EXAMPLE 10.14  A static Data Member

The Widget class maintains a static data member count which keeps track of the number of Widget objects in existence globally. Each time a widget is created (by the constructor) the counter is incremented, and each time a widget is destroyed (by the destructor) the counter is decremented.

```cpp
class Widget
{
public:
    Widget() { ++count; }
    ~Widget() { --count; }
    static int count;
};
int Widget::count = 0;
```

```cpp
ing main()
{
    Widget w, x;
    cout << "Now there are " << w.count << " widgets.\n";
    Widget w, x, y, z;
    cout << "Now there are " << w.count << " widgets.\n";
    cout << "Now there are " << w.count << " widgets.\n";
    cout << "Now there are " << w.count << " widgets.\n";
}
```

Notice how four widgets are created inside the inner block, and then they are destroyed when program control leaves that block, reducing the global number of widgets from 6 to 2.

A static data member is like an ordinary global variable: only one copy of the variable exists no matter how many instances of the class exist. The main difference is that it is a data member of the class, and so may be private.

EXAMPLE 10.15  A static Data Member that is private

```cpp
class Widget
{
public:
    Widget() { ++count; }
    ~Widget() { --count; }
    int numWidgets() { return count; }
private:
    static int count;
};
```

```cpp
int Widget::count = 0;
```

```cpp
ing main()
{
    Widget w, x;
    cout << "Now there are " << w.numWidgets() << " widgets.\n";
    Widget w, x, y, z;
```