5-1

作用域：一个标志符在程序正文中有效的区域。

分为：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

可见性：讨论一个标志服是否可以被引用。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。

可见性的一般规则:

* 标识符要生命在前，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生命了同名标识符，则外层标识符在内层不可见。

5-5

静态数据成员：即类属性，描述类的所有对象共同特征的一个数据项，对于任何对象实例，它的属性值是相同的。

特点：具有静态生存期；可以通过类名进行访问；该数据为整个类所共有，不属于任何一个对象。

5-6

静态函数成员：用static关键字声明的函数成员，属于整个类，有同一个类的所有对象所共同拥有的函数成员。

特点：可以直接防伪该类的静态成员。

5-9

类B不是类A的友元（友元关系不具有双向性）

类C不是类A的友元（友元关系不具有传递性）

类D不是类B的友元（友元关系不能被继承）

5-10

可以：

private:

static int i;

5-16 ？

编译：输入源文件，输出目标文件

连接：输入目标文件，输出可执行文件

不同：编译是将文本形式的源文件转换成描述程序在运行过程中需要放在内存中的内容的目标文件。连接是将各个便一单元的目标文件和运行库当中被调用过的单元加以合并，生成可执行文件。

1. 会报错。编译时报错。
2. 会报错。连接时报错。
3. 不会报错
4. 会报错。连接时报错。

6-6

\* 在声明语句中，表示声明的是指针 如 int \*p;

在执行语句中或声明语句的初始化表达式中,表示访问指针所指对象的内容 如cout<<\*p;

& 在变量声明语句中，表示声明的是引用

在给变量赋初值时出现在等号右边或在执行语句中作为一元运算符出现时，表示取对象的地址，如 int \*pa = &b;

6-7

指针：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一个对象的地址，这个地址中的值就是另一个对象。

6-9

‘\0’字符

6-11

引用一旦被初始化，就不能改为指向其他对象

指针是变量，可以被赋值为另外的地址。

在需要对变量进行多次赋值时只能使用指针，而不使用引用。

6-15

const int \*p1 是常量指针，表示指针p1指向的是常量，不能通过指针改变指向的值

Int \* const p2 是指针常量，表示指针是常量，它一旦被初始化，就不能再指向其他的地址

6-18

指针p没有用delete删除?

#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

声明参数为int, 返回值为long的函数指针： long (\*func)(int i);

声明类A的一个成员函数指针，其参数为int,返回值为long： long(A::\*func)(int i);