# STL之堆的使用

STL里面的堆操作一般用到的只有4个:make\_heap();、pop\_heap();、push\_heap();、sort\_heap();  
他们的头文件函数是#include <algorithm>

首先是make\_heap();  
他的函数原型是：void make\_heap(first\_pointer,end\_pointer,compare\_function);  
一个参数是数组或向量的头指针，第二个向量是尾指针。第三个参数是比较函数的名字。在缺省的时候，默认是大跟堆。（下面的参数都一样就不解释了）  
作用：把这一段的数组或向量做成一个堆的结构。范围是(first,last)  
  
然后是pop\_heap();  
  
它的函数原型是：void pop\_heap(first\_pointer,end\_pointer,compare\_function);  
  
作用：pop\_heap()不是真的把最大（最小）的元素从堆中弹出来。而是重新排序堆。它  
把first和last交换，然后将[first,last-1)的数据再做成一个堆。  
  
接着是push\_heap() void pushheap(first\_pointer,end\_pointer,compare\_function);  
  
作用：push\_heap()假设由[first,last-1)是一个有效的堆，然后，再把堆中的新元素加  
进来，做成一个堆。  
  
最后是sort\_heap()void sort\_heap(first\_pointer,end\_pointer,compare\_function);

作用是sort\_heap对[first,last)中的序列进行排序。它假设这个序列是有效堆。（当然  
，经过排序之后就不是一个有效堆了）

实例：

#include<algorithm>

#include<cstdio>

using namespace std;

bool cmp(int a,int b)

{

return a>b;

}

int main()

{

int i,number[20]={29,23,20,22,17,15,26,51,19,12,35,40};

make\_heap(&number[0],&number[12]);

*//结果是:51 35 40 23 29 20 26 22 19 12 17 15*

for(i=0;i<12;i++)

printf("%d ",number[i]);

printf("\n");

make\_heap(&number[0],&number[12],cmp);

*//结果：12 17 15 19 23 20 26 51 22 29 35 40*

for(i=0;i<12;i++)

printf("%d ",number[i]);

printf("\n");

*//加入元素8*

number[12]=8;

*//加入后调整*

push\_heap(&number[0],&number[13],cmp);

*//结果：8 17 12 19 23 15 26 51 22 35 40 20*

for(i=0;i<13;i++)

printf("%d ",number[i]);

printf("\n");

*//弹出元素8*

pop\_heap(&number[0],&number[13],cmp);

*//结果：12 17 15 19 23 20 26 51 22 29 35 40*

for(i=0;i<13;i++)

printf("%d ",number[i]);

printf("\n");

sort\_heap(&number[0],&number[12],cmp);

*//结果不用说都知道是有序的了！*

for(i=0;i<12;i++)

printf("%d ",number[i]);

return 0;

}