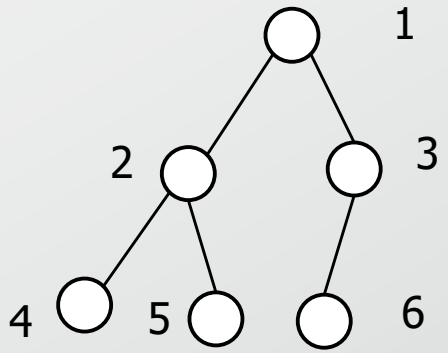
第五章 数的存储与组织

第59课 堆函数与优先队列

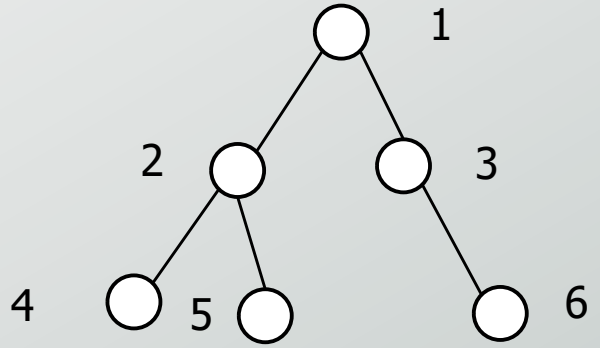
《信息学奥赛一本通·编程启蒙 C++版》

一、完全二叉树

完全二叉树的特点：叶子结点只能出现在最下层和次下层，且最下层的叶子结点集中在树的左部。如下图，左边是完全二叉树，右边不是。



完全二叉树



非完全二叉树

完全二叉树可以用数组很方便地表示：

1	2	3	4	5	6	

对于下标为 i 的节点，可以很容易计算出该节点的父节点、子节点的下标:

Parent(i) = floor(i/2)

LeftChild(i) = 2*i

RightChild(i) = 2*i + 1

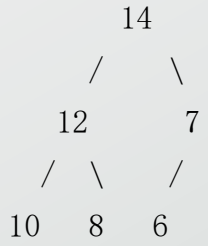
二、最大堆和最小堆

堆是一个完全二叉树(除了最底层以外，其他每一层都是满的)，因此堆可以用数组表示。

（1）大根堆

1、大根堆的最大值都出现在根节点

2、堆中每个父节点的值都大于子节点值

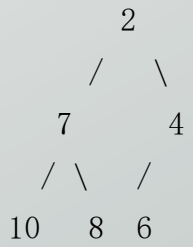


0	1	2	3	4	5	6
	14	12	7	10	8	6

（2）小根堆

1、小根堆的最小值都出现在根节点

2、堆中每个父节点的值都小于子节点值



0	1	2	3	4	5	6
	2	7	4	10	8	6

三、堆（heap）上的操作

1.make_heap() ：-此函数用于将容器中的范围转换为堆。

2.front() ：-此函数显示堆的第一个元素，即最大数量。

3.push_heap() :-此函数用于将元素插入堆。堆的大小增加 1。将新元素适当地放置在堆中。

4.pop_heap():-此函数用于删除堆的最大元素。堆的大小减少 1。在执行此操作后，将相应地重新组织堆元素。

5. sort_heap() ：-此函数用于对堆进行排序。完成此操作后，容器不再是堆。

6. is heap():-此函数用于检查容器是否为堆。通常，在大多数实现中，反向排序的容器被视为堆。如果容器是堆，则返回 true;否则返回 false。

四、优先队列（priority_queue）

priority_queue 又称为优先队列，其底层是用堆来进行实现的。在优先队列中，队首元素一定是当前队列中优先级最高的那一个。

(1)priority_queue 的定义

定义的写法和其他 STL 容器相同， typename 可以是任意基本数据类型或容器：

priority_queue<typename> name;

(2)push()

push(x)将令 x 入队,时间复杂度为 O(logN)

(3)top()

top()可以获得队首元素(即堆顶元素)，时间复杂度为 O(1)，在使用 top()函数前，必须用 empty()判断优先队列是否为空。

(4)pop()

pop()可令首元素(即堆顶元素)出队,时间复杂度为 O(logN)。在使用 pop()函数前，必须用 empty()判断优先队列是否为空。

【例 59.1】 合并果子

【题目描述】

在一个果园里，多多已经将所有的果子打了下来，而且按果子的不同种类分成了不同的堆。多多决定把所有的果子合成一堆。

每一次合并，多多可以把两堆果子合并到一起，消耗的体力等于两堆果子的重量之和。可以看出，所有的果子经过 n-1 次合并之后，就只剩下一堆了。多多在合并果子时总共消耗的体力等于每次合并所耗体力之和。

因为还要花大力气把这些果子搬回家，所以多多在合并果子时要尽可能地节省体力。假定每个果子重量都为 1，并且已知果子的种类数和每种果子的数目，你的任务是设计出合并的次序方案，使多多耗费的体力最少，并输出这个最小的体力耗费值。

例如有 3 种果子，数目依次为 1，2，9。可以先将 1、2 堆合并，新堆数目为 3，耗费体力为 3。接着，将新堆与原先的第三堆合并，又得到新的堆，数目为 12，耗费体力为 12。所以多多总共耗费体力=3+12=15。可以证明 15 为最小的体力耗费值。

【输入格式】

两行，第一行是一个整数 n（1≤n≤30000），表示果子的种类数。第二行包含 n 个整数，用空格分隔，第 i 个整数 ai（1≤ai≤20000）是第 i 种果子的数目。

【输出格式】

一行，这一行只包含一个整数，也就是最小的体力耗费值。输入数据保证这个值小于 2^{31}。

【样例输入】

3

1 2 9

【样例输出】

15

【代码实现 1】

```cpp
#include<bits/stdc++.h>
using namespace std;
int a[100001],tot=0,n,a1,a2,sum;
int main(){
    cin>>n;
    for(int i=1;i<=n;i++) cin>>a[i];
    make_heap(a+1,a+n+1,greater<int>());//建立小根堆
    int m=n;//取出 n 的大小，方便进行维护
    for(int i=1;i<n;i++){
        a1=a[1];//取出最小元素
        pop_heap(a+1,a+m+1,greater<int>());//删除最小的并维护
        a2=a[1];//其次小的元素
        pop_heap(a+1,a+m,greater<int>());//取出其次小的并进行维护
        sum=a1 + a2;//算出需要的体力
        tot=tot+sum;
        a[m-1]=sum;//将生成的元素再次放回
        push_heap(a+1,a+m,greater<int>()); //再次维护
        m--;
    }
    cout<<tot;
    return 0;
}
```

【代码实现 2】

```
#include<bits/stdc++.h>
using namespace std;
int n,m,ans,a,b;
int main( ){
    scanf("%d",&n);
    priority_queue<int,vector<int> ,greater<int> > pq;
    for(int i=0;i<n;i++){
        scanf("%d" , &m );
        pq.push(m);
    }
    if(pq.size()==1){
        printf("%d\n", pq.top());
        return 0;
    }
    while(pq.size()>1){
        a=pq.top();pq.pop();
        b=pq.top();pq.pop();
        pq.push(a+b);
        ans+=a+b;
    }
    printf("%d\n",ans);
    return 0;
}
```

【例 59.2】 [NOIP2010 普及组] 接水问题

【题目描述】

学校里有一个水房，水房里一共装有 m 个龙头可供同学们打开水，每个龙头每秒钟的供水量相等，均为 1。

现在有 n 名同学准备接水，他们的初始接水顺序已经确定。将这些同学按接水顺序从 1 到 n 编号，i 号同学的接水量为 w_i。接水开始时，1 到 m 号同学各占一个水龙头，并同时打开水龙头接水。当其中某名同学 j 完成其接水量要求 w_j后，下一名排队等候接水的同学 k 马上接替 j 同学的位置开始接水。这个换人的过程是瞬间完成的，且没有任何水的浪费。即 j 同学第 x 秒结束时完成接水，则 k 同学第 x+1 秒立刻开始接水。若当前接水人数 n’不足 m，则只有 n’个龙头供水，其它 m - n’个龙头关闭。

现在给出 n 名同学的接水量，按照上述接水规则，问所有同学都接完水需要多少秒。

【输入格式】

第一行两个整数 n 和 m，用一个空格隔开，分别表示接水人数和龙头个数。

第二行 n 个整数 w_1 , w_2 . . . , w_n，每两个整数之间用一个空格隔开，w_i 表示 i 号同学的接水量。

【输出格式】

一个整数，表示接水所需的总时间。

【样例输入】

5 3

4 4 1 2 1

【样例输出】

4

【代码实现】

```
1. #include<bits/stdc++.h>
2. using namespace std;
3. int n,m,x,maxx,t;
4. priority_queue<int,vector<int>,greater<int> > q;
5. int main(){
6.     cin>>n>>m;
7.     for(int i=1;i<=m;i++){
8.       cin>>x;
9.       q.push(x);
10.       maxx=max(maxx,x);
11.     }
12.
13.     for(int i=m+1;i<=n;i++){
14.       cin>>x;
15.       t=q.top();
16.       q.pop();
17.       q.push(t+x);
18.       maxx=max(maxx,t+x);
19.     }
20.     cout<<maxx;
21.     return 0;
22. }
```

练 59.1 第 n 大的数(趣味编程)

【题目描述】

有 10 个互不相同的整数，不用排序，求出其中第 n 大的数（），即有 n-1 个数比它大，其余的数都比它小。定义一个找数列{99，200，95，87，98，-12，30，87，75，-25}中第 n 大的数的函数，利用它输出第 n 大的数？

【输入格式】

一行一个整数，表示 n。若 n 不合法，则重新输入。

【输出格式】

一行一个整数，表示该数列中第 n 大的数。

【样例输入】

2

【样例输出】

99

【代码实现】

```
#include <iostream>
using namespace std;
int maxn(int b[], int m){
    bool p = true;
    int x, num, cnt, i = 0;
    while (p && i < 10)
    {
        x = b[i];
        num = cnt = 0;
        for (int j = 0; j < 10; j++)
            if (x < b[j])
                num++;
            else if (x == b[j])
                ++cnt;
        if (num + 1 <= m && num + cnt >= m) //当前答案符合要求
            p = false;
        else
            i++;
    }
    return x;}
int main(){
    int n, a[10] = {99,200,95,87,98,-12,30,87,75,-25};
    do    {//输入符合标准的n
        cin >> n;
    } while (n < 1 || n > 10);
    cout << maxn(a, n) << endl;
return 0;
}
```

【代码实现 2】

```
#include <bits/stdc++.h>
using namespace std;
int main(){
int n, a[10] = {99,200,95,87,98,-12,30,87,75,-25};
while (cin >> n, n < 1 || n > 10);
    nth_element(a, a + n - 1, a + 10, greater<int>());
printf("%d\n", a[n - 1]);
}
```

谢谢！