**[C++ map应用之insert方法](http://blog.csdn.net/jingjingtr/article/details/7424172)**

直接上代码，两种方式：

map<string,int> m\_map;

1. m\_map.insert(map<string,int>::value\_type("hello",5)）;
2. m\_map.insert(make\_pair("hello",5)）;

也就是说,insert后面的数据是pair类型或者是value\_type类型了，然而对C++有了解的人都明白，其实value\_type和pair<const k,v>是等价的、insert()中的参数必须是value\_type类型，那么为什么insert()中的参数能够使用make\_pair产生的pair<k,v>呢？

其实，因为我们在加入pair<k,v>时的k已经是常量了，所以可以加入。。。而正常来讲这都是所有编译器所能接受的。

在insert插入的同时，还有返回值来说明是否插入成功，就是pair< map<string,int>::iterator,bool> >类型，如本实例pair< map<string,int>::iterator,bool> > rent= m\_map.insert(make\_pair("hello",5)）;

rent->second即是成功与否的标志；rent->first就是返回的map<string,int>::iterator迭代器；rent->first.first就是string类型的数据。

这些看起来都非常的麻烦，但是只要信心的研究、编译、调试就可以了。

例如

map<string, string> parameters;//存储所有的变量和其对应的值

parameters.insert(pair<string, string>(var\_name, value));

即可插入

Strncpy(dest, src, size)

Strcmp

Strcpy

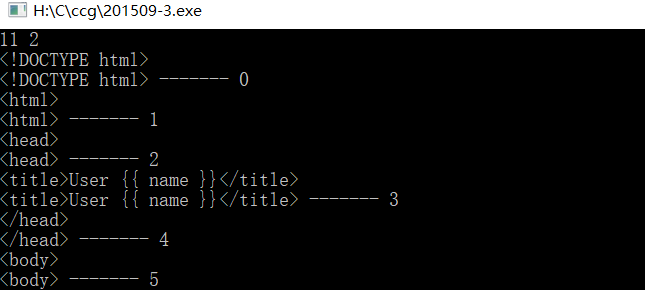
Scanf可以通过正则表达式限定输入结束符号

scanf("%[^\n]", &moban[i]);

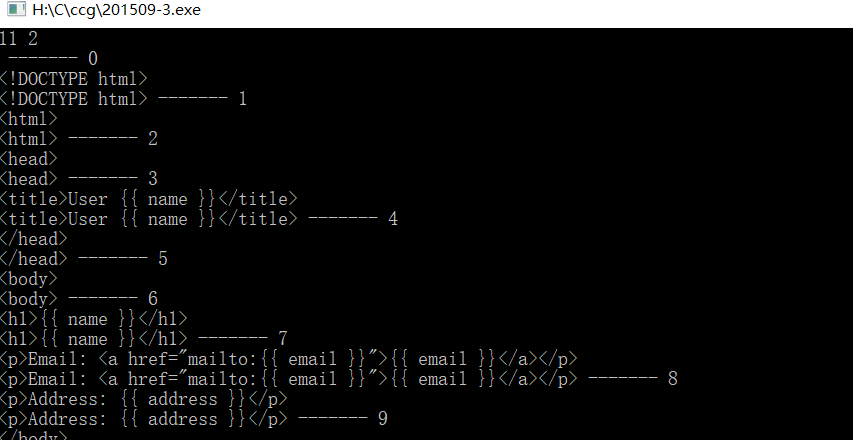
而printf不可以。。直接输出即可

Printf(“%s”, moban[i]);

在用到scanf的正则表达式时，最好清空缓存区，否则会导致之前没用正则的部分也加入到里面，如：



（正确）



（错误， 11和2也被当作输入的字符串 实际上应该为上一个scanf(“%d %d”, &m, &n）

# [C++之substr与substring浅谈](http://blog.csdn.net/qq_18815817/article/details/70239460)

**转自：http://blog.sina.com.cn/s/blog\_9d85c1900102v1rm.html**

###### ****区别：****主要是两者的参数不同

###### 功能：相似。

###### 

###### substr ：返回一个从指定位置开始的指定长度的子字符串

###### ****substring ：****返回位于 String 对象中指定位置的子字符串。

**用法：**

***stringvar*.substr(*start* , [*length* ])**

**参数**

stringvar   必选项。要提取子字符串的字符串文字或 **String** 对象。

      start         必选项。所需的子字符串的起始位置。字符串中的第一个字符的索引为 0。

      length      可选项。在返回的子字符串中应包括的字符个数。

**说明**如果 *length* 为 0 或负数，将返回一个空字符串。如果没有指定该参数，则子字符串将延续到 *stringvar* 的最后。

*strVariable*.**substring(***start*, *end***)**

**参数**

start：指明子字符串的起始位置，该索引从 0 开始起算。

end：指明子字符串的结束位置，该索引从 0 开始起算。

**说明**

**substring** 方法将返回一个包含从 *start* 到最后（不包含 *end* ）的子字符串的字符串。

**substring** 方法使用 *start* 和 *end* 两者中的较小值作为子字符串的起始点。

例如， *strvar*.**substring(**0, 3**)** 和*strvar*.**substring(**3, 0**)** 将返回相同的子字符串。

如果 *start* 或 *end* 为 **NaN** 或者负数，那么将其替换为0。

子字符串的长度等于 *start* 和 *end* 之差的绝对值。

例如，在 *strvar*.**substring(**0, 3**)** 和 *strvar*.**substring(**3, 0**)**返回的子字符串的的长度是 3。

**substr 方法**

**示例**

下面的示例演示了**substr** 方法的用法。

function SubstrDemo(){

   var s, ss;                //声明变量。

   var s = "The rain in Spain falls mainly in the plain.";

   ss = **s.substr(**12**,** 5**)**;  //获取子字符串。

   return(ss);               //返回 "Spain"。

练习题目

在表单(form1)中有一个文本框元素(fname)，用于输入电话号码，格式如：010-82668155，

要求前3位是010，紧接一个“-”，后面是8位数字。要求在提交表单时，根据上述条件验证该文本框中输入内容的有效性，下列语句中，（ ）能正确实现以上功能

     A. varstr=form1.fname.value;

         if(str.substr(0,4)!="010-"|| str.substr(4).length!=8||

               isNaN(parseFloat(str.substr(4))))

             alert("无效的电话号码！");

     B. varstr=form1.fname.value;

         if(str.substr(0,4)!="010-" && str.substr(4).length!=8 &&

               isNaN(parseFloat(str.substr(4))))

             alert("无效的电话号码！");

     C. varstr=form1.fname.value;

         if(str.substr(0,3)!="010-"|| str.substr(3).length!=8||

               isNaN(parseFloat(str.substr(3))))

             alert("无效的电话号码！");

     D. varstr=form1.fname.value;

         if(str.substr(0,4)!="010-"&&str.substr(4).length!=8 &&

               !isNaN(parseFloat(str.substr(4))))

             alert("无效的电话号码！");

**substring 方法**

**示例**

下面的示例演示了 **substring** 方法的用法。

function SubstringDemo(){

   var ss;                         //声明变量。

   var s = "The rain in Spain falls mainly in the plain..";

   ss = **s.substring(**12**,** 17**)**;   //取子字符串。

   return(ss);                     //返回子字符串。

}