Kunio Comey

CS202

Design #1

Major Design Considerations:

The biggest design consideration is how the hierarchy will work. To start, the three different classes that will hold the primary information will be assignment, message, and schedule. Assignment and message will be a part of a linear linked list so that we can have multiple items of each. The schedule will be an array of linear linked lists. Each element will represent a different date and the nodes in that element will be each event that will occur on that date. All of this will be contained in a user class that will hold the message list, assignment list, and schedule table. Through the user class, the client will have access to the schedule, message list, and assignment list.

Classes:

The class at the top of the hierarchy will be the name class. The name class will be used as a title or a name depending on which class it is but will still use the class name “name” for this program. The name class will have a character array and an integer to hold the length of the string passed in. The different functions will be read, edit, display, clear, and copy.

The four classes that will inherit from name will be the user class, assignment class, message class, and event class.

The user class will have a schedule, head pointer for messages, and head pointer for assignments. The functions in this class will be inserting a message for the message list, inserting an assignment for the assignment list, display for all its members, and editing all its members. Since this class is mostly using containing relationships for the other classes, the functions are mostly wrapper functions.

The assignment class will have a character array to hold the description of the assignment. The different functions will be read description, edit description, display, clear description, and copy description. This class only has a hierarchical relationship to name and then an assignment node class. The assignment node class will have a get next function and a set next function with a node pointer to the next node.

The message class will have a character array to hold the message. The different functions will be read message, edit message, display message, clear message, and copy message. This class only has a hierarchical relationship to name and then is a parent to the message node class. The message node class has similar functions to the assignment node class.

The event class will have a character array for description and another for time. The functions will be reading in description, reading in time, editing the event, clearing event, and displaying the event. The event class will be inherited from the name class and then will be a parent for the event node class. The event node class will have set next and get next for functions. Data members for the node class will be next node pointer.

The last class will be the schedule class. This class will have an array of linear linked lists of event nodes. Each of the elements will correspond to a different date of the month. If there are multiple events on the same date, they will just add on to the linear linked list. If it is an event that doesn't share the same date, then it will create a new linear linked list to the corresponding element in the array. The functions will be an insert, display, recursive display, remove, and recursive remove (for destructor). This will have a containing relationship to the event node class and is contained by the user class.

Avoiding getters and setters:

To avoid using getters and setters, most of the prompting and reading will occur inside of the class itself. Rather than reading and prompting from main, the class itself will prompt and read so that main doesn't have to interact directly with the class. The program will have little interactions with passing and receiving different character arrays with each other and will try to stay inside of the scope of the class. The only exception for getters and setters would be for the node classes since they need to interact with the classes that are containing them.