Jorge Castillo

COP3530

Programming Assignment I

What is the computational complexity of the methods in the implementation?

- insertEnd ~ O(1)
- insert ~ O(n)
- search ~ O(n)
- edit ~ O(n)
- delete ~ O(n)
- print ~ O(n)

Your thoughts on the use of linked lists for implementing a line editor. What are the advantages and disadvantages?

Compared to arrays, linked lists are excellent for dynamically inserting and deleting anywhere within the list. In my implementation, I added a tail pointer as well as a head to allow for constant time insertions at both ends. However, tasks that don't require dynamic resizing are not as efficient as arrays as linked list requires traversal from the head in order to obtain access (sequential access), whereas arrays have the advantage of immediately accessing any element in it (random access).

What did you learn from this assignment and what would you do differently if you had to start over?

I learned how to make a custom type in C++ compatible with range based for loops making iteration very simple for linked lists. In the current implementation, only functions that are not methods of the LinkedList class are actually utilizing this. If I had to start over, I would certainly make use of this form of iteration instead of sticking to a while loop that checks if the *curr* pointer is null or not. I have this while loop iteration in place for all methods with O(n), so it makes for a lot of repetitive and hard to read code that could easily break as linked list iteration is a little tricky.