codeforces1392部分题解

1392E

题意

交互题，给出一个$n\times n$的地图，一个人从$(1,1)$走到(n,n)只能往右或者往下走，现在你可以给每个格子赋值，有q组询问，每组询问给出路程权值和，问走过的路径。

题解

因为n很小，可以考虑二进制构造地图，同一行相邻成2，同一列相邻乘4即可。那么对于一个路程权值和，如果二进制是一段连续的1，那么他现在在向右走，如果出现了0就向下走。

代码

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
ll a[30][30];
int main() {
    int n,q;
    scanf("%d",&n);
    for (int i = 1;i<= n;i++)
        for (int j = 1;j<= n;j++)
        {
            if (i&1)printf("0%c",j==n?'\n':' ');
            else printf("%lld%c",1ll << (i+j-3),j==n?'\n':' ');
            fflush(stdout);
        }
    scanf("%d",&q);
    ll k;
    while (q--) {
        scanf("%lld",&k);
        printf("1 1\n");
        int x,y;
        x = 1;y = 1;
        fflush(stdout);
        for (int i = 0;i <= 2*n-3;i++)
        {
            if (k & (1ll<<i)) {
                if (x&1)x++;
                else y++;
            } else {
                if (x&1)y++;
                else x++;
```

```
            }
            printf("%d %d\n",x,y);
            fflush(stdout);
        }
    }
    return 0;
}
```

1392F

题意

给一个单调递增的数组，如果相邻两个元素$a_i<a_{i+1}+1$那么就令a_i加一，令a_{i+1}减一。问最后状态如何。

题解

可以确定的是最后的状态一定是相邻相差一，最多有一对相邻是相等的值。于是我们只需要求和然后直接模拟即可。

代码

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const int N = 1e6+5;
ll h[N];
struct Node {
    ll x,y;
};
vector<Node> vec;
int main() {
    int n;
    scanf("%d",&n);
    for (int i = 1;i<= n;i++)
    {
        scanf("%lld",&h[i]);
        h[i]-=i;
    }
    vec.push_back({h[1],1});
    for (int i = 2;i<= n;i++) {
        if (h[i] < vec.back().x) {
            vec.push_back({h[i],i});
            continue;
```

```
        }
        if (h[i] == vec.back().x) continue;
        while (vec.size() > 1 && h[i]-vec.back().x > i - vec.back().y) {
            h[i]-=i-vec.back().y;
            vec.pop_back();
        }
        if (vec.size() == 1) {
            ll tmp = (h[i] - vec.back().x) / i;
            h[i] -= tmp*(i-1);
            vec.back().x += tmp;
        }
        if (h[i] == vec.back().x) continue;
        ll d = h[i] - vec.back().x;
        Node tmpk = vec.back();
        if (vec.size() == 1)
            vec.back().x++;
        else
            vec.pop_back();
        vec.push_back({tmpk.x,tmpk.y + d});
    }
    vec.push_back({0,n*10});
    int p = 0;
    for (int i = 1;i<= n;i++) {
        if (vec[p+1].y == i)p++;
        printf("%lld ",vec[p].x+i);
    }
    printf("\n");
    return 0;
}
```

1392G

题意

有k个位置，每个位置可以放0或1，有n个人，第i个人可以把a_i位置和b_i位置交换一次。给出初始位置的01和最终位置的01，要选一段人使得操作后位置相同个数最大。

题解

考虑这个序列，其实交换(l,r)相对于对初始序列的$(1,l-1)$和最终序列的$(1,r)$做反向交换然后互相比较。预处理好然后计算更新答案就行。

代码

```
#include <bits/stdc++.h>
```

```
using namespace std;
typedef long long ll;
const int N = 2e6+5;
const int inf = 1e9;
int a[N],b[N],p[30],L[N],R[N];
string ss,tt,s[N],t[N];
int calc(string str) {
    int ans = 0;
    for (int i = 0;i <= str.size()-1;i++)
        if (str[i]=='1')ans |= (1 << i);
    return ans;
}
int getone(int x) {
    int cnt = 0;
    while (x) {
        if (x&1)cnt++;
        x>>=1;
    }
    return cnt;
}
int main() {
    int n,m,k;
    scanf("%d%d%d",&n,&m,&k);
    cin >> ss >> tt;
    s[0] = ss;
    t[0] = tt;
    for (int i = 1;i<= n;i++)
        scanf("%d%d",&a[i],&b[i]),a[i]--,b[i]--;
    for (int i = 0;i<= k;i++)p[i] = i;
    for (int i = 0;i < (1<< k);i++)
        L[i] = inf,R[i] = -inf;
    for (int i = 1;i<= n;i++) {
        s[i] = t[i] = string(k,'0');
        swap(p[a[i]],p[b[i]]);
        for (int j = 0;j < k;j++) {
            s[i][p[j]] = ss[j];
            t[i][p[j]] = tt[j];
        }
    }
    for (int i = 0;i<= n;i++) {
        L[calc(s[i])] = min(L[calc(s[i])],i);
        R[calc(t[i])] = max(R[calc(t[i])],i);
    }
    for (int i = (1<<k)-1;i >= 0;i--)
        for (int j = 0;j < k;j++)
            if ((1<<j)&i) {
                L[i^(1<<j)] = min(L[i^(1<<j)],L[i]);
                R[i^(1<<j)] = max(R[i^(1<<j)],R[i]);
            }
    int ans = 0,l = 1,r = 1;
```

```
    for (int i = 0;i < (1<<k);i++)
        if (R[i]-L[i]>=m && getone(i)>ans) {
            ans = getone(i);
            l = L[i]+1,r = R[i];
        }
    printf("%d \n%d %d",k+2*ans-count(ss.begin(),ss.end(),'1')-
count(tt.begin(),tt.end(),'1'),l,r);
    return 0;
}
```

1392H

题意

给你n张数字牌和$m-n$张特殊牌河一个数字集合。进行如下操作：

拿出最上面的那张卡片，如果卡片是数字牌，那么把这个数字放进集合。

其他情况下把所有卡片重新打乱。检验集合是否包含了所有数字，如果包含了结束游戏。

问期望操作次数。

题解

期望操作次数是期望游戏轮数乘以期望每一轮次数。

可以发现如果在游戏中摸到特殊牌那么一轮就会结束，那么我们算出所有排列中第一张特殊牌之前的牌数，然后+1除以排列数目就是每轮游戏操作次数的期望。这个的答案是$\frac{n}{m+1}+1$

然后另一个部分则是$m\sum_{i=1}^n\frac{1}{i}+1$最终答案就是这两者相乘。

代码

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const int mod = 998244353;
ll quick_pow(ll x,ll y) {
    ll ans = 1;
    while (y) {
        if (y&1) ans = ans*x%mod;
        x = x*x%mod;
        y >>= 1;
    }
    return ans;
}
```

```
int main() {
    int n,m;
    scanf("%d%d",&n,&m);
    ll ans1 = (1ll*n*quick_pow(m+1,mod-2)%mod+1)%mod;
    ll ans2 = 0;
    for (int i = 1;i<= n;i++)
        ans2 = (ans2+quick_pow(i,mod-2))%mod;
    ans2 = (ans2*m+1)%mod;
    printf("%lld\n",ans1*ans2%mod);
    return 0;
}
```