/\*

#include<iostream>

#include<cstring>

#define maxn 100

using namespace std;

char a[maxn];

/\*

int Search(char A[],char value,int n)

{

int i;

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

{

if(A[i]==value)

{

return i;

}

}

return -1;

}\*/

/\*

int BinarySearch(char A[],char value,int left,int right)

{

int i,j;

while(left<=right)

{

int mid=left+(right-left)/2;

if(A[mid]==value)

{

return mid;

}

if(A[mid]>value)

{

right=mid-1;

}

if(A[mid]<value)

{

left=mid+1;

}

}

return -1;

}

int main()

{

int len;

char value;

cin>>a;

cin>>value;

len = strlen(a);

cout<<BinarySearch(a,value,0,len-1)<<endl;

} \*/

/\*

#include<iostream>

#include<cstring>

#define maxn 20

using namespace std;

char a[maxn];

int main()

{

int test,year,month,day;

cin>>test;

while(test>0)

{

cin>>a;

int len=strlen(a);

int i=0,j;

while(i<len)

{

if(a[i]==':')

{

for(j=i;j<len;j++)

{

a[j]=a[j+1];

}

}

else

{

i++;

}

}

int

test--;

}

}

\*/

/\*

#include<iostream>

#define maxn 100

int du[maxn];

int result[maxn];

using namespace std;

int main()

{

int test,n,k=0,i,a,b;

cin>>test;

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

{

du[i]=0;

}

while(test>0)

{

cin>>n;

for(i=0;i<n-1;i++)

{

cin>>a>>b;

du[a]++;

du[b]++;

}

int sum=0;

for(i=0;i<n-1;i++)

{

if(i==0)

{

if(du[i]>=du[i+1])

{

sum++;

}

}

else if(i==n-2)

{

if(du[i]>=du[i-1])

{

sum++;

}

}

else

{

if(du[i]>=du[i-1]&&du[i]>=du[i+1])

{

sum++;

}

}

}

result[k]=sum;

k++;

test--;

}

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

{

cout<<result[i]<<endl;

}

}

\*/

/\*

//是否为中序

#include<iostream>

#define maxn 100000

int a[maxn];

int result[maxn];

using namespace std;

int main()

{

int test,n,k=0,i,j;

cin>>test;

while(test>0)

{

cin>>n;

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

{

cin>>a[i];

}

int flag=0;

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

{

for(j=i;j<n;j++)

{

if(a[i]>a[j])

{

flag=1;

break;

}

}

}

result[k]=flag;

k++;

test--;

}

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

{

if(result[i]==0)

{

cout<<"yes"<<endl;

}

else

{

cout<<"no"<<endl;

}

}

}

\*/

/\*

//最小距离查询

#include<iostream>

#include<cstring>

#include<cmath>

#define maxn 100000

#define maxo 15

char s[maxn];

char op[maxo];

int result[maxn];

using namespace std;

int main()

{

int test,m,i,j,k=0;

char insert;

cin>>test;

while(test>0)

{

cin>>s;

int len=strlen(s);

cin>>m;

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

{

cin>>op>>insert;

if(strcmp(op,"INSERT")==0)

{

s[len]=insert;

len++;

}

if(strcmp(op,"QUERY")==0)

{

int mid = int(insert)-48;

//char midV = s[mid];

int min=len;

int flag=0;

//cout<<s[len-1];

for(j=0;j<len;j++)

{

if(s[j]==s[mid]&&abs(j-mid)<min&&j!=mid)

{

min=abs(j-mid);

flag=1;

}

}

if(flag==1)

{

result[k]=min;

k++;

}

else

{

result[k]=-1;

k++;

}

}

}

test--;

}

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

{

cout<<result[i]<<endl;

}

}

\*/

//二叉排序树

#include<iostream>

#define maxn 100

int result[maxn]={-1};

int a[maxn];

int k=1;

using namespace std;

struct node

{

node \*lchild;

node \*rchild;

int val;

}\*root;

void dfs(node \*p,int tmp)

{

if(tmp<p->val)

{

if(p->lchild)

{

dfs(p->lchild,tmp);

}

else

{

node \*t =new node;

t->lchild=NULL;

t->rchild=NULL;

t->val=tmp;

p->lchild=t;

result[k]=p->val;

k++;

}

}

if(tmp>p->val)

{

if(p->rchild)

{

dfs(p->rchild,tmp);

}

else

{

node \*t = new node;

t->lchild=NULL;

t->rchild=NULL;

t->val=tmp;

p->rchild=t;

result[k]=p->val;

k++;

}

}

}

int main()

{

int test,n,i;

cin>>n;

cin>>a[0];

node \*t=new node;

t->lchild=NULL;

t->rchild=NULL;

t->val=a[0];

root=t;

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

{

cin>>a[i];

dfs(root,a[i]);

}

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

{

cout<<result[i]<<endl;

}

}

/\*

#include<iostream>

#include<cstring>

#define maxn 100

using namespace std;

char a[maxn];

int b[maxn];

int result[maxn];

int main()

{

int i,l=0,k=0,j,g=0;

cin>>a;

int len=strlen(a);

cout<<a[0];

/\*

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

{

if(a[i]>0)

{

b[l]=g;

l++;

}

}

for(i=0;i<l-1;i++)

{

int sum=0;

for(j=b[i];j<=b[i+1];j++)

{

sum+=a[j];

}

result[k]=sum;

k++;

}

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

{

cout<<result[i]<<endl;

}\*/

#include<iostream>

using namespace std;