类的静态数据成员

- 在类的成员前如果加上关键字static修饰的成员就是类的静态成员。类的静态成员包括静态数据成员和静态成员函数。类的静态成员的特点是：
- A.静态成员属于类，不属于任何对象。
- B.静态成员函数没有this指针。因此，静态成员函数不能访问一般的数据成员，它只能访问静态数据成员，也只能调用其他的静态成员函数。
- C.无论对象是否存在，类的一个静态数据成员都只有一个，存于公用内存中，可被该类的所有对象共享。

- 静态数据成员的声明：在类定义中的数据成员声明前加上关键字static，就使该数据成员成为静态数据成员。静态数据成员可以是public（公有）、private（私有）或protected（保护）。

- 【例2-12】静态数据成员示例。在下面定义的Student类中，m_averageAge表示所有学生的平均年龄。由于平均年龄不属于每一个学生的属性，但属于一个Student类要维护的一个属性，所以，将Student类中的数据成员m_averageAge声明为静态数据成员。m_totalNumber表示学生总数，类似地，也要声明为静态数据成员。

```
class Student
{
public:
    static double m_averageAge; // 所有学生平均年龄
    static int m_totalNumber;       // 学生总数，
    Student(char *pname, double age); //构造函数
    ……
private:
    char m_name[20];                //姓名
    double m_age;                   //年龄
};
```

- 类设计者需要在类外对该类的静态数据成员进行定义，其定义形式如下：
- <类型><类名>::<静态数据成员名>[=<初值>];
- 例如，上面的静态数据成员m_averageAge的定义如下：
- double Student::m_averageAge = 0;
- 如同一个成员函数在类外定义一样，静态数据成员的名字必须通过作用域运算符“::”被其类名限定修饰。

- 提示：
- A. 程序中，对静态数据成员的声明在类内进行，对一个静态数据成员的定义和初始化必须在类外进行，且只能出现一次。
- B. 静态数据成员定义时前面不要加关键字static。
- C. 在多文件结构中，静态数据成员定义和初始化最恰当的地方，是将它放在一个类的实现文件中。

- 静态数据成员的访问:
- 类的公有静态数据成员的一般访问形式：
- <类名>::<静态数据成员名>
- 也可以是：
- <对象名>.<静态数据成员名>
- 或
- <对象指针>-><静态数据成员名>
- 后两种访问方式中的“对象名”或“对象指针”只起到类名的作用，与具体对象无关。

- 例如，设student1是Student类的一个对象，ps是指向对象student1的指针变量，要输出Student类的静态数据成员m_averageAge的值，可以用下面的语句：
- cout<<Student::m_averageAge;
- 或 cout<<student1. m_averageAge;
- 或 cout<<ps->m_averageAge;
- 如果Student类的静态数据成员m_averageAge被声明为私有成员，就不能用上面的方法直接访问该静态数据成员了，而需要使用类提供的公有静态成员函数来间接地访问静态数据成员。