

AtCoder Grand Contest 054

[比赛链接](#)

B - Greedy Division

题意

给出 n 个物品，第 i 个物品权重为 w_i 。现有甲乙两人，要求将物品重新排列后依次分配每个物品。

分配物品满足如下规则：如果甲当前所得的物品权重和不大于乙，则将物品分给甲，否则分给乙。

问使得甲乙两人最后得所得物品权重和相等的排列方案共有多少种。

解法

假定甲得到物品的顺序依次为 $x_1, x_2 \dots x_k$ 乙得到物品的顺序依次为 $y_1, y_2 \dots y_{n-k}$

已知当序列 x, y 固定时物品的总排列也是固定的。考虑背包求出不考虑 $x_1, x_2 \dots x_k$ 顺序时甲所得物品权重为 $\frac{\sum w_i}{2}$ 的方案数 g_k

则答案为 $\sum_{k=1}^n k!(n-k)!g_k$ 设 $\text{dp}(i, j, k)$ 表示仅考虑前 i 个物品，甲所得物品总和为 j 已经选中 k 个物品的方案数，不难得到状态转移。

时间复杂度 $O(n^2 \sum w_i)$

```
const int MAXN=105,MAXV=1e4+5,mod=998244353;
int w[MAXN],f[MAXN],dp[2][MAXV][MAXN];
int main()
{
    int n=read_int();
    _for(i,0,n)w[i]=read_int();
    int pos=0;
    dp[pos][0][0]=1;
    _for(i,0,n){
        pos^=1;
        _for(j,0,MAXV)_rep(k,0,n)
            dp[pos][j][k]=dp[!pos][j][k];
        _for(j,w[i],MAXV)_rep(k,1,n)
            dp[pos][j][k]=(dp[pos][j][k]+dp[!pos][j-w[i]][k-1])%mod;
    }
    f[0]=1;
    _rep(i,1,n)f[i]=1LL*f[i-1]*i%mod;
    int s=0;
    _for(i,0,n)
        s+=w[i];
    if(s%2==1)
```

```
puts("0");
else{
    s/=2;
    int p=1,ans=0;
    _rep(i,1,n)
        ans=(ans+1LL*dp[pos][s][i]*f[i]%mod*f[n-i])%mod;
    enter(ans);
}
return 0;
}
```

C - Roughly Sorted

题意

给定一个 $1 \sim n$ 的排序 P 每次操作可以交换相邻两个元素的位置。

要求用最少的操作使得对每个 i 最多有 k 个 j 满足 $P_j > P_i$

现已知 P 问存在多少个可能的最初的 P

解法

记 $f_i = \sum_{j=1}^{i-1} [P_j > P_i]$ 不难发现对每次操作最多可以使得某个 f_i 减一，于是最少操作数的下限为 $\sum_{i=1}^n \max(f_i - k, 0)$

考虑找到最小的 i 使得 $f_i \leq k$ 不难发现此时一定有 $P_{i-1} > P_i$ 否则一定有 $f_{i-1} = f_i > k$ 与 i 的最小性矛盾。

交换 P_{i-1}, P_i 可以使得 f_i 减一。于是可以证明 $\sum_{i=1}^n \max(f_i - k, 0)$ 这个操作下限是可以取到的。

根据离散数学知识，知逆序数与排列存在一一对应关系，同时有 $f(i) \leq n - P_i$

同时，为了使得步数最少，对 $f(i) < k$ 的位置，他的初值应该不变。对 $f(i) = k$ 的位置，他的初值只需要满足 $f(i) \leq n - P_i$

于是不难发现答案为 $\prod_{f(i)=k} (n - P_i - f(i) + 1)$ 时间复杂度 $O(n \log n)$

```
const int MAXN=5e3+5,mod=998244353;
int c[MAXN];
#define lowbit(x) ((x)&(-x))
void add(int pos){
    while(pos<MAXN){
        c[pos]++;
        pos+=lowbit(pos);
    }
}
```

```
    }  
}  
int query(int pos){  
    int ans=0;  
    while(pos){  
        ans+=c[pos];  
        pos-=lowbit(pos);  
    }  
    return ans;  
}  
int a[MAXN],p[MAXN];  
int main()  
{  
    int n=read_int(),k=read_int(),ans=1;  
    _for(i,0,n){  
        a[i]=read_int();  
        p[i]=i-query(a[i]);  
        add(a[i]);  
        if(p[i]==k)  
            ans=1LL*ans*(n-a[i]-p[i]+1)%mod;  
    }  
    enter(ans);  
    return 0;  
}
```

From:

<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:

https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:contest:agc_054



Last update: **2021/06/28 10:33**