CSCI 2270 Lecture Notes

2/8/2019

* Continuation of Stacks
  + Can be implemented as either an array or a linked list
* Stack Abstract Data Type (does not specify implementation)
  + private:

top

maxSize

count

public:

isFull() //only applicable if count and max size

isEmpty()

push(item) // adds item to top

pop()//returns item on top

display()

* SLL Stack Implementation
  + a special case of SLL
  + don’t have to set a max size
  + top is a pointer to Node type
  + stack is empty if top == nullptr
  + push adds new nodes to the stack
    - this is like inserting a Node at the head
  + pop removes Node from the top and returns this Node
    - could be defined differently
* bool isEmpty()
  + if top is null
    - return true
  + else
    - return false
* push(item)
  + create a new node
  + if(isEmpty)
    - point top to new Node
    - set top->next to nullptr
  + else
    - temp->next = top
    - top = temp
    - temp = nullptr
* Node \*pop()
  + if(!isEmpty)
    - temp = top
    - top = top->next
    - return temp
* Array Implementation of a Stack
  + private:

int top, count; //top is the INDEX of the next available spot in stack

string a[MAXSIZE];

* + bool isFull()
    - check if top == MAXSIZE
  + bool isEmpty()
    - check if top == 0
  + void push(newItem)
    - if(!isFull())
      * a[top] = newItem
      * top++
    - else
      * “stack overflow”
  + string pop()
    - if(!isEmpty())
      * out = a[top];
      * top--;
      * return out
    - else
      * “stack is empty”
* Queues
  + similar to stack
  + again, allows for specific order of operations on your structure
  + enque
    - add a new element
    - can only add at the tail
  + deque
    - remove an element
    - can only remove from head
  + First in, first out
    - FIFO method as opposed to LIFO method
* Implementations
  + LL
  + Array
    - Simple linear
    - Circular array