E2-A题

20376388梁林楠

题目

- 本题是堆的一种拓展应用，你的任务是实现一个小根堆，即根节点为所有元素最小值的堆，需要支持插入元素、查询堆顶元素、删除堆顶元素、删除堆中任意元素四个操作。

思路

- 可删除堆也是堆的一个分支。它和对顶堆的使用是差不多的，都是为了解决用朴素堆解决
- 不了的问题。对顶堆解决的是朴素堆不支持单点查询的问题，而可删除堆就解决了朴素堆
- 不支持任意删除的问题。
- 我们知道，优先队列只能删除堆顶元素，然而我们并不能删掉其他元素，有时甚至找不到
- 要删的元素。这时怎么办呢？于是，可删除堆出场了。
- 可删除堆的实现原理也比较简单。我们建一个临时堆，如果要删除哪个元素，就把哪个元
- 素压入临时堆，然后待此元素和正常堆的堆顶元素相同时（即两个堆顶一样），就同时pop
- 掉。
- 那么为什么这样做是正确的呢？
- 我们发现，我们在调用堆的时候，只能取出并就堆顶元素进行操作，也就是说，排在下面
- 的那些元素，在没到达堆顶之前，是不对结果造成任何影响的。那么我们就自然而然的觉
- 得，我先把这个要删除的元素挂出来，什么时候它到达堆顶了，就把它跳过（实现就是
- pop）。就好比追查一个通缉犯，我们卡住安检口，拿着犯人的照片，犯人一露头就直接被
- 带走了，不会对机场的秩序造成任何的影响。由于我们的临时堆跟正常堆的方向是一样
- 的，所以当临时堆里有多个要删除的元素的时候，他们也是排好序的，不会出现下一个要
- 删除的元素比上一个提前出现的情况。

代码

```
#include<stdio.h>
#include<stdlib.h>
#define maxn 200005
int Q[maxn],Q1[maxn],T,op,num,num1,x;
void swap(int *p,int *q){
        int t=*p;
        *p=*q;
        *q=t;
}
void update(int x){
        if(x==1) return;
        if(Q[x>>1]>Q[x]){
                swap(&Q[x],&Q[x>>1]);
                update(x>>1);
        }
}
void update1(int x){
        if(x==1) return;
        if(Q1[x>>1]>Q1[x]){
                swap(&Q1[x],&Q1[x>>1]);
                update1(x>>1);
        }
}
```

代码

```
void download(int x){
        if((x<<1)>num) return;
        if((x<<1|1)>num){
                if(Q[x]>Q[x<<1]) swap(&Q[x],&Q[x<<1]);
                download(x<<1);
                return;
        }
        int temp= Q[x<<1] < Q[x<<1|1] ? x<<1 : x<<1|1;
        if(Q[x]>Q[temp]){
                swap(&Q[x],&Q[temp]);
                download(temp);
        }
}
void download1(int x){
        if((x<<1)>num1) return;
        if((x<<1|1)>num1){
                if(Q1[x]>Q1[x<<1]) swap(&Q1[x],&Q1[x<<1]);
                download1(x<<1);
                return;
        }
        int temp= Q1[x<<1] < Q1[x<<1|1] ? x<<1 : x<<1|1;
        if(Q1[x]>Q1[temp]){
                swap(&Q1[x],&Q1[temp]);
                download1(temp);
        }
}
```

代码

```
scanf("%d",&op);
if(op==1){
		scanf("%d",&x);
		Q[++num]=x;
		update(num);
}
if(op==2){
		int ans=Q[1];
		while(num1>0 && ans==Q1[1]){
		//printf("%d %d\n",ans,Q1[1]);
		swap(&Q[1],&Q[num]);
		num--;
		download(1);
		swap(&Q1[1],&Q1[num1]);
		num1--;
		download1(1);
		ans=Q[1];
		}
		Q1[++num1]=ans;
		update1(num1);
}
```

代码

```
if(op==3){

int ans=Q[1];
while(num1>0 && ans==Q1[1]){
                                //printf("%d %d\n",ans,Q1[1]);
        swap(&Q[1],&Q[num]);
        num--;
        download(1);

        swap(&Q1[1],&Q1[num1]);
        num1--;
        download1(1);

        ans=Q[1];
   }
   printf("%d\n",ans);
}
if(op==4){
    scanf("%d",&x);
    Q1[++num1]=x;
    update1(num1);
}
```