Last Time

- Constructors / Deconstructors
- Class Inheritance
- Polymorphism
  - Virtual Functions
Copy Constructor (again)

- When objects are passed to functions or returned, they are by default passed by value; a copy needs to be created
  - member-wise copying of object (assign each member)
Member Initializer Syntax

- Assign values before default values are assigned
- const data members and data members that are references must be initialized with this
- All data members can be initialized with this
- Member initializer list executes before constructor body
- Member objects either initialized with member initializer or member object's default constructor
Static Data Members

- Classes have only 1 copy of static data members whereas object instances each have their own copy of non-static data members
  - Object instance size determined by non-static members
  - Static member initialization
    - Initialized only once, only static members can be initialized in class definition (.h)
    - static members of fundamental types initialized to 0
- Access using className::staticDataMemberName (can use a particular object instance name if any exist)
Static Functions

- MyClass::myFunction(); /// doesn't need an instance to work
- myclass.myFunction(); /// but it will work if we have an instance
Friends

- Allows selected classes to access protected and private data
  - Class must declare its friends
  - Declare them usually in the line after making your class.
- Friend functions
- Friend classes
this operator

- Instances have access to their own address through the keyword "this"
- "this" pointer passed as an implicit argument to every non-static member function
- It's a const pointer to the type of class (MyClass* const)
- Use like a pointer this->memberVariable, this->method()
Operator Overloading

- Often the syntax of code can be more readable if our classes can use operators instead of functions
  - Example: most mathematical classes
- We can't create new operators, only overload existing
- Basic Guidelines
  - Only use when the "meaning" of the operator for the class is clear
  - Try to use all of the similar relations, if you have + +MyClass, have MyClass++
Operator Overloading

- What operators can I use?
- Some of the operators
  - new new[] delete delete[]
  - = [] -> ,
  - * & ()
  - + - * / %
  - += -= *= /= %=
  - + -, ++ --
  - & | ^ << >> ~
  - &= |= ^= <<= >>==
  - == != < > <= >= || && !
Using `>>` and `<<`

- friend istream& operator>>(istream& is, Person& s);
- friend ostream& operator<<(ostream& os, const Person& s);