

Atcoder Rugular Contest 121

[比赛链接](#)

D - 1 or 2

题意

给定 n 个数，要求将它们分组。每组可以有 $1 \sim 2$ 个数，每组的权重为这组里面的所有数的和。

要求最小化权重最大的组和权重最小的组的权重差。

题解

首先考虑如果强制每组必须有两个数，则对任意两组 $(a_1, a_2), (a_3, a_4)$ 设 a_1 是这四个数中的最小值，则必有 a_2 是这四个数中的最大值。

否则假定 a_4 是这四个数中的最大值，考虑分组 $(a_1, a_4), (a_2, a_3)$

则有 $\max(a_1+a_4, a_2+a_3) \leq a_3+a_4, \min(a_1+a_4, a_2+a_3) \geq a_1+a_2$ 显然更优。

于是设所有数为 $a_1 \leq a_2 \leq \dots \leq a_{2k}$ 则最优分组为 $(a_1, a_{2k}), (a_2, a_{2k-1}), \dots, (a_k, a_{k+1})$

一个数一组相当于这个数和 0 一组。于是可以枚举向原序列中加入 $0 \sim n$ 个 0 的情况，然后按上述方法分组计算答案。

时间复杂度 $O(n^2)$

```
const int MAXN=5e3+5,inf=2e9;
int a[MAXN<<1];
int main()
{
    int n=read_int();
    for(i,0,n)a[i]=read_int();
    sort(a,a+n);
    int ans=inf;
    rep(i,n,n*2){
        if(i%2==0){
            int maxv=-inf,minv=inf;
            for(int j=0;j<i/2;j++){
                maxv=max(maxv,a[j]+a[i-1-j]);
                minv=min(minv,a[j]+a[i-1-j]);
            }
            ans=min(ans,maxv-minv);
        }
    }
}
```

```
    int pos=0;
    while(pos<i&&a[pos]<=0) pos++;
    for(int j=i-1;j>=pos;j--)
        a[j+1]=a[j];
    a[pos]=0;
}
enter(ans);
return 0;
}
```

E - Directed Tree

题意

给定一棵 n 个结点的树，问有多少个 $\sim n$ 的排列 P 满足 p_i 不是结点 i 在树上的祖先结点（不包括自己）。

题解

设 $dp(u,i)$ 表示结点 u 为根的子树中钦定 i 个结点的 p_i 是其祖先结点且该祖先结点位于结点 u 为根的子树的方案数。

dp 要求祖先结点必须位于结点 u 为根的子树是为了防止子树背包合并时的冲突，另外不考虑非钦定结点的 p_i 对方案的贡献，留在最后处理。

首先进行树形背包合并，得到不使用 u 的方案数。接下来考虑用 u 来钦定一个结点，得到状态转移

```
$$ dp(u,i) \leftarrow \sum_{j=0}^{n-i} dp(u,j) * dp(u,i-j) $$
```

根据容斥定理，最终答案为

```
$$ \sum_{i=0}^n (-1)^i (n-i)! dp(0,i) $$
```

时间复杂度 $O(n^2)$

```
const int MAXN=2005,Mod=998244353;
struct Edge{
    int to,next;
}edge[MAXN<<1];
int head[MAXN],edge_cnt;
void Insert(int u,int v){
    edge[++edge_cnt]=Edge{v,head[u]};
    head[u]=edge_cnt;
}
int dp[MAXN][MAXN],temp[MAXN],sz[MAXN],frac[MAXN];
void dfs(int u){
    dp[u][0]=sz[u]=1;
```

```
for(int i=head[u];i;i=edge[i].next){
    int v=edge[i].to;
    dfs(v);
    _rep(j,0,sz[u])_rep(k,0,sz[v])
    temp[j+k]=(temp[j+k]+1LL*dp[u][j]*dp[v][k])%Mod;
    sz[u]+=sz[v];
    _rep(j,0,sz[u]){
        dp[u][j]=temp[j];
        temp[j]=0;
    }
}
for(int i=sz[u]-1;i>=0;i--)
dp[u][i+1]=(dp[u][i+1]+1LL*dp[u][i]*(sz[u]-i-1))%Mod;
}
int main()
{
    int n=read_int();
    _rep(i,2,n)
    Insert(read_int(),i);
    dfs(1);
    int ans=0;
    frac[0]=1;
    _for(i,1,MAXN)
    frac[i]=1LL*frac[i-1]*i%Mod;
    _rep(i,0,n){
        if(i&1)
            ans=(ans-1LL*dp[1][i]*frac[n-i])%Mod;
        else
            ans=(ans+1LL*dp[1][i]*frac[n-i])%Mod;
    }
    ans=(ans+Mod)%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:arc_121&rev=1625130899

Last update: 2021/07/01 17:14

