

For Monday

- Read Weiss, chapter 4, sections 1-2
 - The concepts here should be review.
- Homework:
 - Complete the List Practice described on Blackboard under Notes and Homework. This is written work. Only turn in the requested pieces.

Late Passes

- You have 2 for the semester.
- Only good for programs and papers.
- Allow you to hand in up to 5 days late IF you have a late pass left.
- Each good for $+.05$ on final grade if unused.
- Must acknowledge the use of the late pass.
- Only way to turn in late work in this course.

Programming Assignment 1

Creating Nodes

- We'll always create nodes dynamically.
- Note that we almost never actually work with a node itself; we use pointers to the nodes.
- Important:
 - ALWAYS initialize next to NULL when a node is created unless you are immediately assigning it another meaningful value.

C++ Details

- Deleting nodes . . .

Linked List Variations

- Doubly-linked lists
- Empty head node

Stacks

- What is a stack?
- What would a stack ADT look like?
- How could you implement a stack?
- What are stacks used for?

Stack ADT

- **AbstractDataType** *Stack* {
 instances
 linear list of elements; one end is
 the *top*
 operations
 Create()
 Destroy()
 IsEmpty()
 IsFull() (not always needed)
 Top()
 Push(element)
 Pop()
}

Queues

- What is a queue?
- What would a queue ADT look like?
- How could you implement a queue?
- What are queues used for?

Queue ADT

- **AbstractDataType** *Queue* {
 instances
 linear list of elements; one end is
 the *front*, the other the *rear*
 operations
 Create()
 Destroy()
 IsEmpty()
 IsFull() (not always needed)
 Front()
 Add(elem) (or Put(elem) or Enqueue(elem))
 Delete() (or Get() or Dequeue())
}

A Note on Push and Pop

- In this class, do **not** use push and pop to refer to queue operations.
- They should **only** apply to stack operations.

Queues and Stacks

- Often used in similar contexts to achieve different desired results.
- Often used in conjunction with other data structures.