C++函数返回引用

转载 非长道 于 2016-12-08 13:31:23 发布 37713 收藏 84
分类专栏： C/C++ 文章标签： 引用 C++ 返回值 函数

注：C++ 有三种传递方式：值传递，指针传递，引用传递

返回**“值”**和返回**“引用”**是不同的

函数返回**值**时会产生一个临时变量作为函数返回值的副本，而返回**引用**时不会产生值的副本，既然是引用，那引用谁呢？这个问题必须清楚，否则将无法理解返回引用到底是个什么概念。以下是几种引用情况：

一、千万不要返回局部对象的引用

```
const string &mainip(const string &s)
{
      string ret=s;
      return ret;
}
```

当函数执行完毕，程序将释放分配给局部对象的存储空间。此时，对局部对象的引用就会指向不确定的内存。

同理，指针也是这样，返回指针的时候，不能指向局部临时变量，否则指针将变为野指针；

二、引用函数的参数，当然该参数也是一个引用

```
const string &shorterString(const string &s1,const string &s2)
{
      return s1.size()<s2.size()?s1:s2;
}
```

以上函数的返回值是引用类型。无论返回s1或是s2,调用函数和返回结果时，都没有拷贝这些string对象。简单的说，返回的引用是函数的参数s1或s2，并且参数s1、s2也是引用，不是在函数体内产生的。函数体内局部对象是不能被引用的，因为函数调用完局部对象会被释放。

三、返回this指向的对象

在类的成员函数中，返回引用的类对象，当然不能是函数内定义的类对象（会释放掉），一般为 this 指向的对象，典型的例子是 string类的赋值函数。

```
[cpp]
String& String::operator =(const String &str)   //注意与“+”比较，函数为什么要用引用呢？a=b=c，可以做为左值
{
    if (this == &str)
    {
        return *this;
    }
    delete [] m_string;
    int len = strlen(str.m_string);
    m_string = new char[len+1];
    strcpy(m_string,str.m_string);
    return *this;
}
```

四、引用返回this 的成员变量，或者 引用参数的成员变量

原标题为：引用返回左值（上例的=赋值也是如此，即a=b=c是可以的）

原文这里表达不清晰，因为只要是引用，都可以作为左值使用。只因为下面的例子一般用在等号左边，当左值使用。

可以定义一个和返回值一样的引用类型，来接受函数的返回值，操作此引用值，和直接操作函数的参数是一样的.引用都是使用引用传递：

```
char &get_val(string &str,string::size_type ix)
{
      return str[ix];
}
使用语句调用:
 string s("123456");
 cout<<s<<endl;
 get_val(s,0)='a';
 cout<<s<<endl;
```

这种情况，和第二种是一样的，只不过是返回了参数（引用类型）的一部分。也可以不作为左值，故修改如下:

```
char &ch = get_val(s,0);
ch = ‘A’;
```

此句进行的都是引用传递，故运行之后，s[0] 就变为了 A，s为“A23456”；

此外，可以返回引用参数的成员变量，亲测有效。似乎不是局部临时变量，只要函数结束之后内存没有被销毁的，作为引用返回都没问题：

```
QString& Test(Student &stu)
{
    return stu.m_name;
}
```

```
QString & Student::getRName()
{
    return (*this).m_name;
}
```

五、最后转上一段code作为总结

```
#include<iostream>
using namespace std;
string make_plural(size_t,const string&,const string&);
const string &shorterString(const string &,const string &);
const string &mainip(const string&);
char &get_val(string &,string::size_type);
int main(void)
{
    cout<<make_plural(1,"dog","s")<<endl;
    cout<<make_plural(2,"dog","s")<<endl;

    string string1="1234";
    string string2="abc";
    cout<<shorterString(string1,string2)<<endl;

    cout<<mainip("jiajia")<<endl;


    string s("123456");
    cout<<s<<endl;
    get_val(s,0)='a';

    cout<<s<<endl;

    getchar();
    return 0;
}
//返回非引用
string make_plural(size_t i,const string &word,const string &ending)
{
    return (i==1)?word:word+ending;
}
//返回引用
const string &shorterString(const string &s1,const string &s2)
{
    return s1.size()<s2.size()?s1:s2;
}
//禁止返回局部对象的引用（我的dev c++ 没有报错，比较可怕）
const string &mainip(const string &s)
{
    string ret=s;
    return ret;
}
//引用返回左值
char &get_val(string &str,string::size_type ix)
{
    return str[ix];
}
```