Casey Gale

CS 260

Assignment 2 (Design)

Designing a simple queue that can add objects to the end of the queue and remove objects from the front. LIFO structuring.

For this structure, I'm thinking it would be best to implement a linked list. Elements are constantly entering and leaving, so a linked list will allow for efficient implementation of this.

My first instinct is to have two member variables for the Queue class:

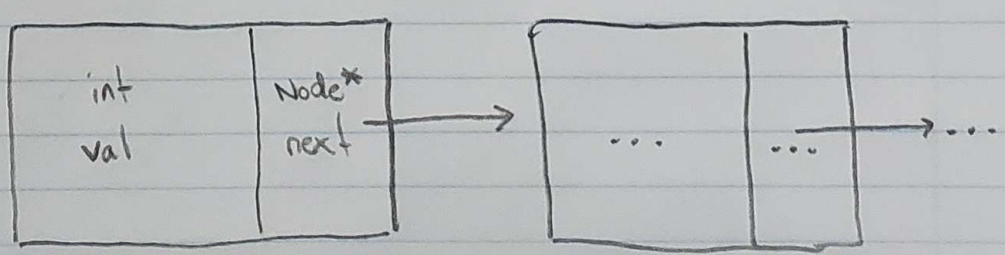
```
Node* head;
Node* tail;
```

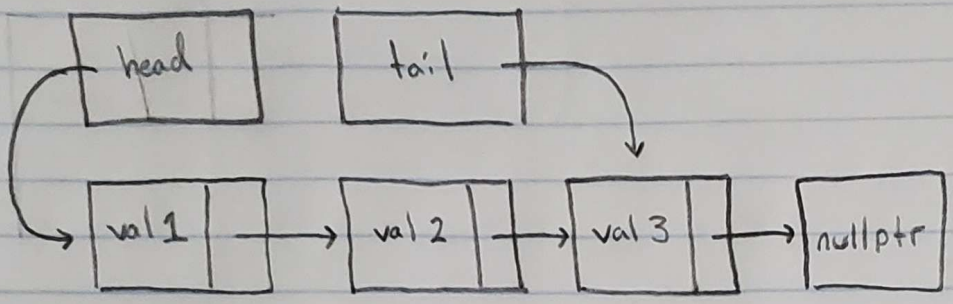
(with the class Node containing an int val and Node* next element)

The head pointer is a standard starting place for the linked list, while the tail pointer will always point to the last object in the queue.

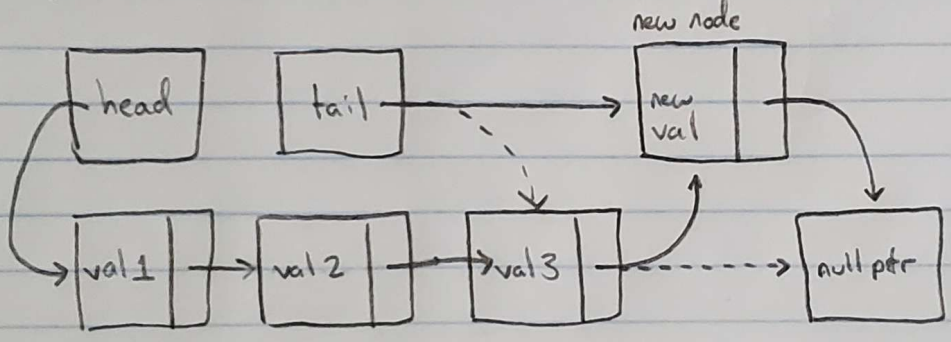
In theory, the tail pointer should reduce the time required to insert into the queue to constant time complexity, because you can always jump to the end of the queue.

Drawings will represent individual nodes as follows:



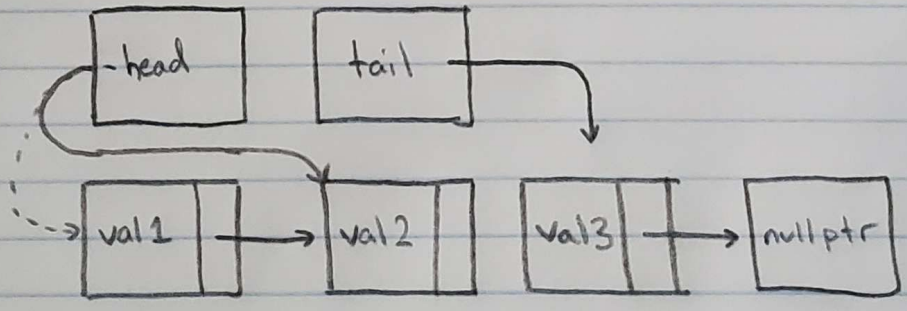


void push_back(new_val)



(dotted lines indicate previous pointer relations)

int pop_front()



some tracking of the old node must be done before changing pointer relations.