


Lecture 34: More C++

The graphic consists of a large blue square on the left containing a white plus sign. To its right are four smaller squares arranged in a 2x2 grid: orange (top-left), green (top-right), purple (bottom-left), and red (bottom-right).

+ Today



- Reading
 - Weiss Ch. 6
- Objectives
 - Recap lessons learned from Wednesday's lab
 - The Big Three
 - Access modifiers

The slide features a blue plus sign followed by the word 'Today' in a bold blue font. To the right is a vertical decorative bar with a thin green line on the left and a wider blue section on the right. The content is organized into two main sections: 'Reading' and 'Objectives', each with a blue square bullet point and a list of sub-items.

+ Announcements

- Final exam is scheduled for
Wednesday May 14th at 9am
- Do not purchase your plane tickets before this day/time!
- Seniors will take the exam early during the last week of classes (before the 8th of May)

+ This week's assignment

- Don't wait until the last day to start this assignment
- The parameter for the `ifstream` constructor is a `char*` not a string
- To convert from `string` class to primitive `char*` use the `c_str` function

```
ifstream tree_file(filename.c_str());  
assert(tree_file.is_open());
```

+ Lessons from Lab

- Shallow copying vs. deep copying
- The default copy constructor and `operator=` provide a shallow copy!
 - They copy the instance variables but not any memory pointed to by the instance variables
 - Consequences include memory leaks and (possible) runtime errors
- If an instance variable is a pointer to heap-allocated memory, need to overwrite the big three

+ Classes in C++: The Big Three

- The Big Three
 - destructor
 - copy constructor
 - `operator=` function
- These special functions are already written for you!
- Rule-of-thumb: If you need to overwrite one of these, overwrite them all!
- To disallow a default copy constructor and `operator=`, declare private ones that do nothing

+ Default constructor in C++

- If you define no constructors
 - C++ generates a default parameter-less constructor
- If you define constructors but none are parameter-less
 - C++ does not generate a default parameter-less constructor
- Examples of when the default constructor is called:
 - `MyClass x;`
 - `MyClass* x = new MyClass();`

+ Access Modifiers in C++

- In Java: public, protected, private, and “default”
- In C++: public, private, and friend
- Friends are allowed access to the private section of class

+ Access Modifiers in C++

```
class Node{

    private:
        int element;
        Node *next;

        // the constructor is private!
        Node(int theElement, Node*n)
            : element(theElement), next(n){ }

        // the integer queue class is the only
        // class that can create Node variables
        friend class IntegerQueue;

};
```

+ Access Modifiers in C++

```
class Node{

    private:
        int element;
        Node *next;

        // the constructor is private!
        Node(int theElement, Node*n)
            : element(theElement), next(n){ }

        // only the enqueue function from the
        // IntegerQueue class is a friend
        friend void IntegerQueue::enqueue(int x);

};
```