第六章

1. 函数的调用

c++中函数的调用通过操作符()调用,一般作用于,函数名字或函数操作符

1. 局部静态对象

由static 变量定义的变量,在程序第一次执行路过时,进行初始化,,并且以后都不在初始化,直到程序终结.如果静态对象,没有进行显示的初始化,内置类型被初始化为0,如果是类类型则调用默认构造函数.

1. 参数的传递

参数的传递一般有三种方式,传递的如果是引用类型,或者指针类型,则是与传递的实参进行绑定,否则进行拷贝,我们认为在大多数情况下,引用与指针的效率,要高于拷贝,另外,对于引用和指针,我们更倾向于引用,另外,在你不想改变的参数,前面加上const,这样对于使用者,就不需要考虑,他是传值,还是传引用.

1. 可变参数的形参函数

这里接受initializer\_list(),关于可变参数模板将在后面的笔记中介绍,initializer\_list定义与<initializer\_list>头文件中,使用时需要保证,参数的类型都是一直的,同时放在{}中,例如标准库中的max函数

新版本可以这样使用:max({1,2,3,4,5,64,56});

1. 返回函数指针

见

1. constexper函数

在从c11中只允许函数体内只能有一条return 语句但是在c14中允许有更多的语句,如递归等.

1. 函数指针

函数指针指向的是函数而非对象,和其他指针一样,函数指针指向某种特定的类型.函数的类型由他的返回类型和参数类型共同决定,与函数名无关.

eg:

bool cmp (const string&,const string&)

它的类型是bool(const string&,const string&)而不是,cmp;

声明一个函数指针;

bool (\*pr) (const string&,const string&);

使用:

pr=cmp;

pr=&cmp

两者等价

bool l=pr(“dsa”,”aaf”);

bool l1=(\*pr)(“fsa”,”fasf”);

bool l2=cmp(“asd”,”asdadf”);

三种调用等价;

1. 函数指针形参

声明:

void a (int,int

bool cmp (const string&,const string &));

void a (int,int,

bool (\*cmp) (const string&,const string&));

这两种声明等价;

可以直接将函数名作为参数传给函数,编译器会自动将函数名转换为函数指针.

也可以定义别名:

typedef bool f (const string&,const string &);

typedef bool (\*f1) (const string &,const string&);

更多细节见: 