CMPT 125: Practice Midterm

Linked Lists

Suppose you have a singly-linked list whose nodes are defined like this:

```
    struct Node {
        int val;
        Node* next;
    };
```

There's a global variable `H` of type `Node*` that should always point to the first node of the list. If the list is empty, then `H == nullptr`. The `next` pointer of the last node on the list has the value `nullptr`.

The following questions ask you to write functions that process this singly-linked list. After each function is finished, `H` should either be `nullptr` (if the list is empty), or pointing to the first node of the list. You can write helper functions if necessary, but **don't** use any other data structures, such as arrays or vectors, in your answer.

a) Write two versions of a function called `pop_tail()` that deletes the last node on the list and returns its value. If the list is empty, use `cmpt::error` to cause an error. The first version of `pop_tail()` should use a loop, and the second version should use recursion.

b) Write a function called `get(n)` that returns the value of the nth node. The index of the first node is 0, so `get(0)` returns the value of the first node, `get(1)` returns the value of the second node, and so on.

If `n` < 0, or `n` >= (the number of nodes on the list), then cause an error with `cmpt::error`.

O-notation

a) Give the mathematical definition of "f(n) is O(g(n))".

b) Using the definition of O-notation, prove that n^2 is $O(2^n)$.

c) Prove or disprove: 500 is O(1).

d) Suppose algorithm A does $O(n^2)$ primitive operations when run on an input of size n. Experiments show that for n = 1000, it takes about 2 seconds to run. About how many seconds would you expect to wait for A to process an input of size n = 10,000?

e) Suppose f(n) is O(g(n)). Is it also possible that g(n) is O(f(n))? If so, give examples of functions for f and g that make it true. If not, explain why it's impossible.

ADTs

a) Define **abstract data type** (**ADT**).

b) Write an abstract data type `Queue` for a queue using C++. Include at least 5 basic functions, and use precise English to write the specifications.

c) Show how you can simulate a stack using only one queue. You only need to show how to implement push and pop in no worse then O(n) time. Don't use any other arrays, lists, stacks, etc. in your answer.