamic array of structures. It uses a pointer to the array of structures and an array index counter, which are the class’s private data members, and uses the opened text file to write the data. It uses the existing delimiters to read into the dynamic array’s fields, by access. The process for writing these values remains the same in this program.

Display function: The purpose of the display function is to display all subjects with their corresponding learned information, rating, and improvement ideas. This function will take the array of structures made in the main function (and revised in the read function) and output its members. This procedure does not change.

Search function: The purpose of this function is to show a matched subject to a use-entered subject. The user inputted subject and match are passed into the function by value. The strcmp function is used to match what the user entered to the desired struct object (“Variables”, “Data Types”, etc.). Before matching, both the struct object and the user-inputted value is turned into all lowercase letters, to avoid case-sensitivity. This is done by calling the convert to uppercase function. If the subject desired is not found, then the user will be alerted that the subject does not exist.

Remove function: The purpose of this function is to remove an element of the dynamic array. This function takes the subject counter by reference and the pointer by value. Delete the object that the pointer points to. Finally, the subject counter is decremented.

Destructor function: The destructor will call the write function, then deallocate all previously allocated dynamic memory, including the array of structs created, the dynamic name, improvement info, and learned info arrays, the pointer, subject counter, and size of the array.

Write function (to accompany optional load function): The purpose of this function is to write the contents of the array of structs back into the file, for the new subjects added in the read function. It takes the array of structs and the array /counter variable as reference. In a loop a local index variable is created (type int). The object members will be outputted to the file (no append), separated with “;” delimiters. At the end of the subject, a “\n” delimiter is placed and the index is incremented. This loops until the index meets the array counter value. It is not necessary to be run for the “add subject” function, as the subject is appended to the end of the file within the function.

Inside the class interface, the private data members of the class are the pointer to the subject structure, the counter for the number of subjects, and the size of the dynamic array.

**Implementation files:**

The first implementation file will contain the above function definitions.

MAIN FUNCTION:

In the main function, the same menu structure will be used. The input file and output file variables (ifstream/ofstream type) are created, and a “menu option” variable (type int, to keep track of the menu item entered). The text file is opened, and the load function is called.

Then, the menu is outputted by calling the menu function. The user can choose which option they would like to execute from the following menu format:

1: Display all subjects.

2: Add a new subject.

3: Remove subject

4: Save and quit archive

5: Quit archive without saving.

The user will be prompted to enter a number, which is stored into the “menu option” variable once the function exits. This process is similar to the previous program, with changes to the options and function executions in the cases. If a 1 is entered, then the “display” function is entered. At this point, all data from the text file will have been read into the array of structs. Once the function is exited, the menu function is entered again. If a 2 is entered, then the “load” function is entered, and the menu function enters. If a 3 is entered, then the “search” function runs, then the “remove” function is entered, followed by the menu function entering. If a 4 is entered, the write function is called, the file is closed, and the program quits. If a 5 is entered, then the file is closed and the loop is exited.

OTHER FUNCTIONS IN IMPLEMENTATION FILE:

Convert to uppercase function: The purpose of this function is to convert the inputted array of characters to uppercase, using the toupper functionality in the cctype library. It takes a source and destination array of characters as input parameters, and changes the contents of destination to be the uppercase version of the source array of characters, going character by character. This will be called in the search function and is in the main implementation file.

Menu function: This function takes no parameters and returns an int value. It outputs the menu for the user to select, and return the user’s selection. The menu displayed is shown in the main function description below. There are no changes to be made to this function to accommodate dynamic memory, aside from the choices given to the user.

Data Flow Diagram:

Pointer, subject count variable

Write function

Pointer, subject count variable

Load function

Matched subject

Pointer, subject count variable (subject count updated)

Pointer, subject count variable

Pointer, subject count variable

Search input, matched subject

Convert uppercase function

Search function

Remove function

Display function

Chosen Number

Read function

Menu function

Main function