R1-排序和二分查找-题解

A 排序

也可以用冒泡排序

如果用qsort的话cmp函数里面不要直接相减返回，会溢出，比较大小然后返回1，-1。

```
#include<stdio.h>
#include<stdlib.h>
int a[1005],n;
int cmp(const void *x,const void *y){
    return *(int *)x>*(int *)y?-1:1;
}
int main(){
    int i;
    scanf("%d",&n);
    for(i=1;i<=n;i++)scanf("%d",&a[i]);
    qsort(a+1,n,sizeof(a[0]),cmp);
    for(i=1;i<=n;i++)printf("%d ",a[i]);
    return 0;
}
```

B 字典序排序（简单）

每次读一个字符串，然后对字符数组中的字符进行排序

```
#include <math.h>
#include <stdio.h>
#include <string.h>

int main()
{
    char a[100] = {0};
    int i, j, hold;
    while (scanf("%s", a) != EOF)
    {
        for (i = 0; i < strlen(a); i++)
        {
            for (j = 0; j < strlen(a) - 1 - i; j++)
            {
                if (a[j] > a[j + 1])
                {
                    hold = a[j];
                    a[j] = a[j + 1];
                    a[j + 1] = hold;
                }
            }
        }
        printf("%s\n", a);
    }

    return 0;
}
```

C 简单的二分法

注意是多组数据输入，下界二分查找

```
#include <stdio.h>
int a[1000002];
int find(int num, int length)
{
    int left = 1, right = length;
    while (left < right)
    {
        int mid = left + (right - left) / 2;
        if (a[mid] >= num)
            right = mid;
        else
            left = mid + 1;
    }
    if (a[left] == num)
        return left;
    else
        return -1;
}
int main()
{
    int n, m, q;
    while (~scanf("%d %d", &n, &m))
    {
        for (int i = 1; i <= n; i++)
            scanf("%d", &a[i]);
        while (m--)
        {
            scanf("%d", &q);
            int des = find(q, n);
            if (des == -1)
                puts("error");
            else
                printf("%d\n", des);
        }
    }
}
```

D 填报高考志愿

排序+下界二分查找的综合问题。

找到了第一个大于等于估分成绩的数之后，那么前一个就是最大的小于估分成绩的数，比较他俩中满意度最小的那个就行了。

注意数据范围，以及第一个大于等于估分成绩的数位最开头的数的边界情况

需要用到qsort

```
#include <stdio.h>
#include <stdlib.h>
int cmp(const void *p1,const void *p2)
{
    return *(int *)p1 < *(int *)p2?-1:1;
}
int lower_bound(int a[],int n,int val)
{
    int l=0,r=n,mid;
    while(l<r)
    {
        mid=(l+r)>>1;
        if(a[mid]>=val)
        r=mid;
        else
        l=mid+1;
    }
    return r;
}
int m,n;
int a[100005],b[100005];
long long ans;
int main()
{
    scanf("%d%d",&m,&n);
    for(int i=0;i<m;i++)scanf("%d",&a[i]);
    qsort(a,m,sizeof(a[0]),cmp);
    for(int i=0;i<n;i++)
    {
        int key,index;
        scanf("%d",&key);
        index=lower_bound(a,m,key);//找到第一个大于等于key的下标index
        if(index==0)ans+=abs(a[index]-key);//考虑是第一个数的情况
        else
        {
            int l1=abs(a[index]-key);//否则比较index和index-1这两个数
            int l2=abs(a[index-1]-key);
            if(l1<l2)ans+=l1;
            else ans+=l2;
        }
    }
    printf("%lld",ans);
}
```

E ModricWang拦截导弹

最长不上升子序列问题

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <time.h>

int a[1002], f[1002];
// f[i]表示以a[i]为结尾的最长下降序列的长度
// 所以, f[i]=max(f[j]+1),1<=j<i, 当a[i]>=a[j]时
int main()
{
    int n, i, j, ans;
    scanf("%d", &n);
    for (i = 1; i <= n; i++)
    {
        scanf("%d", &a[i]);
        f[i] = 1;
    }
    for (i = 1; i <= n; i++)
    {
        for (j = 1; j < i; j++)
            if (a[j] >= a[i] && f[j] + 1 >= f[i])
                f[i] = f[j] + 1;
    }
    for (i = 1; i <= n; i++)
        if (f[i] > ans)
            ans = f[i];
    printf("%d\n", ans);
    return 0;
}
```

F ModricWang的空间折跃

排序之后找不同元素的个数即可

```
#include <stdio.h>
#include <string.h>
#include <math.h>
```

```c
#include <ctype.h>
#include <stdlib.h>
#define ll long long
#define eps 1e-10

int cmp(const void *a, const void *b)
{
    if (*(ll *)a > *(ll *)b)
        return 1;
    else
        return -1;
}

int main(void)
{
    ll height[15000];
    int n;
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
    {
        scanf("%lld", &height[i]);
    }

    qsort(height, n, sizeof(height[0]), cmp);

    ll tmp = height[0];
    int count = 1;
    for (int i = 1; i < n; i++)
    {
        if (height[i] > tmp)
        {
            tmp = height[i];
            count++;
        }
    }
    printf("%d", count);

    return 0;
}
```

G 紧急救援

所有可能性排好序之后二分查找就行了

```c
#include <stdio.h>
#include <math.h>
#include <ctype.h>
#include <string.h>

```

```c
#include <stdlib.h>
#define ll long long
#define un unsigned
#define eps 1e-13
int cmp(const void *a, const void *b)
{
    if (*(int *)a > *(int *)b)
        return 1;
    else
        return -1;
}
ll rec_bin_find(ll b[], ll key, ll low, ll high)
{
    ll m;
    while (low < high)
    {
        m = (low + high) >> 1;
        if (b[m] >= key)
            high = m;
        else
            low = m + 1;
    }
    return low;
}
ll a[2022], b[2022], c[4088484];
int main()
{
    ll k = 0;
    ll x;
    ll n, m;
    scanf("%lld%lld", &n, &m);
    for (int i = 0; i < n; i++)
        scanf("%lld", &a[i]);
    for (int i = 0; i < n; i++)
        scanf("%lld", &b[i]);
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            c[k] = a[i] + b[j];
            k++;
        }
    }
    qsort(c, k, sizeof(c[0]), cmp);
    for (int i = 1; i <= m; i++)
    {
        scanf("%lld", &x);
        ll judge = rec_bin_find(c, x, 0, k);
        printf("%lld\n", k - judge);
    }
    return 0;
}
```