CSCI-2270
Recitation 6
Instructor: Boese

Stacks and Queues

Objectives

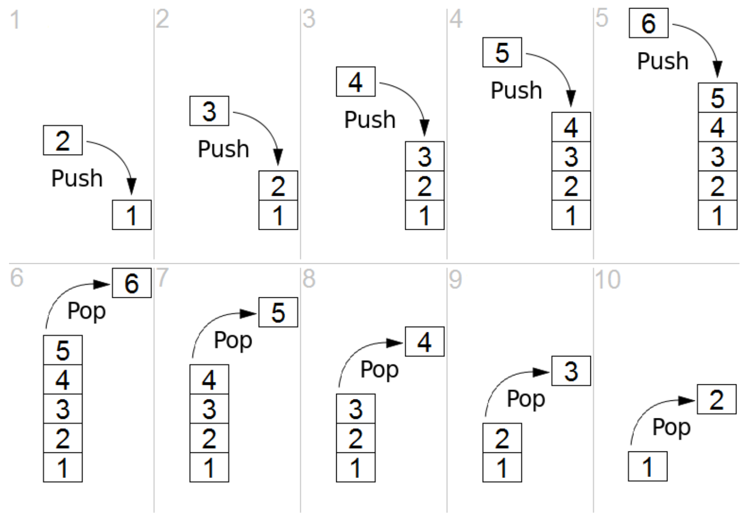
- Review the similarities and differences between stacks and queues
- Construct basic implementations of stacks and queues

In this recitation, you will review stacks and queues and then write basic implementations of both in C++. Stacks and queues are data structures that contain information stored and retrieved in a particular order, depending on the structure.

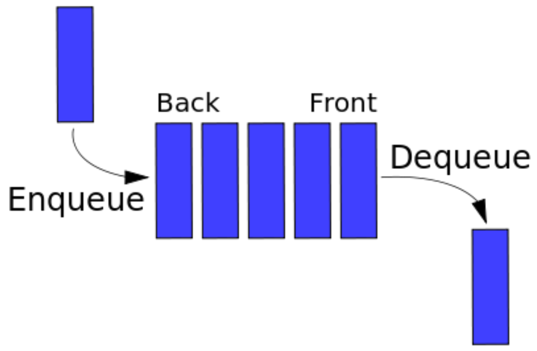
Both stacks and queues are sequential collections of elements, differing in how elements are accessed. Stacks use a last out, first in (LIFO) method for accessing elements, whereas queues use a first out, first in (FIFO) method.

Questions: Describe what LIFO and FIFO mean. Describe a tangible (i.e., "real-world") example of both stacks and queues.

Items are added to a stack via a *push* operation and removed from a stack via a *pull* operation:



Items are added to and removed from a queue through *enqueue* and *dequeue* operations:



Images from https://techwelkin.com/differences-between-stack-and-queue

Other helpful functions for both stacks and queues are *peek*, which returns the value of the item at the beginning of the list without removing it from the list, and isEmpty, which checks if a list is empty.

Questions: Stacks and queues can both be implemented using arrays or linked lists. Why might you use one data structure over the other in your implementation? With a linked list implementation, what do you need to keep track of in a queue that you don't need to keep track of in a stack? Why?

Programming Exercise
Create classes for both stacks and queues using linked lists for your implementation. Stacks should have methods for push, pop, peek, and isEmpty. Queues should have methods for enqueue, dequeue, peek, and isEmpty. Use the following struct for list nodes:

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

To get credit for this lab exercise, show the TA your work and sign the lab's completion log.