DP

1.最长公共子序列

```
#include<iostream>
#include<cstdio>
using namespace std;
int a[100001],b[100001],map[100001],f[100001];
int main()
{
        int n;
        cin>>n;
        for(int i=1;i≤n;i++){scanf("%d",&a[i]);map[a[i]]=i;}
        for(int i=1;i≤n;i++){scanf("%d",&b[i]);f[i]=0x7fffffff;}
        int len=0;
        f[0]=0;
        for(int i=1;i≤n;i++)
        {
                int l=0,r=len,mid;
                if(map[b[i]]>f[len])f[++len]=map[b[i]];
                else
                {
                while(l<r)
                {
                    mid=(l+r)/2;
                    if(f[mid]>map[b[i]])r=mid;
                        else l=mid+1;
                }
                f[l]=min(map[b[i]],f[l]);
        }
    }
    cout<<len;
    return 0
}```
# 2.最长(不)上升子序列
#### 树状数组优化
```cpp
#include<iostream>
#include<stdio.h>
#include<string.h>
#include<algorithm>
#include<queue>
#include<math.h>
#include<vector>
#define ls (p<<1)
#define rs (p<<1|1)
```
```

```cpp
#define mid (l+r)/2
#define over(i,s,t) for(register long long i=s;i≤t;++i)
#define lver(i,t,s) for(register long long i=t;i≥s;--i)
//#define int __int128
using namespace std;
typedef long long ll;//全用ll可能会MLE或者直接WA,试着改成int看会不会A
const ll N=100007;
const ll INF=1e10+9;
const ll mod=21474836647;
const double EPS=1e-10;//-10次方约等于趋近为0
const double Pi=3.14159265358897;

ll n,m,ans1,ans2,tree[N],a[N],maxn;
inline void update(ll k,ll val)//更新
{
    while(k≤maxn)//要注意这里是maxn也就是a数组里的最大值来作为树状数组的上限
    {
        tree[k]=max(tree[k],val);//维护最大值
        k+=k&(-k);
    }
}
inline ll query(ll k)
{
    ll res=0;
    while(k)
    {
        res=max(res,tree[k]);//求以小于等于x的数为结尾的最长不上升子序列的长度的最大值
        k-=k&(-k);
    }
    return res;
}
int main()
{
    while(scanf("%lld",&a[++n])≠EOF)maxn=max(maxn,a[n]);
    n--;
    for(int i=n;i≥1;--i)
    {
        ll q=query(a[i])+1;
        /*相当于朴素做法的：for(int j=i+1;j≤n;j++) if(a[j]≤a[i]) 因为是按照顺序从后往前循环，所以当前的i所query到的所有的值都是在i后面的， 解决了i<j的问题，树状数组维护最大值，就解决了a[j]≤a[i]的问题， 所以这里求的就是以i为开头的最长不上升子序列的长度*/
        update(a[i],q);//这个最大值+1就是以当前这个数开头的最长不上升子序列的长度，丢到树状数组里面去更新后面的值
        ans1=max(ans1,q);
    }
    memset(tree,0,sizeof tree);
    for(int i=1;i≤n;++i)
    {
```

```
        ll q=query(a[i]-1)+1;
        /*查询以小于（没有等于！！！）x的数为结尾的最长上升子序列的长度的最大值 因为不能等于所以要-1*/
        update(a[i],q);//这个最大值+1就是以当前这个数结尾的最长上升子序列的长度，丢到树状数组里面去
        ans2=max(ans2,q);
    }
    printf("%lld\n%lld\n",ans1,ans2);
    return 0;

}
```

二分优化

```
#include<iostream>
#include<cstdio>
#include<algorithm>
#define R register
using namespace std;
const int N=100010;
int a[N],d1[N],d2[N],n;
inline bool read(int &x) {
    char c=getchar();
    if(c==EOF)return false;
    while(c>'9'||c<'0')c=getchar();
    while(c≥'0'&&c≤'9') {
        x=(x<<1)+(x<<3)+(c^48);
        c=getchar();
    }
    return true;
}
int main() {
    while(read(a[++n]));n--;
    R int len1=1,len2=1;
    d1[1]=d2[1]=a[1];
    for(R int i=2; i≤n; i++) {
        if(d1[len1]≥a[i])d1[++len1]=a[i];
        else *upper_bound(d1+1,d1+1+len1,a[i],greater<int>())=a[i];
        if(d2[len2]<a[i])d2[++len2]=a[i];
        else *lower_bound(d2+1,d2+1+len2,a[i])=a[i];
    }
    printf("%d\n%d",len1,len2);
    return 0;
}
```