

Programming Assignment 1

Commentary

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

Linked list class:

- push_back: O(1)
- insert: O(n)
- erase: O(n)
- replace: O(n)
- print: O(n)
- search: O(n)
- begin: O(1)
- end: O(1)
- size: O(1)
- constructor: O(1)
- destructor: O(n)

Iterator class: all methods are O(1)

Line Editor class: the computational complexities of its methods are those of the corresponding Linked List class methods (since they are basically a call to the linked list methods)

The main “while” uses different methods to parse the input string (input from console), which are part of the STL and included in the headers: `<string>` and `<cstring>`

- atoi: O(n)
- strtok: O(n)
- strcmp: O(n)
- strcpy: O(n)
- find_first_not_of: O(n)

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

The use of a linked list for implementing a line editor has its pros and its cons.

Pros:

- Inserting and deleting lines doesn't require to shift subsequent lines, so these two operations offer better computational complexity (O(n)) than other forms of implementation like in an array
- Space in memory is allocated dynamically, so there's no need to worry about the size of the list being increased and having to reallocate the elements once the initial capacity is reached, like in an array based

Cons:

- The replace operation offers a worse computational complexity (O(n)) than that of an array based (O(1)) where the index of an element can be accessed in constant time

- The linked list implementation is more complex than that of an array based
- An improper use of pointers could lead to memory corruption, allocating space not needed or not releasing allocated space once it is not needed anymore
- The destructor offers a worse computational complexity ($O(n)$) than that of an array based, and the memory used by the list must be cleared once the object is out of scope

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

I learned how to implement a custom iterator for the linked list class. I also learned how to parse an input string and extract tokens from it, that were later sent as parameters to other functions.

I can't think of anything that I would change in my linked list implementation if I had to start over, though perhaps I would try to optimize or find a more elegant implementation for the functions that parse the input string