5-1

答：作用域是一个标识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（块作用域）、类作用域和命名空间作用域。

5-2

答：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。作用域可见性的一般规则如下：①标识符要声明在前，引用在后；②在同一作用域中，不能声明同名的标识符；③在没有互相包含关系的不同作用域中声名的同名标识符，互不影响；④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，则外层标识符内层不可见。

5-5

答：静态数据成员是类中具有静态生存期的数据成员，一般用法是“类名：：标识符”。在类的定义中仅仅对静态数据成员进行引用性声明，必须在命名空间作用域的某个地方使用类名限定定义性说明，也可以初始化，即在类定义之外再加以定义。初始化引用的方式也值得注意，首先要利用类名来引用，其次可以直接初始化，除此之外不得直接访问。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据，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。

5-6

答：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好处，一是由于静态成员函数只能直接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的负面影响；二是同一类只维护一个静态函数成员拷贝，节约了系统的开销，提高程序的运行效率。

5-9

答：因为友元关系是不能传递的，是单向的，也是不能被继承的，所以类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

答：可以私有，例如：private:

static int I;

5-16

答：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，输出时可执行文件。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。连接是将各个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没有报错；

④连接错误，函数重复定义。

6-6

答：\*是指针运算符，是一元操作符，表示指针所指向对象的值；&是取地址运算符，也是一元操作符，是用来得到一个对象的地址。

6-7

答：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

答：NULL字符。

6-11

答：引用是一个别名，不能被重新分配，不可以是NULL值；指针是一个存放地址的变量。当需要对变量重新赋予另外的地址或赋值为NULL值时只能使用指针。

6-15

答：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

答：#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

答：long(\*p\_ fn1)(int);

long(A::\*p\_ fn2)(int);