void Stock::update(double price)
{
    ...
}

In short, within a class declaration or a member function definition you can use an unadorned member name (the unqualified name), as when `sell()` calls the `set_tot()` member function. A constructor name is recognized when called because its name is the same as the class name. Otherwise, you must use the direct membership operator (.), the indirect membership operator (->), or the scope resolution operator (:), depending on the context, when you use a class member name. The following code fragment illustrates how identifiers with class scope can be accessed:

```cpp
class Ik
{
    private:
        int fuss;       // fuss has class scope
    public:
        Ik(int f = 9) { fuss = f; }  // fuss is in scope
        void ViewIk() const;       // ViewIk has class scope
    };

    void Ik::ViewIk() const  //Ik:: places ViewIk into scope
    {
        cout << fuss << endl;  // fuss in scope within class methods
    }

    ...

    int main()
    {
        Ik * pik = new Ik;
        Ik ee = Ik(8);  // constructor in scope because has class name
        ee.ViewIk();   // class object brings ViewIk into scope
        pik->ViewIk(); // pointer-to-Ik brings ViewIk into scope
        ...
    }
```

**Class Scope Constants**