


常用代码模板2——数据结构

作者:  YXC (<https://www.acwing.com/user/myspace/index/1/>), 2019-07-31 21:34:51, 阅读 15101

瀛

算法基础课相关代码模板

113

- 活动链接 —— 算法基础课 (<https://www.acwing.com/activity/content/11/>)

瀛

单链表 —— 模板题 AcWing 826. 单链表 (<https://www.acwing.com/problem/content/828/>)

→

150

// head存储链头, e[]存储节点的值, ne[]存储节点的next指针, idx表示当前用到了哪个节点
int head, e[N], ne[N], idx;

```
// 初始化
void init()
{
    head = -1;
    idx = 0;
}

// 在链表头插入一个数a
void insert(int a)
{
    e[idx] = a, ne[idx] = head, head = idx ++ ;
}

// 将头结点删除, 需要保证头结点存在
void remove()
{
    head = ne[head];
}
```

双链表 —— 模板题 AcWing 827. 双链表 (<https://www.acwing.com/problem/content/829/>)

// e[]表示节点的值, l[]表示节点的左指针, r[]表示节点的右指针, idx表示当前用到了哪个节点
int e[N], l[N], r[N], idx;

```
// 初始化
void init()
{
    // 0是左端点, 1是右端点
    r[0] = 1, l[1] = 0;
    idx = 2;
}

// 在节点a的右边插入一个数x
void insert(int a, int x)
{
    e[idx] = x;
    l[idx] = a, r[idx] = r[a];
    l[r[a]] = idx, r[a] = idx ++ ;
}

// 删除节点a
void remove(int a)
{
    l[r[a]] = l[a];
    r[l[a]] = r[a];
}
```

栈 —— 模板题 AcWing 828. 模拟栈 (<https://www.acwing.com/problem/content/830/>)

```
// tt表示栈顶
int stk[N], tt = 0;

// 向栈顶插入一个数
stk[ ++ tt] = x;

// 从栈顶弹出一个数
tt --;

// 栈顶的值
stk[tt];

// 判断栈是否为空
if (tt > 0)
{

}
```

队列 —— 模板题 AcWing 829. 模拟队列 (<https://www.acwing.com/problem/content/831/>)

1. 普通队列：

```
// hh 表示队头，tt表示队尾
int q[N], hh = 0, tt = -1;

// 向队尾插入一个数
q[ ++ tt] = x;

// 从队头弹出一个数
hh ++;

// 队头的值
q[hh];

// 判断队列是否为空
if (hh <= tt)
{

}
```

2. 循环队列

```
// hh 表示队头，tt表示队尾的后一个位置
int q[N], hh = 0, tt = 0;

// 向队尾插入一个数
q[tt ++] = x;
if (tt == N) tt = 0;

// 从队头弹出一个数
hh ++;
if (hh == N) hh = 0;

// 队头的值
q[hh];

// 判断队列是否为空
if (hh != tt)
{

}
```

单调栈 —— 模板题 AcWing 830. 单调栈 (<https://www.acwing.com/problem/content/832/>)

常见模型：找出每个数左边离它最近的比它大/小的数

```
int tt = 0;
for (int i = 1; i <= n; i++)
{
    while (tt && check(stk[tt], i)) tt--;
    stk[++tt] = i;
}
```

单调队列 —— 模板题 AcWing 154. 滑动窗口 (<https://www.acwing.com/problem/content/156/>)

常见模型：找出滑动窗口中的最大值/最小值

```
int hh = 0, tt = -1;
for (int i = 0; i < n; i++)
{
    while (hh <= tt && check_out(q[hh])) hh++; // 判断队头是否滑出窗口
    while (hh <= tt && check(q[tt], i)) tt--;
    q[++tt] = i;
}
```

KMP —— 模板题 AcWing 831. KMP字符串 (<https://www.acwing.com/problem/content/833/>)

// s[]是长文本，p[]是模式串，n是s的长度，m是p的长度

求模式串的Next数组：

```
for (int i = 2, j = 0; i <= m; i++)
{
    while (j && p[i] != p[j + 1]) j = ne[j];
    if (p[i] == p[j + 1]) j++;
    ne[i] = j;
}
```

// 匹配

```
for (int i = 1, j = 0; i <= n; i++)
{
    while (j && s[i] != p[j + 1]) j = ne[j];
    if (s[i] == p[j + 1]) j++;
    if (j == m)
    {
        j = ne[j];
        // 匹配成功后的逻辑
    }
}
```

Trie树 —— 模板题 AcWing 835. Trie字符串统计 (<https://www.acwing.com/problem/content/837/>)

```

int son[N][26], cnt[N], idx;
// 0号点既是根节点，又是空节点
// son[][]存储树中每个节点的子节点
// cnt[]存储以每个节点结尾的单词数量

// 插入一个字符串
void insert(char *str)
{
    int p = 0;
    for (int i = 0; str[i]; i++)
    {
        int u = str[i] - 'a';
        if (!son[p][u]) son[p][u] = ++idx;
        p = son[p][u];
    }
    cnt[p]++;
}

// 查询字符串出现的次数
int query(char *str)
{
    int p = 0;
    for (int i = 0; str[i]; i++)
    {
        int u = str[i] - 'a';
        if (!son[p][u]) return 0;
        p = son[p][u];
    }
    return cnt[p];
}

```

并查集 —— 模板题 AcWing 836. 合并集合 (<https://www.acwing.com/problem/content/838/>), **AcWing 837. 连通块中点的数量** (<https://www.acwing.com/problem/content/839/>)

(1)朴素并查集：

```

int p[N]; //存储每个点的祖宗节点

// 返回x的祖宗节点
int find(int x)
{
    if (p[x] != x) p[x] = find(p[x]);
    return p[x];
}

// 初始化，假定节点编号是1~n
for (int i = 1; i <= n; i++) p[i] = i;

// 合并a和b所在的两个集合：
p[find(a)] = find(b);

```

(2)维护size的并查集：

```

int p[N], size[N];
//p[]存储每个点的祖宗节点, size[]只有祖宗节点的有意义，表示祖宗节点所在集合中的点的数量

// 返回x的祖宗节点
int find(int x)
{
    if (p[x] != x) p[x] = find(p[x]);
    return p[x];
}

// 初始化，假定节点编号是1~n
for (int i = 1; i <= n; i++)
{

```

```
p[i] = i;
size[i] = 1;
}

// 合并a和b所在的两个集合：
size[find(b)] += size[find(a)];
p[find(a)] = find(b);
```

(3)维护到祖宗节点距离的并查集：

```
int p[N], d[N];
//p[]存储每个点的祖宗节点, d[x]存储x到p[x]的距离

// 返回x的祖宗节点
int find(int x)
{
    if (p[x] != x)
    {
        int u = find(p[x]);
        d[x] += d[p[x]];
        p[x] = u;
    }
    return p[x];
}

// 初始化，假定节点编号是1~n
for (int i = 1; i <= n; i++)
{
    p[i] = i;
    d[i] = 0;
}

// 合并a和b所在的两个集合：
p[find(a)] = find(b);
d[find(a)] = distance; // 根据具体问题，初始化find(a)的偏移量
```

堆 —— 模板题 AcWing 838. 堆排序 (<https://www.acwing.com/problem/content/840/>), AcWing 839. 模拟堆 (<https://www.acwing.com/problem/content/841/>)

```
// h[N]存储堆中的值, h[1]是堆顶, x的左儿子是2x, 右儿子是2x + 1
```

```
// ph[k]存储第k个插入的点在堆中的位置
```

```
// hp[k]存储堆中下标是k的点是第几个插入的
```

```
int h[N], ph[N], hp[N], size;
```

```
// 交换两个点, 及其映射关系
```

```
void heap_swap(int a, int b)
```

```
{  
    swap(ph[hp[a]],ph[hp[b]]);  
    swap(hp[a], hp[b]);  
    swap(h[a], h[b]);  
}
```

```
void down(int u)
```

```
{  
    int t = u;  
    if (u * 2 <= size && h[u * 2] < h[t]) t = u * 2;  
    if (u * 2 + 1 <= size && h[u * 2 + 1] < h[t]) t = u * 2 + 1;  
    if (u != t)  
    {  
        heap_swap(u, t);  
        down(t);  
    }  
}
```

```
void up(int u)
```

```
{  
    while (u / 2 && h[u] < h[u / 2])  
    {  
        heap_swap(u, u / 2);  
        u >>= 1;  
    }  
}
```

```
// O(n)建堆
```

```
for (int i = n / 2; i; i --) down(i);
```

一般哈希 —— 模板题 AcWing 840. 模拟散列表 (<https://www.acwing.com/problem/content/842/>)

(1) 拉链法

```
int h[N], e[N], ne[N], idx;

// 向哈希表中插入一个数
void insert(int x)
{
    int k = (x % N + N) % N;
    e[idx] = x;
    ne[idx] = h[k];
    h[k] = idx ++ ;
}

// 在哈希表中查询某个数是否存在
bool find(int x)
{
    int k = (x % N + N) % N;
    for (int i = h[k]; i != -1; i = ne[i])
        if (e[i] == x)
            return true;

    return false;
}
```

(2) 开放寻址法

```
int h[N];

// 如果x在哈希表中，返回x的下标；如果x不在哈希表中，返回x应该插入的位置
int find(int x)
{
    int t = (x % N + N) % N;
    while (h[t] != null && h[t] != x)
    {
        t ++ ;
        if (t == N) t = 0;
    }
    return t;
}
```

字符串哈希 —— 模板题 AcWing 841. 字符串哈希 (<https://www.acwing.com/problem/content/843/>)

核心思想：将字符串看成P进制数，P的经验值是131或13331，取这两个值的冲突概率低

小技巧：取模的数用 2^{64} ，这样直接用unsigned long long存储，溢出的结果就是取模的结果

```
typedef unsigned long long ULL;
ULL h[N], p[N]; // h[k]存储字符串前k个字母的哈希值, p[k]存储  $P^k \bmod 2^{64}$ 

// 初始化
p[0] = 1;
for (int i = 1; i <= n; i ++ )
{
    h[i] = h[i - 1] * P + str[i];
    p[i] = p[i - 1] * P;
}

// 计算子串 str[l ~ r] 的哈希值
ULL get(int l, int r)
{
    return h[r] - h[l - 1] * p[r - l + 1];
}
```

C++ STL简介

vector, 变长数组，倍增的思想

size() 返回元素个数

empty() 返回是否为空

clear() 清空

front()/back()
push_back()/pop_back()
begin()/end()
[]
支持比较运算，按字典序

pair<int, int>
first, 第一个元素
second, 第二个元素
支持比较运算，以first为第一关键字，以second为第二关键字（字典序）

string, 字符串
size()/length() 返回字符串长度
empty()
clear()
substr(起始下标, (子串长度)) 返回子串
c_str() 返回字符串所在字符数组的起始地址

queue, 队列
size()
empty()
push() 向队尾插入一个元素
front() 返回队头元素
back() 返回队尾元素
pop() 弹出队头元素

priority_queue, 优先队列，默认是大根堆
push() 插入一个元素
top() 返回堆顶元素
pop() 弹出堆顶元素
定义成小根堆的方式：priority_queue<int, vector<int>, greater<int>> q;

stack, 栈
size()
empty()
push() 向栈顶插入一个元素
top() 返回栈顶元素
pop() 弹出栈顶元素

deque, 双端队列
size()
empty()
clear()
front()/back()
push_back()/pop_back()
push_front()/pop_front()
begin()/end()
[]

set, map, multiset, multimap, 基于平衡二叉树（红黑树），动态维护有序序列
size()
empty()
clear()
begin()/end()
++, -- 返回前驱和后继，时间复杂度 $O(\log n)$

set/multiset
insert() 插入一个数
find() 查找一个数
count() 返回某一个数的个数
erase()
 (1) 输入是一个数x，删除所有x $O(k + \log n)$
 (2) 输入一个迭代器，删除这个迭代器
lower_bound()/upper_bound()
 lower_bound(x) 返回大于等于x的最小的数的迭代器
 upper_bound(x) 返回大于x的最小的数的迭代器
map/multimap
insert() 插入的数是一个pair


```
erase() 输入的参数是pair或者迭代器
find()
[] 注意multimap不支持此操作。时间复杂度是 O(logn)
lower_bound()/upper_bound()
```

unordered_set, unordered_map, unordered_multiset, unordered_multimap, 哈希表
和上面类似，增删改查的时间复杂度是 O(1)
不支持 lower_bound()/upper_bound()，迭代器的++, --

bitset, 压位
bitset<10000> s;
~, &, |, ^
>>, <<
==, !=
[]

count() 返回有多少个1


any() 判断是否至少有一个1
none() 判断是否全为0




set() 把所有位置成1
set(k, v) 将第k位变成v
reset() 把所有位变成0
flip() 等价于~
flip(k) 把第k位取反

评论列表：


在这里写评论...（支持MarkDown和Latex语法）


提交评论

 henhen敲 (<https://www.acwing.com/user/myspace/index/28194/>) 13天前 回复
(<https://www.acwing.com/user/myspace/index/28194/>)
维护到祖宗节点距离的并查集中 最后 这两句
p[find(a)] = find(b);
d[find(a)] = distance;
find(a) 这句执行一次就行了吧
应该是
int r = find(a);
p[r] = find(b);
d[r] = distance;

 aiffy (<https://www.acwing.com/user/myspace/index/34853/>) 1个月前 回复
(<https://www.acwing.com/user/myspace/index/34853/>)
如果用java，有什么可以代替STL的嘛~(小白想问一下)
 traveleryyk (<https://www.acwing.com/user/myspace/index/38223/>) 1个月前 回复
(<https://www.acwing.com/user/myspace/index/38223/>)
java有自己数据结构啊，比如ArrayList、HashMap、HashSet之类（小白发言）
 yxc (<https://www.acwing.com/user/myspace/index/1/>) 1个月前 回复了 traveleryyk 的评论
(<https://www.acwing.com/user/myspace/index/1/>)
对滴。

 憨憨_9 (<https://www.acwing.com/user/myspace/index/33811/>) 2个月前 回复
(<https://www.acwing.com/user/myspace/index/33811/>)
y总，循环对列那有点不太理解，tt表示对尾的下一个位置，那为啥hh==tt时队列为空呢，tt到N不是会变成零吗，那队满时不也是tt==hh?

 憨憨_9 (<https://www.acwing.com/user/myspace/index/33811/>) 2个月前 回复
(<https://www.acwing.com/user/myspace/index/33811/>)
这样的话，队空时满不都是hh==tt，那不是不能做为队空的判断条件吗？

 陌_5 (<https://www.acwing.com/user/myspace/index/11139/>) 2个月前 回复了 憨憨_9 的评论
(<https://www.acwing.com/user/myspace/index/11139/>)

初始化的时候tt = -1, hh = 0, tt始终是指向尾的, hh始终是指向头的,

tt并不是如你所说是表示对尾的下一个位置

注意y总代码, 在入队列的时候是 tt 而不是 tt

当tt < hh 表示队列为空

当tt >= hh 表示队列里面有数

憨憨_9 (<https://www.acwing.com/user/myspace/index/33811/>) 2个月前 回复了 陌_5 的评论
(<https://www.acwing.com/user/myspace/index/33811/>)
普通对列tt初始化为-1, 可是循环对列那tt初始值是0啊

yxc (<https://www.acwing.com/user/myspace/index/1/>) 2个月前 回复了 憨憨_9 的评论 回复
(<https://www.acwing.com/user/myspace/index/1/>)
循环对列最多装n-1个元素

憨憨_9 (<https://www.acwing.com/user/myspace/index/33811/>) 2个月前 回复了 yxc 的评论
(<https://www.acwing.com/user/myspace/index/33811/>)
哦哦, 这样啊, 谢谢y总!



alin (<https://www.acwing.com/user/myspace/index/30105/>) 3个月前 回复
(<https://www.acwing.com/user/myspace/index/30105/>)
在单链表中, 如果要删除链表中第k个节点怎么操作呢? ne[k] = ne[ne[k]]这种只是删除了第k个插入的数后面的数是吧

alin (<https://www.acwing.com/user/myspace/index/30105/>) 3个月前 回复
(<https://www.acwing.com/user/myspace/index/30105/>)
while(p != -1 && j < k-1)
{
p = ne[p]; j++;
}
ne[p] = ne[ne[p]];
我这样写删除第2个节点没事 但是删除第一个节点不知道除了特判还有别的办法没

alin (<https://www.acwing.com/user/myspace/index/30105/>) 3个月前 回复
(<https://www.acwing.com/user/myspace/index/30105/>)
// 将在链表中节点号是k的点删除 是从零开始

```
void remove(int k)
{
    if(k==0)
    {
        head = ne[head];
        return;
    }
    int p = head, j = 0;
    while(p != -1 && j < k-1)
    {
        p = ne[p]; j++;
    }
    ne[p] = ne[ne[p]];

    //ne[k] = ne[ne[k]];
}
```

alin (<https://www.acwing.com/user/myspace/index/30105/>) 3个月前 回复
(<https://www.acwing.com/user/myspace/index/30105/>)
求大佬指点

留基伯 (<https://www.acwing.com/user/myspace/index/31837/>) 3个月前 回复了 alin 的评论
(<https://www.acwing.com/user/myspace/index/31837/>)
题目要求是删去“第k个加入的数”, 所以这么做, 同理对k-1这么操作就可, 但是若要删除链表中第k个数, 需要把链表遍历至第k个数

yxc (<https://www.acwing.com/user/myspace/index/1/>) 3个月前 回复
(<https://www.acwing.com/user/myspace/index/1/>)
对滴, 如果要删除链表中第k个节点, 就只能先从头遍历把它的前一个点找到, 然后再删。




九頭竜_八一 (<https://www.acwing.com/user/myspace/index/15301/>) 3个月前 回复
(<https://www.acwing.com/user/myspace/index/15301/>)
bitset 压位...


yxc (<https://www.acwing.com/user/myspace/index/1/>) 3个月前 回复
(<https://www.acwing.com/user/myspace/index/1/>)
棒!





wjl (<https://www.acwing.com/user/myspace/index/15872/>) 4个月前 回复
(<https://www.acwing.com/user/myspace/index/15872/>)
并查集中, 维护到祖先距离的方法中, $d[I] = 0$; 的I应该是小写?


yxc (<https://www.acwing.com/user/myspace/index/1/>) 3个月前 回复
(<https://www.acwing.com/user/myspace/index/1/>)
已修正

 lxxs (<https://www.acwing.com/user/myspace/index/5870/>) 4个月前 回复
(<https://www.acwing.com/user/myspace/index/5870/>)
推荐一种变成小根堆的方式，把输入的数乘以-1存入堆，取出的时候再乘-1，最后就实现了小根堆，而且也不是很麻烦。记忆量小。


 yxc (<https://www.acwing.com/user/myspace/index/1/>) 4个月前 回复
(<https://www.acwing.com/user/myspace/index/1/>)
不错，可以的。


 lxxs (<https://www.acwing.com/user/myspace/index/5870/>) 4个月前 回复
(<https://www.acwing.com/user/myspace/index/5870/>)
如果计算队列中元素的个数，普通队列可以直接用hh-tt计算，循环队列只能用变量来存储。还是有循环队列的计算公式。


 yxc (<https://www.acwing.com/user/myspace/index/1/>) 4个月前 回复
(<https://www.acwing.com/user/myspace/index/1/>)
如果求个数就是 $ser - hh$ ，如果 $index(1)$ 个数是0；如果 $hh > tt$ ，个数就是 $n + tt - hh$ ， n 是队列数组的长度。


 lxxs (<https://www.acwing.com/user/myspace/index/5870/>) 4个月前 回复了 yxc 的评论 回复
(<https://www.acwing.com/user/myspace/index/5870/>)
蟹蟹


 烛之武 (<https://www.acwing.com/user/myspace/index/1099/>) 4个月前 回复
(<https://www.acwing.com/user/myspace/index/1099/>)
是不是能用普通队列的地方，都可以用循环队列？

 yxc (<https://www.acwing.com/user/myspace/index/1/>) 4个月前 回复
(<https://www.acwing.com/user/myspace/index/1/>)
对滴

 Onebelieve (<https://www.acwing.com/user/myspace/index/27463/>) 5个月前 回复
(<https://www.acwing.com/user/myspace/index/27463/>)
tql


 牙疼 (<https://www.acwing.com/user/myspace/index/11683/>) 5个月前 回复
(<https://www.acwing.com/user/myspace/index/11683/>)
kmp那里，模式串和模板串的名称是不是反了啊？模式串是不是在模板串里查找的串？


 yxc (<https://www.acwing.com/user/myspace/index/1/>) 5个月前 回复
(<https://www.acwing.com/user/myspace/index/1/>)
对更反


 Diamondz (<https://www.acwing.com/user/myspace/index/1156/>) 6个月前 回复
(<https://www.acwing.com/user/myspace/index/1156/>)
维护size的并查集里合并a和b所在的两个集合时：


```
p[find(a)] = find(b);  
size[b] += size[a];
```

应该是 $size[find(b)] += size[find(a)]$ 吧，因为只有祖宗节点的 size 才是有意义的。


 yxc (<https://www.acwing.com/user/myspace/index/1/>) 6个月前 回复
(<https://www.acwing.com/user/myspace/index/1/>)
感谢指正，已修正。另外需要执行 $size[find(b)] += size[find(a)]$ ；再执行 $p[find(a)] = find(b)$ ；，否则在更新 size 时 find(a) 就不是自己本身的祖先，而是 b 的祖先了。

 发光二极管 (<https://www.acwing.com/user/myspace/index/13176/>) 6个月前 回复
(<https://www.acwing.com/user/myspace/index/13176/>)
y总这里单调队列是不是本质上还是一个单调栈，只是用头部来确保只有在窗口内的值才能保留在队列内？


 yxc (<https://www.acwing.com/user/myspace/index/1/>) 6个月前 回复
(<https://www.acwing.com/user/myspace/index/1/>)
单调队列中的“队列”其实是“双端队列”的简写，它可以在队头队尾添加或者删除元素，确实比较容易让人误解。

 发光二极管 (<https://www.acwing.com/user/myspace/index/13176/>) 6个月前 回复了 yxc 的评论 回复
(<https://www.acwing.com/user/myspace/index/13176/>)

了解~。不过就模板题还有多重背包三来说，单调队列的操作实际上就是在单调栈的基础上，新增了通过队头删除元素来保证窗口大小的限制。换句话说，单调栈也就相当于窗口大小为无穷的单调队列。我只是觉得单调栈和单调队列为了使得栈(队列)内部单调而执行的操作的基本思想是一样的^ - ^。谢谢y总答复，y总记得早点休息啊，回复我这条消息都挺晚了，保命要紧(ω\)

 yxc (<https://www.acwing.com/user/myspace/index/1/>) 6个月前 回复了 发光二极管 的评论 回复
(<https://www.acwing.com/user/myspace/index/1/>)

对滴，这两者本质相同，都是先去冗余元素，然后发现所有元素单调，所以求最值时在开头或者结尾找就行。我虽然睡得晚，但是起得夜晚啊hh，只是时差不同。

 发光二极管 (<https://www.acwing.com/user/myspace/index/13176/>) 6个月前 回复了 yxc 的评论 回复
(<https://www.acwing.com/user/myspace/index/13176/>)
嗯嗯，谢谢y总，:)



Prisoner (<https://www.acwing.com/user/myspace/index/1655/>) 7个月前 回复

问老师可不可以补充一下循环队列的写法，辛苦了！
(<https://www.acwing.com/user/myspace/index/1655/>)



yxc (<https://www.acwing.com/user/myspace/index/1/>) 7个月前 回复

(<https://www.acwing.com/user/myspace/index/1/>)



Prisoner (<https://www.acwing.com/user/myspace/index/1655/>) 7个月前 回复了 yxc 的评论

(<https://www.acwing.com/user/myspace/index/1655/>)

^_^



阿力 (<https://www.acwing.com/user/myspace/index/8557/>) 7个月前 回复

STL总结的很好
(<https://www.acwing.com/user/myspace/index/8557/>)



yxc (<https://www.acwing.com/user/myspace/index/1/>) 7个月前 回复

(<https://www.acwing.com/user/myspace/index/1/>)



a645852220 (<https://www.acwing.com/user/myspace/index/4276/>) 9个月前 回复

(<https://www.acwing.com/user/myspace/index/4276/>)

建议把ph和hp这两个数组名字换成pos,rk

//pos[k]存储第k个插入的点在堆中的位置

//rk[k]存储堆中下标是k的点是第几个插入的

这样子交换节点会容易看很多，看y大佬这个交换节点想了好久，很容易乱，最后还是在纸上画图模拟才算理解

```
void heap_swap(int a, int b) {
    int x = rk[a], y = rk[b];
    swap(pos[x], pos[y]);
    swap(rk[a], rk[b]);
    swap(h[a], h[b]);
}
```



yxc (<https://www.acwing.com/user/myspace/index/1/>) 8个月前 回复

(<https://www.acwing.com/user/myspace/index/1/>) 哈哈这里确实比较坑，这里使用pos表示在堆外的位置，也就是第几个插入的，用heap表示在堆中的位置，所以ph表示的就是position -> heap 相当于求出当前点对应到堆中的位置，hp表示heap -> position，相当于求出堆中点对应到position中的位置~



番茄酱 (<https://www.acwing.com/user/myspace/index/6828/>) 9个月前 回复

单调栈那里check里面应该是stk[tt]
(<https://www.acwing.com/user/myspace/index/6828/>)



yxc (<https://www.acwing.com/user/myspace/index/1/>) 9个月前 回复

(<https://www.acwing.com/user/myspace/index/1/>) 多谢指正已修改



T-SHLoRk (<https://www.acwing.com/user/myspace/index/6314/>) 10个月前 回复

想请问一下“维护到祖宗节点距离的并查集”有哪些典型的应用场景嘛？做了一些题目还没有用到过
(<https://www.acwing.com/user/myspace/index/6314/>)



yxc (<https://www.acwing.com/user/myspace/index/1/>) 10个月前 回复

(<https://www.acwing.com/user/myspace/index/1/>) AcWing 239. 银河英雄传说 (<https://www.acwing.com/problem/content/description/240/>) 和 AcWing 240. 食物链 (<https://www.acwing.com/problem/content/242/>)



T-SHLoRk (<https://www.acwing.com/user/myspace/index/6314/>) 10个月前 回复了 yxc 的评论

(<https://www.acwing.com/user/myspace/index/6314/>)

多谢呀，这类题目相较于前面两种是稍微难了一点嘛



yxc (<https://www.acwing.com/user/myspace/index/1/>) 10个月前 回复了 T-SHLoRk 的评论

(<https://www.acwing.com/user/myspace/index/1/>)

难了不少



Clever_Jimmy (<https://www.acwing.com/user/myspace/index/6289/>) 8个月前 回复了 yxc

(<https://www.acwing.com/user/myspace/index/6289/>)

是不是还可以用拓展域来解决啊qwq



yxc (<https://www.acwing.com/user/myspace/index/1/>) 8个月前 回复了 Clever_Jimmy 的评论

(<https://www.acwing.com/user/myspace/index/1/>)

可以的，不过推荐用维护距离的并查集，会省空间，数组长度和类别的数量无关。



Clever_Jimmy (<https://www.acwing.com/user/myspace/index/6289/>) 8个月前 回复了 yxc

(<https://www.acwing.com/user/myspace/index/6289/>)

谢谢闫总！



TK (<https://www.acwing.com/user/myspace/index/5667/>) 10个月前 回复

老师，multimap和unordered_multimap不支持 []啊
(<https://www.acwing.com/user/myspace/index/5667/>)



yxc (<https://www.acwing.com/user/myspace/index/1/>) 10个月前 回复

啊是的，已修正
(<https://www.acwing.com/user/myspace/index/1/>)



bseazh (<https://www.acwing.com/user/myspace/index/5416/>) 10个月前 回复

STL——String中里面的size函数写错了
(<https://www.acwing.com/user/myspace/index/5416/>)



yxc (<https://www.acwing.com/user/myspace/index/1/>) 10个月前 回复

已修正
(<https://www.acwing.com/user/myspace/index/1/>)