Python工程师	Q	≥ 消息	-
) · · · — · — · · · · ·		_	

原 C与C++关于*与&(引用方式传参,传值方式传参)

2018年08月13日 23:29:43 夏普通 阅读数:6164

©CSDD 版权声明:本文为博主原创文章 , 未经博主允许不得转载。 https://blog.csdn.net/qq_34243930/article/details/81638852

一个例子

```
#include <cstdio>
using namespace std;

void swap(){

int main()
{
  int a=5,b=10;
  swap(a,b);
  return 0;
}
```

补全该代码:

用C++:

```
void swap( int &x,int &y){
   int temp;
   temp=x;
   x=y;
   y=temp;
}
```

用C:是把地址传进去,然后用*运算取地址的内容

```
#include <cstdio>
using namespace std;

void swap(){

int main()
{
  int a=5,b=10;
  swap(a,b);
  return 0;
}
```

1、引用方式传参,引用相当于是外部变量的别名,实际操作的就是该变量,即在函数内对该变量进行修改的话,在外部该变量也会相应被修改。

void ChangeRef(int& ref);

2、传值方式传参,实际上是复制了一个外部变量的副本进入函数中,在函数内操作的为该副本,对该变量的修改并不会实际修改函数外的该变量。

void ChangeNotRef(int ref);

3、注意,此处未考虑传入的参数为指针的情况,<mark>如果传值方式传入的是指针</mark>,那么如果单纯修改指针,那么在跳出函数后,该<mark>指针指向</mark>不受影响,但是如果修改的是指针指向的值,那么外部指针所指向的内容也是相应修改了的。因为以指针为参数进行传值,也是在函数内部复制了一个指针的副本,因而在内部修改指针本身即修改其副本并不影响外部的指针,但是两个指针都是指向同一个内存空间的,因而如果在函数内部修改了指针指向的内容,外部指针指向的内容也就相应改变了。

//所以! 引用方式传参 &p里修改p 等价于 传值方式(传入的是指针)传参*p里修改指针指向的值 *p

```
#include <iostream>
   using namespace std;
3
   //测试如果引用进入函数后进行修改,跳出函数后变量是否修改;
   void ChangeRef(int& ref){
      ref++;
6
7
      cout << "引用内修改后的值为:" << ref << endl;
8
9
   //测试以传值方式进入函数进行修改,跳出函数后变量是否修改
   void ChangeNotRef(int ref){
11
12
      cout << "通过传值方式在函数内修改后的值为:" << ref << endl;
13
14 }
```

```
15
    16 //测试以指针为参数传入函数,修改指针指向,是否影响外部指针指向
17 | void ChangePtr(int* ptr){
18
      ptr++;
       cout << "函数内修改后的指针指向: " << ptr << endl;
19
20
21
   //测试在函数中修改指针指向的内容,是否影响外部指针指向内容
22
23 | void ChangePtrVal(int* ptr){
24
       (*ptr)++;
       cout << "函数内修改后指针指向的值: " << *ptr << endl;
25
26
27
28
   int main(){
      int Ref = 0;
29
30
      cout << "传入引用之前的值为:" << Ref << endl;
31
      ChangeRef(Ref);
      cout << "在函数内修改引用后, 跳出函数的值为:" << Ref << endl<<endl;
32
33
34
       ChangeNotRef(Ref);
       cout << "通过传值方式在函数内修改后, 跳出函数的值为:" << Ref << endl<<endl;
35
36
37
       int* ptr = &Ref;
38
       cout << "传入函数前的指针指向: " << ptr << endl;
39
      ChangePtr(ptr);
40
      cout << "传值方式传入函数跳出后的指针指向: " << ptr << endl<<endl;
41
42
       cout << "传入前指针指向的值为: " << *ptr << endl;
43
      ChangePtrVal(ptr);
       cout << "从函数中出来后指针指向的值为: " << *ptr << endl<<endl;
44
45
46
47
```



其中:

int* ptr = &Ref;

*ptr是指向这个地址(&Ref)的值

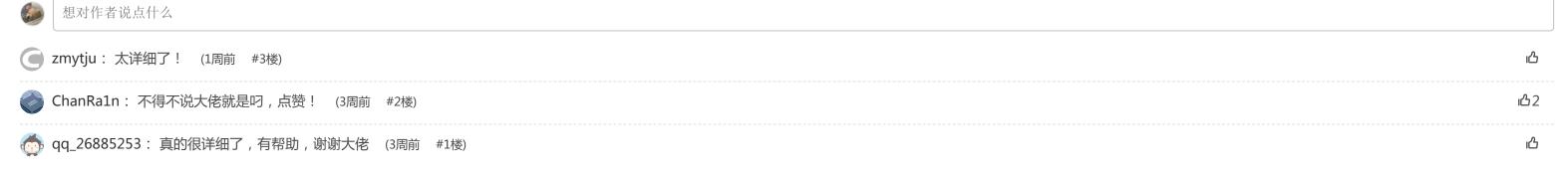
补充:

ptr=&Ref

int x=1;

int *y=&x; //用于指针传递,y有自己独立的内存地址,存储的内容是x的地址,*y是x的值int &z=x; //用于引用传递,可以理解为z就是x,x就是z,只不过名字不一样

还不理解看一下: https://www.cnblogs.com/codingmengmeng/p/5865510.html



上一页 1 下一页