原 简单的程序诠释C++ STL算法系列之十四：copy_backward

前文中展示了copy的魅力，现在我们来看一下它的孪生兄弟copy_backward，copy_backward算法与copy在行为方面相似，只不过它的复制过程与copy背道而驰，其复制过程是从最后的元素开始复制，直到首元素复制出来。也就是说，复制操作是从last-1开始，直到first结束。这些元素也被从后向前复制到目标容器中，从result-1开始，一直复制last-first个元素。举个简单的例子：已知vector {0, 1, 2, 3, 4, 5}，现我们需要把最后三个元素（3, 4, 5）复制到前面三个（0, 1, 2）位置中，那我们可以这样设置：将first设置值3的位置，将last设置为5的下一个位置，而result设置为3的位置，这样，就会先将值5复制到2的位置，然后4复制到1的位置，最后3复制到0的位置，得到我们所要的序列{3, 4, 5, 3, 4, 5}。下面我们来看一下copy_backward的函数原型：

函数原型：

```
template<class BidirectionalIterator1, class BidirectionalIterator2>
 BidirectionalIterator2 copy_backward ( BidirectionalIterator1 first,
                                        BidirectionalIterator1 last,
                                        BidirectionalIterator2 result);
```

参数：

first, last
指出被复制的元素的区间范围[first，last).
result
指出复制到目标区间的具体位置[result-(last-first),result)

返回值：

返回一个迭代器，指出已被复制元素区间的起始位置

程序示例：

先通过一个简单的示例来阐述copy_backward的使用方法，程序比较简单，代码中做了详细的说明，在此不再累赘。

```
/*****************************************************************
 * Copyright (C) Jerry Jiang
 *
 * File Name   : copy_backward.cpp
 * Author      : Jerry Jiang
 * Create Time : 2012-3-21 23:14:57
 * Mail        : jbiaojerry@gmail.com
 * Blog        : http://blog.csdn.net/jerryjbiao
 *
 * Description :  简单的程序诠释C++ STL算法系列之十四
 *                变易算法 : 反向复制copy_backward
 *
 *****************************************************************/

#include <iostream>
#include <algorithm>
#include <vector>

using namespace std;

int main()
{
 vector<int> myvector;
 vector<int>::iterator iter;

 //为容器myvector赋初始值:10 20 30 40 50
 for ( int i = 1; i <= 5; ++i )
 {
  myvector.push_back( i*10 );
 }

 //将myvector容器的大小增加3个单元
 myvector.resize( myvector.size()+3 );

 //将容器元素20、10拷贝到第八、第七个单元中：10 20 30 40 50 0 10 20
 //注意copy_backward是反向复制，先将20拷贝到第八个单元，再将10拷贝到第七个单元
 copy_backward( myvector.begin(), myvector.begin()+2, myvector.end() );

 for ( iter = myvector.begin(); iter != myvector.end(); ++iter )
 {
  cout << " " << *iter;
 }

 cout << endl;


 //清除myvector容器
 myvector.clear();

 //还原容器myvector的初始值:10 20 30 40 50
 for ( i = 1; i <= 5; ++i )
 {
  myvector.push_back( i*10 );
 }

 //将容器元素40、50覆盖10、20，即：40 50 30 40 50：
 copy_backward( myvector.end()-2, myvector.end(), myvector.end()-3 );

 for ( iter = myvector.begin(); iter != myvector.end(); ++iter )
 {
  cout << " " << *iter;
 }

 cout << endl;
 return 0;
}
```

通过上例的简单介绍相信大家对copy_backward 的基本使用不再陌生了吧，^_^，下面我们结合前面所讲的for_search算法来巩固一下copy_backward的使用。

```
/*****************************************************************
 * Copyright (C) Jerry Jiang
 *
 * File Name   : copy_backward02.cpp
 * Author      : Jerry Jiang
 * Create Time : 2012-3-21 23:48:14
 * Mail        : jbiaojerry@gmail.com
 * Blog        : http://blog.csdn.net/jerryjbiao
 *
 * Description :  简单的程序诠释C++ STL算法系列之十四
 *                变易算法 : 反向复制copy_backward
 *
 *****************************************************************/

#include <iostream>
#include <algorithm>
#include <vector>
#include <iterator>
#include <string>

using namespace std;

class output_element
{
public:
 //重载运算符()
 void operator() (string element)
 {
  cout << element
    << ( _line_cnt++ % 7 ? " " : "\n\t"); //格式化输出,即每7个换行和制表位
 }

 static void reset_line_cnt()
 {
  _line_cnt = 1;
 }

private:
 static int _line_cnt;

};

int output_element::_line_cnt = 1; //定义并初始静态数据成员

int main()
{
 string sa[] = {
  "The", "light", "untonusred", "hair",
  "grained", "and", "hued", "like", "pale", "oak"
 };

 vector<string> svec(sa, sa+10);

 //还记得for_each吧，呵呵，这里用它来作为输出
 //for_each具体用法参考 http://blog.csdn.net/jerryjbiao/article/details/6827508
 cout << "Original list of strings:\n\t";
 for_each( svec.begin(), svec.end(), output_element() );
 cout << "\n" << endl;

 //将"The", "light", "untonusred", "hair","grained",
 //"and", "hued"后移三个单元覆盖了"like", "pale", "oak"
 copy_backward(svec.begin(), svec.end()-3, svec.end());

 output_element::reset_line_cnt();

 cout << "sequence after "
   << "copy_backward(svec.begin(), svec.end()-3, svec.end()): \n\t";
 for_each( svec.begin(), svec.end(), output_element() );
 cout << "\n" << endl;

 return 0;
}
```

****

C++经典书目索引及资源下载：http://blog.csdn.net/jerryjbiao/article/details/7358796

****

阅读更多　　登录后自动展开