

原 STL的Deque介绍

2013年09月24日 12:23:25 阅读数：2093

Deque是双端队列，在画动态图表的时候使用起来比较方便。因为当数据填满图表后，需要从队列的后方插入一个元素，然后再从队列的前方删除一个元素。使用Deque可以很方便的做到。使用push(pop)-back(front)就ok了

下面摘录了一个Deque的使用教程

```
[cpp]    
1.  /*deque: 是一个double-ended queue,  
2.     1)支持随即存取, 也就是[]操作符,  
3.     2)支持两端操作, push(pop)-back(front), 在两端操作上与list效率差不多  
4.  
5.     因此在实际使用时, 如何选择这三个容器中哪一个, 应根据你的需要而定, 一般应遵循下面的原则:  
6.     1、如果你需要高效的随即存取, 而不在乎插入和删除的效率, 使用vector  
7.     2、如果你需要大量的插入和删除, 而不关心随即存取, 则应使用list  
8.     3、如果你需要随即存取, 而且关心两端数据的插入和删除, 则应使用deque。  
9. */  
10.  
11. #include <iostream>  
12. #include <deque>  
13. using namespace std;  
14.  
15. void printDeque(deque<int> d)  
16. {  
17.     //使用下标  
18.     //for (unsigned int i = 0; i < d.size(); i++)  
19.     //{  
20.     //  cout<<"d["<<i<<" = "<<d[i]<<" ,  
21.     //}  
22.  
23.     //使用迭代器  
24.     //deque<int>::iterator iter = d.begin();  
25.     //for (;iter != d.end(); iter ++)  
26.     //{  
27.     //  cout<<"d["<<iter-d.begin()<<" = "<<(*iter)<<" ,  
28.     //}  
29.  
30.     //使用迭代器指针  
31.     deque<int>::iterator *pIter = new deque<int>::iterator;  
32.     if ( NULL == pIter )  
33.     {  
34.         return ;  
35.     }  
36.     for (*pIter = d.begin(); *pIter != d.end(); (*pIter)++)  
37.     {  
38.         cout<<"d["<<*pIter - d.begin() <<"]="<<*pIter<<" ,  
39.     }  
40.     if (NULL != pIter)  
41.     {  
42.         delete pIter;  
43.         pIter = NULL;  
44.     }  
45.  
46.     cout<<endl;  
47. }  
48.  
49. void main()  
50. {  
51.     //创建deque  
52.     deque<int> d1; //创建一个没有任何元素的deque对象  
53.     deque<int> d2(10); //创建一个具有10个元素的deque对象, 每个元素值为默认  
54.     deque<double> d3(10, 5.5); //伊妹一个具有10个元素的deque对象, 每个元素的初始值为5.5  
55.     deque<double> d4(d3); //通过拷贝一个deque对象的元素值, 创建一个新的deque对象  
56.     int iArray[] = {11, 13, 19, 23, 27};  
57.     deque<int> d5(iArray, iArray+5); //将迭代器区间[first, last)所指的元素拷贝到一个新创建的deque对象中  
58.  
59.     //初始化赋值: 同vector一样, 使用尾部插入函数push_back()  
60.     for (int i = 1; i < 6 ; i++)  
61.         d1.push_back(i*10);  
62.     //遍历元素: 1-下标方式 2-迭代器方式 反向遍历 (略)  
63.     cout<<"printDeque(d1) : "<<endl;  
64.     printDeque(d1);  
65.  
66.     //元素插入: 尾部插入用push_back(), 头部插入用push_front(), 其它位置插入用insert(&pos, elem)  
67.     cout<<"d1.push_front(100): "<<endl;  
68.     d1.push_front(100);  
69.     printDeque(d1);  
70.     cout<<"d1.insert(d1.begin()+3, 200): "<<endl; //支持随机存取(即[]操作符), 所以begin()可以+3  
71.     d1.insert(d1.begin()+2,200);  
72.     printDeque(d1);  
73.  
74.     //元素删除 尾部删除用pop_back();头部删除用pop_front();  
75.     //任意迭代位置或迭代区间上的元素删除用erase(&pos)/erase(&first, &last); 删除所有元素用clear();  
76.     cout<<"d1.pop_front(): "<<endl;
```

```

77.     d1.pop_front();
78.     printDeque(d1);
79.
80.     cout<<"d1.erase(d1.begin()+1): "<<endl;
81.     d1.erase(d1.begin()+1); //删除第2个元素d1[1]
82.     printDeque(d1);
83.
84.     cout<<"d1.erase(d1.begin(), d1.begin() + 2) = "<<endl;
85.     d1.erase(d1.begin(), d1.begin() + 2);
86.     printDeque(d1);
87.
88.     cout<<"d1.clear() : "<<endl;
89.     d1.clear();
90.     printDeque(d1);
91.
92.
93.     //其它常用
94.     cout<<"其它常用用法: "<<endl;
95.     int flag = 0;
96.     while(flag < 2)
97.     {
98.         if (0 == flag )
99.         {
100.             for (int i = 1; i < 6 ; i++) //恢复
101.                 d1.push_back(i*10);
102.         }
103.         else
104.         {
105.             d1.clear();
106.             cout<<"after d1.clear() , d1.front(), d1.back() is abnormal! other info.:"<<endl;
107.         }
108.         cout<<"d1.empty() = "<<d1.empty()<<endl;
109.         cout<<"d1.size() = "<<d1.size()<<endl;
110.         cout<<"d1.max_size() = "<<hex<<d1.max_size()<<endl;
111.         if (!d1.empty())
112.         {
113.             cout<<"d1.front() = "<<d1.front()<<endl;
114.             cout<<"d1.back() = "<<d1.back()<<endl;
115.         }
116.
117.         flag++;
118.     }
119.
120.
121.     //交换
122.     cout<<"d1.swap(d5)= "<<endl;
123.     d1.swap(d5);
124.     cout<<"d1 = ";
125.     printDeque(d1);
126.     cout<<"d5 = ";
127.     printDeque(d5);
128.     //printDeque(d)
129.
130.
131. }

```

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/leixiaohua1020/article/details/11970867>

文章标签： stl deque

个人分类： [纯编程](#)

此PDF由spygg生成, 请尊重原作者版权!!!

我的邮箱: liushidc@163.com