Lecture 06
Classes (Inheritance)

- Derived class
  - Has all attributes of base class.
  - Can access non-private members of base class.
  - Protected members of base class are accessible to members and friends of any derived classes.
  - Does not inherit constructor or destructor of base.
  - Can redefine member functions of the base class

- Size of derived class = non-static data members of derived class + non-static data members of base class (even if private)

- C++ requires derived class constructor to call base class constructor.
class Person{
    public:
        string name;
        void print(){ /* some code to print */}
}

class Student : public Person{
    public:
        void print(){ /* some code to print a student */}
}

Classes (Constructor / Deconstructor)

- **When we instantiate a derived class:**
  - Base classes member object constructors execute (if they exist)
  - Base class constructor executes
  - Derived classes member object constructors execute
  - Derived class constructor executes

- **Destructors called in reverse order.**

- **Base class constructors, destructors and overloaded assignment operators are not inherited by derived classes. However derived class can call base classes version of these.**
Classes (Polymorphism)

- Polymorphism (many shapes)
- Example: Say we have two classes
  - Class Person;
  - Class Student : public Person{};
  - If we have a function void f(Person p) we can pass a Student
- This works for references and pointers as well
Classes (virtual functions)

- If we want to use the underlying classes method (use virtual)
  - This is called dynamic dispatch
- Allows objects to have many types
  - Enables the compiler to call the more specific type
- virtual void print(){...}
Classes (pure virtual)

- When the base class defines the method with no implementation
  - forces derived classes to provide the implementation
- virtual const string print() = 0; /// All virtual
c++ allows multiple base classes

- class Student : public Person, public Academic
- Potentially messy
  - If both Person and Academic have the same member you must distinguish.
    - Person::x or Academic::x
  - If both Person and Academic have are both derived from a common class, you get two instances of the common class (diamond problem)
- Coding practice says try to avoid unless really necessary (and you know what you are doing)