C++ Classes

access specifiers
private: // only class access
protected: // only class and subclasses can access
public: // anyone can access

friend
Used to allow access to private or protected members by another specified function or class
Often used for operator< and operator<< etc.
friend class X; // class X is a friend
friend int f(int); // function f is friend

struct and class
struct defaults everything to public
class defaults everything to private

Typical class declaration and definition

// X.h
class X {
public:
  void f(int i);
}; // X

// X.cpp
void X::f(int i) {
  cout << i;
}

Definition and declaration at same time
This is required when declaring a templated class using g++

// X.h
class X {
public:
  void f(int i) {
    cout << i;
  }
}; // X

This way “inlines” the function definitions

Subclassing
access on base classes specifies access by users of the subclass to superclass members
must use virtual member function declaration to allow member to be overridden in subclass
pure virtual member functions can be overridden but do not need a definition in the base class
Any class that has a pure virtual member function cannot be instantiated
but a subclass can be instantiated if it has no pure virtual members
Such a class is an abstract class

class Super {// abstract class-pure virtual member
  int f(int i); // not overrideable
  virtual int g(int i); // virtual - can override
  virtual int h(int i) = 0; // pure virtual
};

class Sub : public Super {
  virtual int f(int i); // not overriding
  virtual int g(int i); // overriding
  virtual int h(int i); // overriding
};

Special class members
default constructor
default copy constructor
default operator=
default destructor