Lauren Wonicker-Cook
Section: 12137

Programming Assignment 1

Computational Complexity

insertEnd

insertEnd has a computational complexity of O(n), with n being the number of elements in the list, because the method has to go through the entire list to change the reference of the last node to the new node.

insert

insert has a computational complexity of O(m), with m being the index at which the element is to be inserted into the list. The method traverses through the list until it gets to the index m, at which point the node is inserted.

delete

delete has a computational complexity of O(m), with m being the index at which an element is to be deleted from the list. The method iterates through the list until it reaches the m^{th} index, and the element is deleted and the reference of the previous node is changed to reference the next node.

edit

edit has a computational complexity of O(m), with m being the index of the line to be edited. The edit method traverses the list until it reaches m and then replaces the current line with a new line given by the user.

print

print has a computational complexity of O(n), with n being the number of elements in the list, because it prints all of the lines in the linked list.

search

search has a computational complexity of O(n), with n being the number of elements in the list, because it goes through the entire linked list to find all lines that contain the given line.

quit

quit has a computational complexity of O(n), with n being the number of elements in the list, because the destructor has to iterate through the linked list and delete each node.

Advantages and Disadvantages of Linked Lists

One of the advantages of using linked lists for a line editor is that it is extremely easy to insert or delete a line. To insert a line, the list must be iterated through to the end or a desired index, a line (node) object is then created, and then the next reference of the previous node is updated. To delete a line, the list must be iterated through to the desired index and then the next reference of the previous node is set equal to the next reference of the node being deleted. If an array were used, then all of the data would have to be shifted every time a line was inserted or deleted.

One of the main disadvantages of using a linked list for a line editor is that it takes longer to traverse the list. With an array, a simple for loop can be used with an integer iterator. To iterate through a linked list, a node object has to be used as an iterator and the location that the node references has to be changed at every iteration. Therefore, it is easier to insert at the end of an array, compared to inserting at the end of a linked list. It is also very difficult to iterate through a linked list in reverse order, whereas it simply takes decrementing the integer iterator to traverse an array backwards.

One disadvantage of the system that we designed was that it required very specific input and if the required input was not given, then no error would be thrown. The command had to have the exact capitalization, the line had to be in quotes and the line had to be less than 80 characters. Normally if improper input is given to a system, it throws an error. I think that it would be smarter to throw an error so that the user knew why the command they gave was invalid and could use the system properly.

What I Learned

I learned that I personally need to plan out my programs before I start writing them. My instructors have always said to write pseudocode before even writing one line of code, but I never found it necessary. This time that habit came back to bite me. I had to correct my main method multiple times because I did not account for edge cases when I first wrote my code. I also forgot to do one of the most basic things in object-oriented programming: create constructor methods. It took me a little while to realize why nothing would run, but once I did, it all became much easier. I also did not initially completely understand the >> operator for std::cin. I thought that it took in an entire line, but, after many frustrating errors, I noticed that it was only taking in the text before the first whitespace. I was able to correct that and then only ran into a few more errors. I think that I also need to start my programming assignments earlier so that I can flesh out the silly mistakes with enough time to handle the real ones. I think that this was a good exercise for working with linked lists, but it took more input parsing than I thought it would.