# void func1( MYCLASS \*&pBuildingElement );

# 1.指针与引用

指针传递只是传了一个地址的复制，在函数内部改变形参所指向的地址，不能改变原实参指向的地址，仅可以通过修改形参地址的内容，来达到修改实参内容的目的，所以如果想通过被调函数来修改原实参的地址或给重新分配一个对象都是不能完成的，只能使用双指针或指针引用。

★★相同点：

●都是地址的概念：

指针指向一块内存，它的内容是所指内存的地址；而引用则是某块内存的别名。

★★不同点：

指针指向一块内存，它的内容是所指内存的地址；而引用则是某块内存的别名，引用不改变指向

●指针是一个实体，而引用仅是个别名；

●引用只能在定义时被初始化一次，之后不可变；指针可变；引用“从一而终”，指针可以“见异思迁”；

●引用没有const，指针有const，const的指针不可变；（具体指没有int& const a这种形式，指引用本身即别名不可以改变，这是当然的，所以不需要这种形式；而const int& a是有的，是指引用所指的值不可以改变）

●引用不能为空，指针可以为空；

●“sizeof 引用”得到的是所指向的变量(对象)的大小，而“sizeof 指针”得到的是指针本身的大小；

●指针和引用的自增(++)运算意义不一样：

指针是对内存地址的自增，引用是对值的自增

●引用是类型安全的，而指针不是（引用比指针多了类型检查）

# 指针的指针和指针的引用

先来看“int \*\*pp”和“int \*&rp”区别。前者是一个指向指针的指针；后者是一个指针的引用。如果这样看不明白的话，变换一下就清楚了：

typedef int \* LPINT;

LPINT \*pp;

LPINT &rp;

而指针的指针和指针的引用作为传递参数时，如下面的两个函数在被调用时，编译器编译的二进制代码都将传递一个双重指针，只不过两者的调用方法不同：

*[cpp] view plaincopy*

*void function1(int \*\*p)*

*{*

*\*\*p=100;*

*\*p=NULL;*

*}*

*void function2(int \*&ref)*

*{*

*\*ref=100;*

*ref=NULL;*

*}*

可见，“引用”仅仅是为了给重载操作符提供了方便之门，其本质和指针是没有区别的。所以只要你碰到\*&，就应该想到\*\*。也就是说这个函数修改或可能修改调用者的指针，而调用者象普通变量一样传递这个指针，不使用地址操作符&。

# const关键字

常量指针VS常量引用

常量指针：**指向常量**的指针。

Const char \*name = “chen”; // 声明指向常量的指针

// 声明一个名为name的指针变量，它指向一个字符型常量。

即不允许改变指针所指地之中的常量；但允许改变name所指的地址

Name[3] = ‘s’; // 错误 name = “zhang” // 正确

常量引用：**指向常量**的引用，在引用定义语句的类型前加const，表示指向的对象是常量。也跟指针一样不能利用引用对指向的变量进行重新赋值操作。

比如，常量引用定义"const int &ref=a"告诉编译器，ref是常量，不能将ref作为左值进行操作。

指针常量VS引用常量

指针常量：在指针定义语句的指针名前加const，表示指针本身是常量。在定义指针常量时必须初始化！

即，指针所指的地址为常量，不可更改；而指针指向的内容可以更改。

Char \* const name = “chen”; // 声明一个指针常量

// 声明一个名为name的指针变量，该指针是不能移动的固定指针。

name = “zhang” // 错误 Name[3] = ’s’ // 正确

而引用的不可更改是天生具来的属性，不用再引用指针定义语句的引用名前加const。

常量指针常量VS常量引用常量

常量指针常量：指向常量的指针常量，可以定义一个指向常量的指针常量，它必须在定义时初始化。常量指针常量定义"const int\* const pointer=&c"告诉编译器，pointer和\*pointer都是常量，他们都不能作为左值进行操作。

Const char \* const name = “chen”; // 声明一个指向常量的指针常量

// 声明一个名为name的指针变量，该指针是不能移动的固定指针且指向一个字符型常量。

即，不能改变指针所指的地址；也不能改变指针所指地址中的数据

name = “zhang” // 正确 Name[3] = ’s’ // 正确

不存在所谓的"常量引用常量"，因为跟上面讲的一样引用变量就是引用常量。C++不区分变量的const引用和const变量的引用。程序决不能给引用本身重新赋值，使他指向另一个变量，因此引用总是const的。如果对引用应用关键字const，起作用就是使其目标称为const变量。即没有：Const double const& a=1；只有const double& a=1；