5-1作用域：一个标识符在程序正文中有效的区域。

类型：函数原型作用域、局部作用域、类作用域、命名空间作用域

5-2 从标识符引用的角度，来看标识符的有效范围，即为标识符的可见性。程序运行到某一点，能够引用到的标识符，就是该处可见的标识符。

一般规则：

标识符要声明在前，引用在后。

在同一作用域中，不能声明同名的标识符。

在没有互相包含关系的不同的作用域中声明的同名标识符，互不影响。

如果在两个或多个具有包含关系的作用域中声明了同名标识符，则外层标识符在内层不可见。

5-5 静态数据成员：类的静态数据成员是类的数据成员的一种特例，采用static关键字来声明。对于类的普通数据成员，每一个类的对象都拥有一个拷贝，就是说每个对象的同名数据成员可以分别存储不同的数值，这也是保证对象拥有自身区别于其他对象的特征的需要，但是静态数据成员，每个类只要一个拷贝，由所有该类的对象共同维护和使用，这个共同维护、使用也就实现了同一类的不同对象之间的数据共享。

5-6 静态函数成员：使用static关键字声明的函数成员是静态的，静态函数成员属于整个类，同一个类的所有对象共同维护，为这些对象所共享，静态函数成员具有以下两个方面的好处，一是由于静态成员函数只能访问同一个类的静态数据成员，可以保证不会对该类的其余数据成员造成负面影响；二是同一个类只维护一个静态函数成员的拷贝，节约了系统的开销，提高程序的运行效率。

5-9 类B不是类A的友元，类C不是类A的友元，类D不是类B的友元。

5-10 可以

Private:

Static int a;

6-6 \*称为指针运算符，是一个一元操作符，表示指针所指向的对象的值；&称为取地址运算符，也是一个一元操作符，是用来得到一个对象的地址。

6-7 指针是一种数据类型，具有指针类型的变量称为指针变量，指针变量存放的是另外一个对象的地址，这个地址中的值就是另一个对象的内容。

6-9 NULL字符

6-11 引用是一个别名，不能为NULL值，不能被重新分配；指针是一个存放地址的变量。当需要对变量重新赋以另外的地址或赋值为NULL时只能使用指针。

6-15 const int \* p1声明了一个指向整型常量的指针p1，因此不能通过指针p1来改变它所指向的整型值；int \* const p2声明了一个指针型常量，用于存放整型变量的地址，这个指针一旦初始化后，就不能被重新赋值了。

6-18 此程序中给\*p分配的内存没有被释放掉

改：

#include <iostream>

Using namespace std;

Int \*fn1(){

Int\*p=new int(5);

Return p;

}

Int main(){

Int \*a=fn1();

Cout<<”the value of a is:”<<\*a;

Delete a;

Return0;}

6-19

long(\*p\_fn1)(int);

long(A::\*p\_fn2)(int);