MemberWise Initialization L/L 14.5

So far we have seen how to initialize data members of an object *inside* the constructor body.

We can also initialize *outside* the constructor body using a *member initialization list.* Such a list explicitly initializes members that are given in a comma-separated list of member-name/argument pairs. As an example, we can replace

```
CC::CC(int x, int y)
    { Real =x; Imag =y;}
```

by

```
CC::CC(int x, int y): Real(x), Imag(y)
  {}
```

As another example suppose

```
class thing
{  public:
    thing(int x_val, int y_val, int z_val );
   private:
    int* x;
    int  y, z;
};
```

The definition of the constructor could be rewritten as

```
thing::thing(int x_val,int y_val,int z_val):
          y(y_val), z(z_val)
{ x = new int; *x = x_val;}
```

Note that the static members are initialized in the list but the dynamic member is not. Why?

Our uses of the *member initialization list* have been pretty simple so far. We will see its real importance later, when we study inheritance.