参数的传递方式-2

杨振平

参数的传递方式（续）

2.引用传递

（1）引用

引用是一种特殊的变量，它被认为是一个变量的别名。

引用定义的格式如下：

<数据类型> &<引用名>=<目标变量名>;

其中：&为引用（变量）的标志符号，<引用名>是一个标识符。

<数据类型>为<目标变量>的类型。

例如：int a,&b=a;

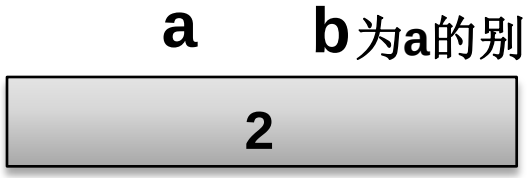
该例说明了a是一个整型变量，b是一个引用整型变量a的引用，即b是a变量的一个别名。这时，使用a与使用b是等价的。

参数的传递方式（续）

对引用的说明：

- 定义一个引用，其实是为目标变量起一个别名。引用并不分配独立的内存空间，它与目标变量共用其内存空间。
- 定义一个引用（变量）时，如果该引用不是用作函数的参数或返回值，则必须提供该引用的初值（即必须提供引用的目标变量名）。
- 使用引用与使用目标变量效果是相同的。

```
int main()
{
    int a=2,&b=a;
    cout<<&a<<“ ”<<&b<<endl;//输出变量的地
    cout<<a<<“ ”<<b<<endl; //输出变量的值
    return 0;
}
```




```
0x23fe34 0x23fe34
2 2
```

参数的传递方式（续）

（2）引用传递

为实现引用传递，这时函数的形参应定义为引用类型变量，而对应的实参应为变量名，该变量将作为引用的目标变量名。

函数调用时，作为形参的引用变量并不分配新的内存空间，它将作为实参变量的别名与其共用内存。

- 使用引用参数可以直接操作实参变量，从而能够实现通过修改形参的值而达到修改对应实参值的目的。
- 通过设置多个引用参数，可以从函数中带回多个结果值。

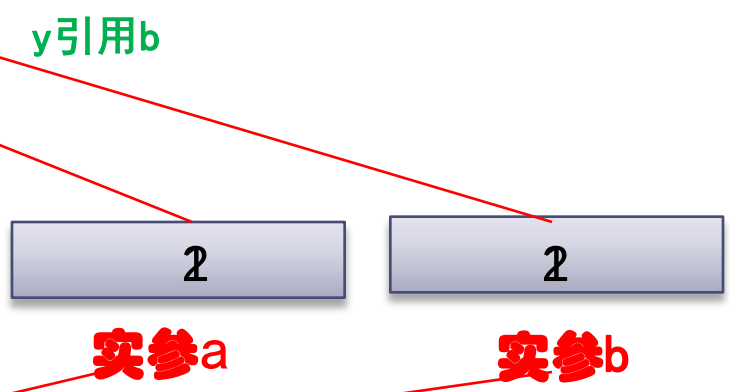
说明：引用作为函数形参，其引用的目标变量默认为调用该函数时对应的实参变量名，所以，在定义函数时，对于引用类型参数不必提供引用的初值。

例：使用引用传递在被调函数中改变实参的值

x,y为引用参数

```
void swap(int &x, int &y)
{   int  tmp;
    tmp = x;   x = y;  y = tmp;
}
int main( )
{   int a = 1, b = 2;
    cout << "Before exchange：a= " << a << ",b= " << b << endl;
    swap(a, b);  // 独立语句调用
    cout << "After  exchange：a= " << a << ",b= " << b << endl;
    return 0;
}
```

交换x,y就是交换a,b




```
Before exchange：a= 1,b= 2
After  exchange：a= 2,b= 1
```