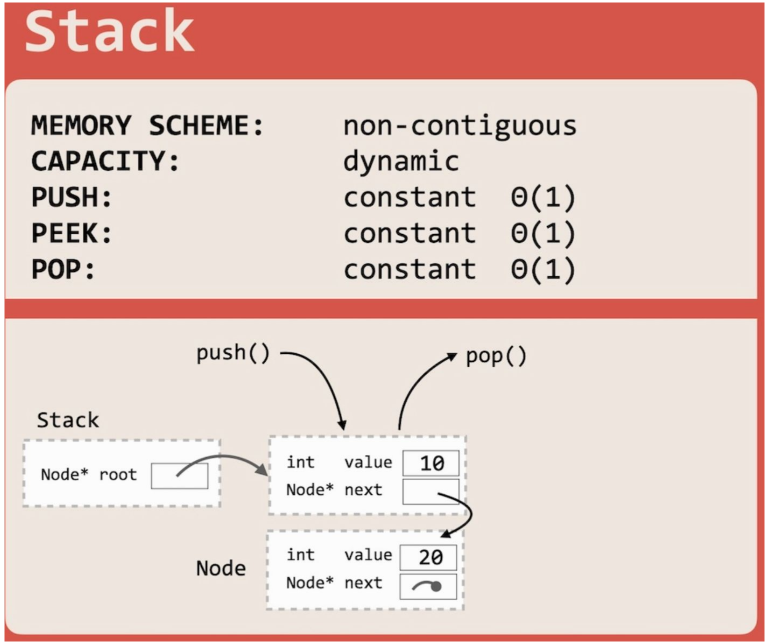
Discussion 6(11/9)

ECE 17

Stacks

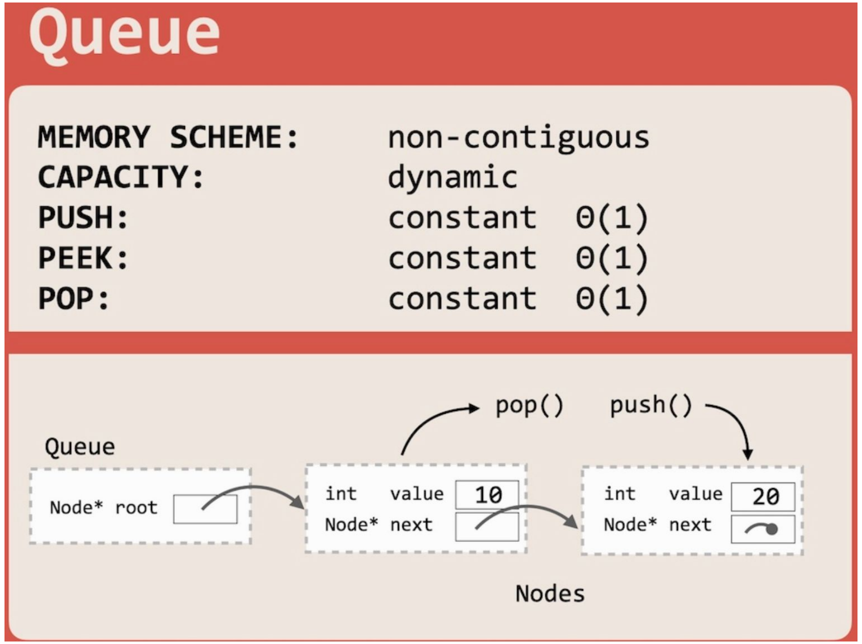
- grows from bottom up
- Last-in-first-out
- imagine a stack of books
    - you add to the top of the stack
    - you remove books from the top


Stack

MEMORY SCHEME:	non-contiguous	
CAPACITY:	dynamic	
PUSH:	constant	Θ(1)
PEEK:	constant	Θ(1)
POP:	constant	Θ(1)

Queues

- grows top down
- First-in-first-out
- imagine a todo list
    - you add to the bottom of the list
    - you do what is on the top of the list



Queue

MEMORY SCHEME:	non-contiguous
CAPACITY:	dynamic
PUSH:	constant Θ(1)
PEEK:	constant Θ(1)
POP:	constant Θ(1)

Templating Linked List

- improves reuse and flexibility

- Adds a new dimension to extensibility

Templating Linked List

- Templates are reusable, type -generic code
    - you specify a type placeholder
    - also provide generic code inside the template

Templating Linked List

- For linkedList template, we are templating a class

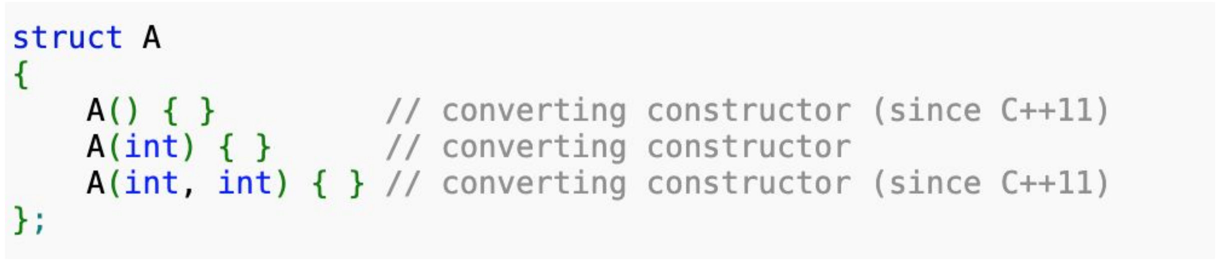
- be sure to follow the hw instructions

Visitor

- read the hw instructions carefully
- this is a pattern
    - used to provide a way to perform some action on all the elements in your container, without needing to change your container

Conversion Constructors

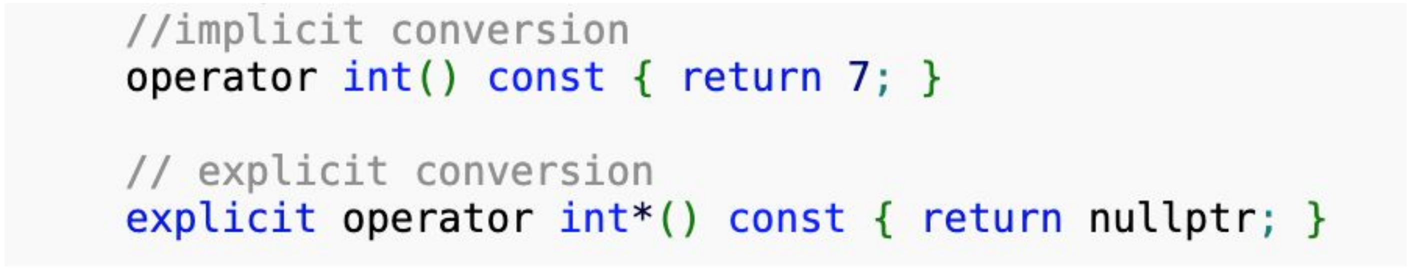
- constructor that isn’t specifying anything explicit



```
struct A
{
    A() { }         // converting constructor (since C++11)
    A(int) { }      // converting constructor
    A(int, int) { } // converting constructor (since C++11)
};
```

Conversion Operators

- Conversion function is declared like a non-static member function or member function template with no parameters, no explicit return type



```cpp
//implicit conversion
operator int() const { return 7; }

// explicit conversion
explicit operator int*() const { return nullptr; }
```

Any questions?