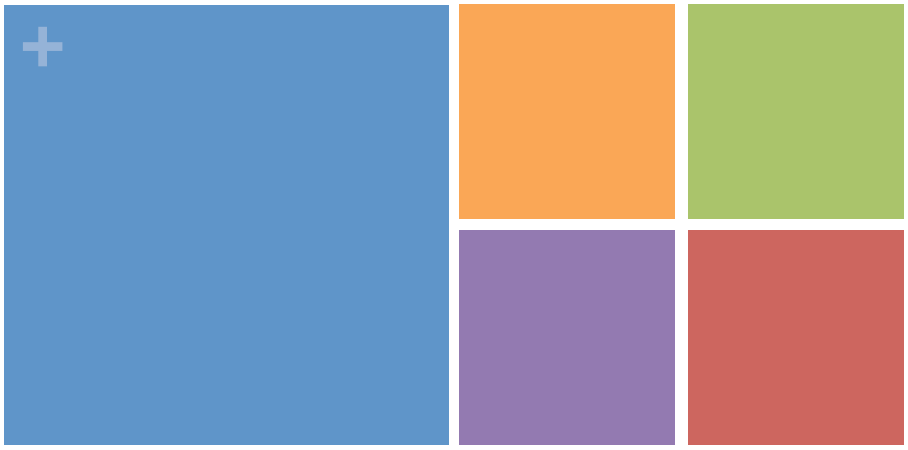

Lecture 33: More C++

Today

- Reading
  - Weiss Chapter 4 and 6
- Objectives
  - This week's assignment
  - The Big Three

This Week's Assignment

- Create a Twenty-Questions Animal Game
- Split into two parts
  - Part 1 is due *Tuesday* April 22nd
  - Part 2 is due *Tuesday* 29th
- Part 1 is writing the I/O functions
- Part 2 is implementing a BinaryTree class and implementing the main function that controls the game

Recap

```
void swap(int &m, int &n){
    int temp = m;
    m = n;
    n = temp;
}
```

formal parameters refer to actual input arguments not copies!

- A pointer is a variable that stores the memory address of another entity (e.g. variable)
- Call-by-value versus call-by-reference
- C++ allows call-by-reference using the & symbol
- *A reference is a pointer to a variable that is automatically dereferenced*
  - *The reference is just an alias for the original variable*

Classes in C++: The Big Three

- The Big Three: destructor, copy constructor, `operator=`
- These special functions are already written for you!
- Rule-of-thumb: If you need to overwrite one of these, overwrite them all!
- Destructor
  - Called when object goes out of scope or is deleted
  - Frees resources (e.g. deletes memory, closes files)

Copy Constructor

- Constructs a new object from an existing object
- Examples of when the copy constructor is called:
  - `IntCell copy = original;`
  - `IntCell copy(original);`
  - `a formal parameter of a call-by-value function`
  - `an object returned by value`
- It would not be called in this instance:

```
IntCell copy;

copy = original;
```

operator=

- Assignment for two *already constructed* objects
- Example usage,

```
IntCell first(3);

IntCell scnd;

scnd = first;
```

The Big Three for IntCell class

```
class IntCell {
public:
	// destructor
	~IntCell() {
		// does nothing
	}

	// copy constructor
	IntCell(const IntCell & rhs) : value(rhs.value) {
		//does nothing
	}

	// operator= on next slide
private:
	int value;
};
```

The Big Three for IntCell class

```
// assignment operator
IntCell & operator=(const IntCell& rhs) {
   if(this != rhs){  // alias test
      value = rhs.value;
   }
   return *this;   // *this is the actual object
}
```

The Big Three for IntCell class

- The `this` keyword has the same functionality as in Java
  - `this` is a pointer to the current object
- The alias test is extremely important
  - disallows x = x
- Function returns a reference to object
  - allows for a = b = c

The Big Three with Pointers

- If your class contains a pointer to heap-allocated memory
- Default destructor does not call delete
- Default copy constructor and operation= functions only copy the value of the pointer (not the data pointed to)
  - Called a shallow-copy
- A deep-copy is when an entirely separate copy is made
  - Often times, this is what we really want!

The Big Three for IntCell class

- Change the IntCell class to hold a pointer to an integer
- See `intcell_default.h` and `intcell_bigthree.h`
  - One uses the default big three
  - The other (`intcell_bigthree.h`) correctly overwrites the big three
- Miscellaneous final comments
  - To disallow default copy constructor and operator=, move to private section
  - The `this` keyword has type `Class* const` (can't modify)

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