Caameron Nakasone CS202 Programming Systems caameron@pdx.edu

Program #3 Design Write Up

For Program #3 I plane to have 5 different classes that will work together to create the desired “discord” like application. The 5 classes will be a meeting, node, person, response, and poster class. The person class will be a base class that will contain basic information about a single person who wants to enter the application. However since, someone will either be using this application to post a meeting or to respond the poster and response classes will be derived from the person class. They will everything a person is but a few extra traits depending on if they are responding to a meeting or posting one. The meeting class will also be another base class that will contain all the details of the meeting such as the name, and list of people attending, etc. The meeting class will also have a “has a” relationship with the poster and response class. Where each meeting will have a poster and a response. Finally the node class will be used to create the balance tree where each node will have a linear linked list of meetings. Therefore the node will primarily have functions to interact with the balance tree and meeting pointer that will the start of the list.

Since this program revolves around operator overloading, many of the operators will be overloaded to help with the implementation of the program. The assignment operator will be used for every class, as well as the input and output operators to make displaying the information easy. The relational operators will mainly be overloaded in the meeting class which be helpful when adding into the balance tree. The == operator will check if the key word matches and the > and < will be used to determine which way to traverse in the tree to find the right key word. The + and += operator will be used to add responses to the meeting class.

So the major design that I want to have with these 5 classes is that at the very beginning of the program it will ask the user if they are going to post a meeting or respond to one, and create an object accordingly. It will then bring up the balance tree and allow the user to search for a meeting to respond too or create one. If they want to create one, numerous questions will be asked about the meeting and then it will be added to the tree with the user as the poster. However if the user wants to respond, they will search for a meeting based on a key word then add their response to that specific meeting.

To avoid “getter” function, I plan to not have the meeting class’s and person class’s try to change each other data members. The only interaction that these two classes will have is that they will call each other’s public member functions. This will prevent any information needing to be passed back and forth between the classes which would result in getter functions. Therefore I do not plan on the classes trying to change each other’s data members in any way.

As for the functions for the classes, I plan to have a display function for each class to make sure they are working as well as the standard copy constructors and destructors. For the meeting class there will be functions that will add a response and edit some of the members in the class. The person class will mainly have the standard functions because the main action that will be used for them is display. Lastly the node class will many function that will act with the balance tree. These will include a search (by keyword), add, and removeAll function.