5-1

作用域讨论的是标识符的有效范围，作用域是一个标识符在程序正文中有效的区域。

C++的作用域分为函数原型作用域，块作用域，类作用域和文件作用域。

5-2

可见性是标识符是否可以引用的问题。

可见性的一般规则是：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。对于在不同的作用域声明的标识符，遵循的原则是：若有两个或多个具有包含关系的作用域，外层声明的标识符如果在内层没有声明同名标识符时仍可见，如果内层声明了同名标识符外层标识符不可见

5-5

在c++类中声明成员时可以加上static关键字，这样声明的成员就叫做静态成员。

普通数据成员属于类的一个具体的对象，只有对象被创建了，普通数据成员才会被分配内存。而静态数据成员属于整个类，即使没有任何对象创建，类的静态数据成员变量也存在。因为类的静态数据成员的存在不依赖与于任何类对象的存在，类的静态数据成员应该在代码中被显式地初始化，一般要在类外进行。虽然静态成员不属于类的某个对象，但是我们仍然可以使用类的对象、引用或指针来访问静态成员

5-6

函数调用的结果不会访问或者修改任何对象（非static）数据成员，这样的成员声明为静态成员函数. 静态成员函数的声明除了在类体的函数声明前加上关键字static，以及不能声明为const或者volatile之外，与非静态成员函数相同。出现在类体之外的函数定义不能制定关键字static，静态成员函数没有this指针

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;

5-16

编译的输入文件是源文件，输出是目标文件；连接的输入文件是目标文件，输出是可执行文件。

编译器对源代码进行编译，是将以文本形式存在的源代码翻译为机器语言形式的目标文件的过程。连接是将格格编译单元的目标文件和运行库当中被调用过的单元加以合并后生成的可执行文件的过程

1. 编译时报错，函数参数不匹配
2. 连接错误，函数未定义
3. 不报错
4. 连接错误，函数重复定义

6-6

“\*”被称为指针运算符，是一个一元操作符，表示指针所指向对象的值。“&”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成为指针变量。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外一个对象中的内容。

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

整型常量指针p1

整形指针常量p2

6-18

#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;

return 0;

}

6-19

long (\* p\_fn1)(int)

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