list 和 deque

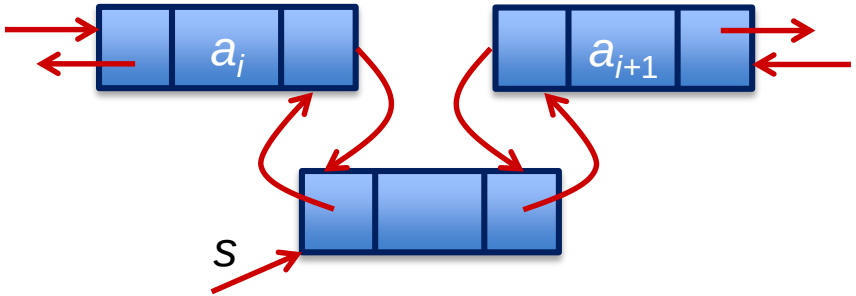
郭 炜　刘家瑛

北京大学

list 容器

- 双向链表 #include <list>

- 在任何位置插入/删除都是常数时间



- 不支持根据下标随机存取元素
- 具有所有顺序容器都有的成员函数

list 容器

- 还支持8个成员函数:

成员函数	作　　用
push_front	在链表最前面插入
pop_front	删除链表最前面的元素
sort	排序 (list 不支持 STL 的算法 sort)
remove	删除和指定值相等的所有元素
unique	删除所有和前一个元素相同的元素
merge	合并两个链表, 并清空被合并的链表
reverse	颠倒链表
splice	在指定位置前面插入另一链表中的一个或多个元素, 并在另一链表中删除被插入的元素

list容器之sort函数

- list容器的迭代器不支持完全随机访问

→不能用标准库中sort函数对它进行排序

- list自己的sort成员函数

```
list<T> classname
classname.sort(compare);    //compare函数可以自己定义
classname.sort();    //无参数版本，按<排序
```

- list容器只能使用双向迭代器

→不支持大于/小于比较运算符，[]运算符和随机移动

(即类似“list的迭代器+2”的操作)

```
#include <list>
#include <iostream>
#include <algorithm>
using namespace std;
class A {      //定义类A, 并以友元重载<, ==和<<
      private:
            int n;
      public:
            A( int n_ ) {    n = n_;    }
            friend bool operator<( const A & a1, const A & a2);
            friend bool operator==( const A & a1, const A & a2);
            friend ostream & operator <<(  ostream & o, const A & a);
};
```

```
bool operator<( const A & a1, const A & a2) {
    return a1.n < a2.n;
}
bool operator==( const A & a1, const A & a2) {
    return a1.n == a2.n;
}
ostream & operator <<(  ostream & o, const A & a) {
    o << a.n;
    return o;
}
```

```
//定义函数模板PrintList, 打印列表中的对象
template <class T>
void PrintList(const list<T> & lst) {
        int tmp = lst.size();
        if( tmp > 0 ) {
                typename list<T>::const_iterator i;
                i = lst.begin();
                for(i = lst.begin(); i != lst.end(); i ++)
                        cout << * i << ",";
        }       //与其他顺序容器不同, list容器只能使用双向迭代器,
}               //因此不支持大于/小于比较运算符, []运算符和随机移动
                // typename用来说明 list<T>::const_iterator是个类型
                //在VS中不写也可以
```

```
int main()   {
        list<A> lst1,lst2;
        lst1.push_back(1); lst1.push_back(3);
        lst1.push_back(2); lst1.push_back(4); lst1.push_back(2);
        lst2.push_back(10); lst2.push_front(20);
        lst2.push_back(30); lst2.push_back(30);
        lst2.push_back(30); lst2.push_front(40); lst2.push_back(40);
        cout << "1) "; PrintList( lst1); cout << endl;
        cout << "2) "; PrintList( lst2); cout << endl;
        lst2.sort();    //list容器的sort函数
        cout << "3) "; PrintList( lst2); cout << endl;
```

1) 1,3,2,4,2,

2) 40,20,10,30,30,30,40,

3) 10,20,30,30,30,40,40,

```
lst2.pop_front();
cout << "4) "; PrintList( lst2); cout << endl;
lst1.remove(2);    //删除所有和A(2)相等的元素
cout << "5) "; PrintList( lst1); cout << endl;
lst2.unique();    //删除所有和前一个元素相等的元素
cout << "6) "; PrintList( lst2); cout << endl;
lst1.merge (lst2);    //合并 lst2到lst1并清空lst2
cout << "7) "; PrintList( lst1); cout << endl;
cout << "8) "; PrintList( lst2); cout << endl;
lst1.reverse();
cout << "9) "; PrintList( lst1); cout << endl;
```

```
4) 20,30,30,30,40,40,
5) 1,3,4,
6) 20,30,40,
7) 1,3,4,20,30,40,
8)
9) 40,30,20,4,3,1,
```

```
    lst2.push_back (100); lst2.push_back (200);
    lst2.push_back (300); lst2.push_back (400);
    list<A>::iterator p1, p2, p3;
    p1 = find(lst1.begin(), lst1.end(), 3);
    p2 = find(lst2.begin(), lst2.end(), 200);
    p3 = find(lst2.begin(), lst2.end(), 400);
    lst1.splice(p1, lst2, p2, p3); //将[p2,p3)插入p1之前，
                                   //并从lst2中删除[p2, p3)
    cout << "11) "; PrintList( lst1); cout << endl;
    cout << "12) "; PrintList( lst2); cout << endl;
}
```

11) 40,30,20,4,200,300,3,1,

12) 100,400,

输出:

1) 1,3,2,4,2,

2) 40,20,10,30,30,30,40,

3) 10,20,30,30,30,40,40,

4) 20,30,30,30,40,40,

5) 1,3,4,

6) 20,30,40,

7) 1,3,4,20,30,40,

8)

9) 40,30,20,4,3,1,

11) 40,30,20,4,200,300,3,1,

12) 100,400,

deque 容器

- 双向队列
- 必须包含头文件 #include <deque>
- 所有适用于vector的操作 → 都适用于deque
- deque还有 push_front (将元素插入到容器的头部)

和 pop_front (删除头部的元素) 操作

In-Video Quiz

deque没有以下哪个成员函数

- A) sort
- B) push_front
- C) pop_front
- D) back