2020牛客暑期多校训练营（第四场）

比赛情况

题号	A	B	C	D	E	F	G	H	I	J	K	L	M
状态	-	O	O	-	-	O	-	O	Ø	-	-	-	-

O 在比赛中通过 Ø 赛后通过 ! 尝试了但是失败了 - 没有尝试

比赛时间

2020-07-20 12:00-17:00

提交记录

题解

B - Basic Gcd Problem

题意：当$x>1$时，$f_c(x)=max_{i=1...x-1}c·f_c(gcd(i,x))$□当$x=1$时，$f_c(x)=1$□给出若干$n,c$□求$f_c(n)$

题解□n的质因数的数目为cnt_n□可以分析出来问题的答案是c^{cnt_n}□硬分解可能会T□写个递推求质因数数目就行。

```
#include <bits/stdc++.h>
using namespace std;
#define maxn 1000005
using namespace std;
typedef long long ll;
const int mod = 1e9+7;
bool isprime[maxn];
int prime[maxn];
int f[maxn];
int cnt=0;
void PJ(){
    memset(isprime,0,sizeof(isprime));
    cnt=0;
    for(int i=2;i<=1000000;i++){
        if(!isprime[i]){
            prime[cnt++]=i;
        }
        for(int j=0;j<cnt&&prime[j]*i<=maxn;j++){
            isprime[prime[j]*i]=1;
            if(i%prime[j]==0) break;
        }
    }
}
```

```
ll quick_pow(ll x,int y) {
    ll ans = 1;
    while(y) {
        if (y&1)
            ans = (ans*x)%mod;
        x = x*x%mod;
        y >>= 1;
    }
    return ans;
}
int main()
{
    PJ();
    for (int i = 1;i<= 1000000;i++)
        for (int j = 0;j<cnt&&(ll)i*prime[j]<=1000000ll;j++)
            f[i*prime[j]] = f[i]+1;
    int cas;
    scanf("%d",&cas);
    while (cas--) {
        int n,c;
        scanf("%d%d",&n,&c);
        int y = f[n];
        printf("%lld\n",quick_pow(c,y));
    }
    return 0;
}
```

C - Count New String

可以发现所谓所有$f(f(S,x_1,y_1),x_2-x_1+1,y_2-x_1+1)$本质不同字符串的个数，就相当于$\sum_{i=1}^nf(S,i,n)$中本质不同字符串个数，然后发现根据这个字符串的性质，如果从后向前更新，那么每个位置最多只需要修改10次，所以在广义sam上乱搞一下就行。

```
#include <cstdio>
#include <algorithm>
#include <cstring>
#define setIO(s) freopen(s".in","r",stdin)
#define maxn 2000005
#define N 12
#define ll long long
using namespace std;
char str[maxn];
int last=1,tot=1,n,m;
int ch[maxn][N],f[maxn][7],dis[maxn],rk[maxn];
int lst[maxn];
long long C[maxn],ans;
```

```
void ins(int c,int id){
    int np=++tot,p=last; last=np;
    if(ch[p][c]){
        int q=ch[p][c];
        if(dis[q]==dis[p]+1) last=q;
        else {
            int nq=++tot; last=nq;
            f[nq][id]=f[q][id],dis[nq]=dis[p]+1;
            memcpy(ch[nq],ch[q],sizeof(ch[q]));
            f[q][id]=nq;
            while(p&&ch[p][c]==q) ch[p][c]=nq,p=f[p][id];
        }
    }
    else{
        dis[np]=dis[p]+1;
        while(p&&!ch[p][c]) {  ch[p][c]=np,p=f[p][id]; }
        if(!p) f[np][id]=1;
        else{
            int q=ch[p][c],nq;
            if(dis[q]==dis[p]+1) f[np][id]=q;
            else{
                nq=++tot;
                dis[nq]=dis[p]+1;
                memcpy(ch[nq],ch[q],sizeof(ch[q]));
                f[nq][id]=f[q][id],f[q][id]=f[np][id]=nq;
                while(p&&ch[p][c]==q) ch[p][c]=nq,p=f[p][id];
            }
        }
        ans+=(dis[np]-dis[f[np][id]]);
    }
}
int main(){
    //setIO("input");
    //int t; scanf("%d",&t);
    //while(t--)
    //{
        scanf("%s",str+1),n=strlen(str+1);
        lst[n+1] = 1;
        for (int i = n;i>=1;i--)
        {
            int j = i+1;
            while (str[i]>str[j] && j <= n) {
                str[j] = str[i];
                j++;
            }
            last = lst[j];
            while (j > i) {
                j--;
                ins(str[j]-'a',0);
                lst[j] = last;
            }
```

```
        }
        printf("%lld\n",ans);
        last = 1;
    //}
    return 0;

}
```

F - Finding the Order

已知直线 AB 平行于直线 CD，且已知 $|AC|,|AD|,|BC|,|BD|$ 四个值，问 $\overrightarrow{AB}$ 与 $\overrightarrow{CD}$ 同方向，还是与 $\overrightarrow{DC}$ 同方向。

分类讨论，如果 $|AC| > |BC|$ 说明点 C 在靠近点 B 而对点 D 的讨论同样。

```
#include<bits/stdc++.h>
#define ALL(x) (x).begin(),(x).end()
#define ll long long
#define ull unsigned long long
#define pii_ pair<int,int>
#define mp_ make_pair
#define pb push_back
#define fi first
#define se second
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define per(i,a,b) for(int i=(a);i>=(b);i--)
#define show1(a) cout<<#a<<" = "<<a<<endl
#define show2(a,b) cout<<#a<<" = "<<a<<"; "<<#b<<" = "<<b<<endl
using namespace std;
const ll INF = 1LL<<60;
const int inf = 1<<30;
const int maxn = 2e5+5;
inline void fastio() {ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);}


int main()
{
    fastio();
    int _;
    for(cin>>_;_;_--){ int ac,ad,bc,bd;
        cin>>ac>>ad>>bc>>bd;
        if(ad>bd){
            if(ac>bc){
                if(ad>ac){
                    cout<<"AB//CD"<<endl;
                }else cout<<"AB//DC"<<endl;
```

```
            }else{
                cout<<"AB//CD"<<endl;
            }
        }else if(ad==bd){
            if(ac>bc){
                cout<<"AB//DC"<<endl;
            }else{
                cout<<"AB//CD"<<endl;
            }
        }else{
            if(ac<bc){
                if(bd>bc){
                    cout<<"AB//DC"<<endl;
                }else cout<<"AB//CD"<<endl;
            }else{
                cout<<"AB//DC"<<endl;
            }
        }
    }
    return 0;
}
```

I - Investigating Legions

个人理解，相当于给一个含有多个近似团的图，要还原出这些团。

大概就是乱搞。我的做法是先贪心取最小的没有确定的点，把和他相邻的点加到一个集合 V 里，遍历所有其他没有确定的点，如果这个点和集合 V 的连接度大于某个和 S 有关的阈值，就把这个点加入当前正在确定的团里。最后仍有些点没有被确定，这时遍历所有确定的团，看他与哪个团的连接度最高就放进哪里。

```
#include<bits/stdc++.h>
#define ALL(x) (x).begin(),(x).end()
#define ll long long
#define ull unsigned long long
#define pii_ pair<int,int>
#define mp_ make_pair
#define pb push_back
#define fi first
#define se second
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define per(i,a,b) for(int i=(a);i>=(b);i--)
#define show1(a) cout<<#a<<" = "<<a<<endl
#define show2(a,b) cout<<#a<<" = "<<a<<"; "<<#b<<" = "<<b<<endl
using namespace std;
const ll INF = 1LL<<60;
const int inf = 1<<30;
```

```
const int maxn = 2e5+5;
inline void fastio() {ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);}

int g[305][305],vis[305],ans[305],be[305],rk[305],c[305];
int n,s;
vector<int> block[305];
int main()
{
    fastio();
    int _; char x;
    for(cin>>_;_;_--){
        cin>>n>>s;
        rep(i,0,n-1) vis[i] = 0,block[i].clear();
        rep(i,0,n-1){
            rep(j,i+1,n-1){
                cin>>x;
                if(x=='1') g[i][j] = g[j][i] = 1;
                else g[i][j] = g[j][i] = 0;
            }
        }
        int id = 0;
        rep(i,0,n-1) if(!vis[i]){
            vector<int> tmp;
            tmp.pb(i);
            rep(j,0,n-1) if(g[i][j] && !vis[j]){
                tmp.pb(j);
            }
            int sz = tmp.size();
            int th = 8 * sz / s;
            rep(j,0,n-1) if(!vis[j]){ int cnt = 0;
                for(int x:tmp) if(g[j][x]) cnt++;
                if(cnt >= th) block[id].pb(j),be[j] = id,vis[j] = 1;
            }
            id++;
        }
        rep(i,0,n-1) if(!vis[i]){
            memset(c,0,sizeof(c));
            rep(j,0,n-1) if(vis[j]){
                c[be[j]]++;
            }
            int pos,mx=0;
            rep(j,0,id-1){
                if(c[j] > mx) pos=j,mx = c[j];
            }
            block[pos].pb(i);
        }
        rep(i,0,id-1){
            rk[i] = i;
            sort(ALL(block[i]));
        }sort(rk,rk+id,[](int x,int y){return block[x][0] < block[y][0];});
```

```
        rep(i,0,id-1){
            for(int x:block[rk[i]]) ans[x] = i;
        }
        rep(i,0,n-1) cout<<ans[i]<<" ";
        cout<<endl;
    }
    return 0;
}
```

比赛总结与反思