CS163 Projects – Data Structures

myChat ADT

- A simple chat ADT and testing application that simulates popular functions in chat applications.
- Utilizes a node chain to connect chat rooms and another node chain in each chat room for the messages

Plane_Boarding_ADT

- Utilizes a queue as a circular linked list to add and remove passengers from a 'preboard' queue and then a sorted 'board' queue.
- Passengers from the sorted 'board' queue are then 'pushed' in sorted order into the plane 'stack', such that the person deepest into the plane boards first and then the lower rows. Passengers in each row also board in the order such that seats closer to the windows board first.
- The program is customizable to a plane of any 'halfrow' size - i.e there needs to be an even number of seats in each row and only a single aisle.
- The plane 'stack' is created as a linear linked list of arrays. Each node represents a row and its array the seats in that row.

Event_Management_hasht_ADT

- Uses a hashtable to store events hashed by its keywords. Each event contains 'n' keywords and therefore instanced 'n' number of times in the hashtable. Instead of storing copies of events for the different keywords, a single event instance is pointed to by several pointers from possibly different hash indexes.
- LLL chain used as a collision resolution technique.
- O(1) performance on average for event retrieval and addition
- A sample data file 'events_tej.txt' included for performance benchmarking

Event_Management_BST_ADT

- Essentially the same program as before, but this time using a binary search tree as the data structure.
- Search tree is maintained by event title rather than the keywords unlike before. This means that there aren't any duplicate events.
- The 'events_tej.txt' external data file with 73 titles is used here as well, and the height in accordance to these titles comes to 13. Therefore a relatively balanced performance can be expected with an average performance of O(log(2)(n)).