Day's Goals

Wednesday, February 14, 2018 9:24 AM

Stack

Finish SLL implementation
Recap ADP
Array Impl.

Queue

- high level
- ADP
- skip LL
- array
  ↳ two approaches

Stack Abstract Data Type

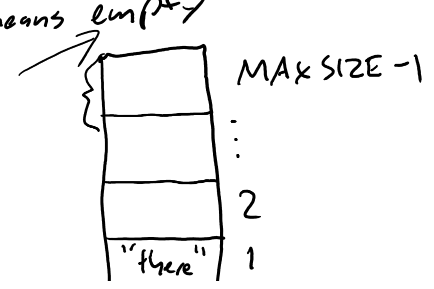
e.g.

```
private:
    top      // keeps track of "top" element
    maxSize  //
    count    //

public:
    initialize() (constructor)
    isFull( )
    isEmpty( )
    push ( )
    pop ( )
    disp ( )
```

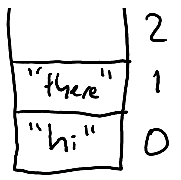
Array Impl.

```
private:
    int top, count;
    string a[MAXSIZE];
```

top==0 means empty



string a[MAXSIZE];

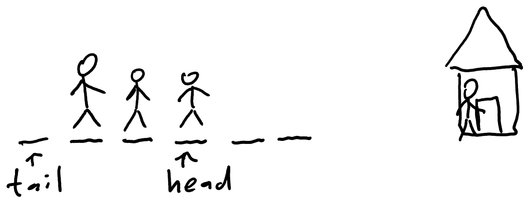


top = 2

```
public:
    constructor ( )
    {
        top = 0;
        ~~count = 0;~~
    }

    bool isFull( )
    {
        check if
        top == MAXSIZE
    }
    bool isEmpty( )
    {
        if top is 0,
        we know stack is empty
    }

    void push(newItem)
    {
        if (!isFull())
        {
            a[top] = newItem;
            top++;
        }
        else
        {
            "stack overflow"
        }
}

String   pop()
{       out="";
    if (!isEmpty())
        out =a[top];
      top--;
    else
      "Stack is empty"

    return  out

  }
```

Queue

- Similar to stack
- again, allows for specific order of operations



- enqueue : - to add new element
  - always added @ tail
- dequeue : - to remove element
  - always remove from head

First In, First Out (FIFO)

Sample Application

- Read and write cmnds get queued up in fast DRAM

Queue ADP

e.g.

```
private:
    head
    tail
    count
    maxQsize
public
    init( )
    isEmpty( )
    isFull( )
    enque( )
   value deque( )
```

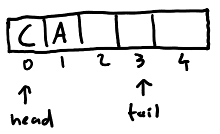
Implementations

1) LL

2) Array
   2 approaches

   I) Simple "linear" que
      e.g.

B



q.enque(A)

q.deque() // remove item from head

Worst case for deque operation is when que is full.

O(n)

II Circular array Que

Allow for both head and tail to shift when dequeing and queing.

Queue

Wednesday, February 14, 2018 9:24 AM