# Splay的基本操作（模板）

# 【bzoj3224】普通平衡树

## **Description**

您需要写一种数据结构（可参考题目标题），来维护一些数，其中需要提供以下操作：  
1. 插入x数  
2. 删除x数(若有多个相同的数，因只删除一个)  
3. 查询x数的排名(若有多个相同的数，因输出最小的排名)  
4. 查询排名为x的数  
5. 求x的前驱(前驱定义为小于x，且最大的数)  
6. 求x的后继(后继定义为大于x，且最小的数)

## **Sample Input**

10  
1 106465  
4 1  
1 317721  
1 460929  
1 644985  
1 84185  
1 89851  
6 81968  
1 492737  
5 493598

## **Sample Output**

106465  
84185  
492737

## **HINT**

1.n的数据范围：n<=100000

2.每个数的数据范围：[-2e9,2e9]

## **Source**

[平衡树](http://www.lydsy.com/JudgeOnline/problemset.php?search=%E5%B9%B3%E8%A1%A1%E6%A0%91" \t "https://blog.csdn.net/clove_unique/article/details/_blank)

【题解】

【代码】

#include<iostream>

#include<cstring>

#include<cstdio>

using namespace std;

#define MAXN 1000000

int ch[MAXN][2],f[MAXN],size[MAXN],cnt[MAXN],key[MAXN];

int sz,root;

inline void clear(int x){

ch[x][0]=ch[x][1]=f[x]=size[x]=cnt[x]=key[x]=0;

}

inline bool get(int x){

return ch[f[x]][1]==x;

}

inline void update(int x){

if (x){

size[x]=cnt[x];

if (ch[x][0]) size[x]+=size[ch[x][0]];

if (ch[x][1]) size[x]+=size[ch[x][1]];

}

}

inline void rotate(int x){

int old=f[x],oldf=f[old],whichx=get(x);

ch[old][whichx]=ch[x][whichx^1]; f[ch[old][whichx]]=old;

ch[x][whichx^1]=old; f[old]=x;

f[x]=oldf;

if (oldf)

ch[oldf][ch[oldf][1]==old]=x;

update(old); update(x);

}

inline void splay(int x){

for (int fa;fa=f[x];rotate(x))

if (f[fa])

rotate((get(x)==get(fa))?fa:x);

root=x;

}

inline void insert(int x){

if (root==0){sz++; ch[sz][0]=ch[sz][1]=f[sz]=0; root=sz; size[sz]=cnt[sz]=1; key[sz]=x; return;}

int now=root,fa=0;

while(1){

if (x==key[now]){

cnt[now]++; update(now); update(fa); splay(now); break;

}

fa=now;

now=ch[now][key[now]<x];

if (now==0){

sz++;

ch[sz][0]=ch[sz][1]=0;

f[sz]=fa;

size[sz]=cnt[sz]=1;

ch[fa][key[fa]<x]=sz;

key[sz]=x;

update(fa);

splay(sz);

break;

}

}

}

inline int find(int x){

int now=root,ans=0;

while(1){

if (x<key[now])

now=ch[now][0];

else{

ans+=(ch[now][0]?size[ch[now][0]]:0);

if (x==key[now]){

splay(now); return ans+1;

}

ans+=cnt[now];

now=ch[now][1];

}

}

}

inline int findx(int x){

int now=root;

while(1){

if (ch[now][0]&&x<=size[ch[now][0]])

now=ch[now][0];

else{

int temp=(ch[now][0]?size[ch[now][0]]:0)+cnt[now];

if (x<=temp) return key[now];

x-=temp; now=ch[now][1];

}

}

}

inline int pre(){

int now=ch[root][0];

while (ch[now][1]) now=ch[now][1];

return now;

}

inline int next(){

int now=ch[root][1];

while (ch[now][0]) now=ch[now][0];

return now;

}

inline void del(int x){

int whatever=find(x);

if (cnt[root]>1){cnt[root]--; update(root); return;}

if (!ch[root][0]&&!ch[root][1]) {clear(root); root=0; return;}

if (!ch[root][0]){

int oldroot=root; root=ch[root][1]; f[root]=0; clear(oldroot); return;

}

else if (!ch[root][1]){

int oldroot=root; root=ch[root][0]; f[root]=0; clear(oldroot); return;

}

int leftbig=pre(),oldroot=root;

splay(leftbig);

ch[root][1]=ch[oldroot][1];

f[ch[oldroot][1]]=root;

clear(oldroot);

update(root);

}

int main(){

int n,opt,x;

scanf("%d",&n);

for (int i=1;i<=n;++i){

scanf("%d%d",&opt,&x);

switch(opt){

case 1:insert(x); break;//插入

case 2:del(x); break;//删除

case 3:printf("%d\n",find(x)); break;//查询数据x的排名

case 4:printf("%d\n",findx(x)); break;//查询第x名的数据

case 5:insert(x);printf("%d\n",key[pre()]);del(x);break;//求x的前驱(前驱定义为小于x，且最大的数)

case 6:insert(x);printf("%d\n",key[next()]);del(x);break;//求x的后继(后继定义为大于x，且最小的数)

}

}

}