2020牛客暑期多校训练营（第九场）

比赛网址

训练结果

- 时间:2020-08-08 12:00~17:00
- rank:120/??
- 完成情况：5/6/12

题解

A.Groundhog and 2-Power Representation

题意

给你一个神奇的表达式（难以描述），让你计算答案

题解

直接上python的eval大法

```
str=input()
ans=''
for i in str:
    if i == '(':
       ans+='**'
    ans+=i
print(eval(ans))
```

B.Groundhog and Apple Tree

题意

题解

```

```

E.Groundhog Chasing Death

题意

求 $ //{i=a}^b//{j=c}^{d(x}i,y^j)$

题解

将 x,y 的每一个公共质因子提出来分别计算，我们枚举 i ,都可以算出了所有 j 对答案的贡献，算出幂数和，最后在快速幂就好了

```
#include<iostream>
#include<cstdio>
#include<algorithm>
#include<cstring>
#define ll long long
using namespace std;
int read()
{
    int k=0,f=1;char c=getchar();
    for(;!isdigit(c);c=getchar()) if(c=='-') f=-1;
    for(;isdigit(c);c=getchar()) k=k*10+c-'0';return k*f;
}
const ll mod=998244353;
ll a,b,c,d,x,y;
ll t1,t2,tot,p1[105],num1[105],num2[105],p2[105],p[105],n1[105],n2[105];
ll sum[105];
ll ksm(ll x,ll k) {ll as=1;for(;k;k>>=1,x=x*x%mod) if(k&1)
as=as*x%mod;return as;}
int main()
{
    a=read();b=read();c=read();d=read();x=read();y=read();
    for(ll i=2;i*i<=x;i++)
        if(x%i==0)
        {
            p1[++t1]=i;
            while(x%i==0)
                num1[t1]++,x/=i;
        }
    if(x!=1) p1[++t1]=x,num1[t1]=1;
    for(ll i=2;i*i<=y;i++)
        if(y%i==0)
        {
            p2[++t2]=i;
            while(y%i==0)
                num2[t2]++,y/=i;
```

```
        }
    if(y!=1) p2[++t2]=y,num2[t2]=1;
    for(int i=1;i<=t1;i++)
    {
        for(int j=1;j<=t2;j++)
            if(p1[i]==p2[j])
                p[++tot]=p1[i],n1[tot]=num1[i],n2[tot]=num2[j];
    }
    ll as=1;
    for(ll i=a;i<=b;i++)
    {
        for(int j=1;j<=tot;j++)
        {
            ll s1=n1[j]*i;
            ll lw=(s1-1)/n2[j]+1;
            if(lw<=c)
                sum[j]+=s1*(d-c+1);
            else if(lw>d)
            {
                sum[j]+=(c+d)*(d-c+1)/2*n2[j];
            }
            else
            {
                sum[j]+=(d-lw+1)*s1;
                sum[j]+=(lw+c-1)*(lw-c)/2*n2[j];
            }
            if(sum[j]>=(1LL<<55))
                as=as*ksm(p[j],sum[j])%mod,sum[j]=0;
        }
    }
    for(int i=1;i<=tot;i++)
        as=as*ksm(p[i],sum[i])%mod;
    cout<<as<<endl;
    return 0;
}
```

F.Groundhog Looking Dowdy

题意

在每个集合中选一个数，使得所有数的极差最小

题解

将所有数标上集合的颜色后排序，双指针移动，每次移动到一个包含所有颜色的区间，计算极差

```
#include<algorithm>
```

```
#include<iostream>
#include<cstdlib>
#include<cstring>
#include<cstdio>
#include<vector>
#include<queue>
#include<cmath>
#include<map>
#include<set>
#define LL long long int
#define REP(i,n) for (int i = 1; i <= (n); i++)
#define Redge(u) for (int k = h[u],to; k; k = ed[k].nxt)
#define cls(s,v) memset(s,v,sizeof(s))
#define mp(a,b) make_pair<int,int>(a,b)
#define cp pair<int,int>
using namespace std;
const int maxn = 2000005,maxm = 100005,INF = 0x3f3f3f3f;
inline int read(){
    int out = 0,flag = 1; char c = getchar();
    while (c < 48 || c > 57){if (c == '-') flag = 0; c = getchar();}
    while (c >= 48 && c <= 57){out = (out << 1) + (out << 3) + c - 48; c =
getchar();}
    return flag ? out : -out;
}
struct node{
    int x,c;
}A[maxn];
inline bool operator <(const node& a,const node& b){
    return a.x < b.x;
}
int N,n,m,ans = INF;
int cnt[maxn],tot;
void work(){
    int l = 1,r = 0;
    while (r < N){
        while (tot < m && r < N){
            r++;
            if (!cnt[A[r].c]) tot++;
            cnt[A[r].c]++;
        }
        ans = min(ans,A[r].x - A[l].x);
        while (l <= r && tot >= m){
            if (cnt[A[l].c] == 1) tot--;
            cnt[A[l].c]--;
            l++;
            if (tot >= m) ans = min(ans,A[r].x - A[l].x);
        }
    }
    while (l <= r && tot >= m){
```

```
        if (cnt[A[l].c] == 1) tot--;
        cnt[A[l].c]--;
        l++;
        if (tot >= m) ans = min(ans,A[r].x - A[l].x);
    }
    printf("%d\n",ans);
}
int main(){
    n = read(); m = read();
    for (int i = 1; i <= n; i++){
        int t = read();
        while (t--) A[++N] = (node){read(),i};
    }
    sort(A + 1,A + 1 + N);
    work();
    return 0;
}
```

I.The Crime-solving Plan of Groundhog

题意

有n个一位数，让你组成两个数（不能有前导零），问乘积最小是多少

题解

发现一定是把这n个数排序，然后选第一个数作为第一个乘数，后面的作为第二个乘数最小，讨论一下0即可。

```
#include<bits/stdc++.h>
using namespace std;
#define mem(a,b) memset(a,b,sizeof(a))
typedef long long LL;
typedef pair<int,int> PII;
#define X first
#define Y second
inline int read()
{
    int x=0,f=1;char c=getchar();
    while(!isdigit(c)){if(c=='-')f=-1;c=getchar();}
    while(isdigit(c)){x=x*10+c-'0';c=getchar();}
    return x*f;
}
const int maxn=100010;
int n,a[maxn],b[maxn];
struct data
```

```
{
    int l,v[maxn];
    data(){l=1;memset(v,0,sizeof(v));}
    data operator * (const int& t)
     {
        data c;c.l=l;
        for(int i=1;i<=l;i++)c.v[i]=v[i]*t;
        for(int i=1;i<c.l;i++)c.v[i+1]+=c.v[i]/10,c.v[i]%=10;
        while(c.v[c.l]>9)c.v[c.l+1]+=c.v[c.l]/10,c.v[c.l]%=10,c.l++;
        return c;
    }
};
void print(data s){for(int i=s.l;i;i--)printf("%d",s.v[i]);}
int main()
{
    for(int T=read();T;T--)
    {
        n=read();
        for(int i=0;i<n;i++)a[i]=read();
        sort(a,a+n);
        int j=0;
        while(a[j]==0 && j<n)j++;
        a[0]=a[j];a[1]=a[j+1];
        for(int i=0;i<j;i++)a[i+2]=0;
        data A;
        A.l=n-1;
        for(int i=1;i<n;i++)A.v[i]=a[A.l-i+1];
        print(A*a[0]);
        printf("\n");
    }
    return 0;
}
```

J.The Escape Plan of Groundhog

题意

01矩阵，问有多少个子矩阵，满足：边长≥2□最外围一圈全是1，内部的01个数的差值不超过1

题解

先枚举上下边界，然后我们只处理上下边界中全是1的部分，对于如何快速算“ 内部的01个数不超过1 ”的条件，我们用前缀和记录一下当前有多少个1和0，（1记为1,0记为-1），开个桶记录一下当前$pre,pre-1,pre+1$分别有多少个，直接统计答案即可，复杂度$O(n^3)$

```
#include<bits/stdc++.h>
using namespace std;
#define mem(a,b) memset(a,b,sizeof(a))
typedef long long LL;
typedef pair<int,int> PII;
#define X first
#define Y second
inline int read()
{
    int x=0,f=1;char c=getchar();
    while(!isdigit(c)){if(c=='-')f=-1;c=getchar();}
    while(isdigit(c)){x=x*10+c-'0';c=getchar();}
    return x*f;
}
const int maxn=510;
int n,m,mat[maxn][maxn],a[maxn],b[maxn],c[maxn],ans,have[maxn*maxn*2];
vector <int> V;
void work(int l,int r,int len)
{
    int last=l;
    while(a[last]!=len && last<=r)last++;
    int pre=n*m;
    V.push_back(pre);
    have[pre]++;
    for(int i=last+1;i<=r;i++)
    {
        if(a[i]==len)
        {
            ans+=have[pre]+have[pre+1]+have[pre-1];
            have[pre+b[i]]++;
            V.push_back(pre+b[i]);
        }
        pre+=b[i];
    }
    for(int i=0;i<V.size();i++)have[V[i]]--;
    V.clear();
}
int main()
{
    n=read();m=read();
    for(int i=1;i<=n;i++)for(int j=1;j<=m;j++)mat[i][j]=read();
    for(int i=1;i<=n;i++)
    {
        for(int j=1;j<=m;j++)a[j]=(mat[i][j]==1)? 1 : -1;
        for(int j=i+1;j<=n;j++)
        {
            for(int k=1;k<=m;k++)a[k]+=(mat[j][k]==1)? 1 : -1,b[k]=a[k]-
(mat[i][k]+mat[j][k]);
            for(int k=1;k<=m;k++)c[k]=mat[i][k]&mat[j][k];
            for(int k=1;k<=m;k++)
            {
```

```
                int l=k;
                while(k<=m && c[k]==1)k++;
                int r=k-1;
                if(l+1<=r)work(l,r,j-i+1);
            }
        }
    }
    printf("%d\n",ans);
    return 0;
}
```

K.The Flee Plan of Groundhog

题意

一个人从一棵树的1号点走向n号点，每秒走一步，在这个过程中第t秒n点有一个人出现开始每秒走两步去追第一个人，问第一个人此时开始逃跑到被追上最长的时间

题解

分两种情况，要么回头找最远的叶子走，要么向前走到每个点进入一个子树的最远叶子

分类求一下即可

```
#include<algorithm>
#include<iostream>
#include<cstdlib>
#include<cstring>
#include<cstdio>
#include<vector>
#include<queue>
#include<cmath>
#include<map>
#include<set>
#define LL long long int
#define REP(i,n) for (int i = 1; i <= (n); i++)
#define Redge(u) for (int k = h[u],to; k; k = ed[k].nxt)
#define cls(s,v) memset(s,v,sizeof(s))
#define mp(a,b) make_pair<int,int>(a,b)
#define cp pair<int,int>
using namespace std;
const int maxn = 100005,maxm = 100005,INF = 0x3f3f3f3f;
inline int read(){
    int out = 0,flag = 1; char c = getchar();
    while (c < 48 || c > 57){if (c == '-') flag = 0; c = getchar();}
```

```
    while (c >= 48 && c <= 57){out = (out << 1) + (out << 3) + c - 48; c =
getchar();}
    return flag ? out : -out;
}
int h[maxn],ne;
struct EDGE{
    int to,nxt;
}ed[maxn * 2];
void build(int u,int v){
    ed[++ne] = (EDGE){v,h[u]}; h[u] = ne;
    ed[++ne] = (EDGE){u,h[v]}; h[v] = ne;
}
int n,t,tag[maxn],dep[maxn],fa[maxn];
int st[maxn];
void dfs1(int u){
    Redge(u) if ((to = ed[k].to) != fa[u]){
        fa[to] = u; dep[to] = dep[u] + 1;
        dfs1(to);
        if (tag[to]) tag[u] = true;
    }
    if (u == n) tag[u] = true;
}
int dfs2(int u,int f){
    int mx = 0;
    Redge(u) if ((to = ed[k].to) != f){
        mx = max(mx,dfs2(to,u) + 1);
    }
    return mx;
}
void work(){
    dfs1(1);
    if (t >= dep[n]){puts("0"); return;}
    int rt = 0,ans = 0;
    REP(i,n) if (tag[i] && dep[i] == t){rt = i; break;}
    REP(i,n) if (tag[i]) st[dep[i]] = i;
    //for (int i = 0; i <= dep[n]; i++) printf("%d ",st[i]); puts("");
    int mx = 0;
    Redge(rt) if (!tag[to = ed[k].to] || to == fa[rt]){
        mx = max(mx,dfs2(to,rt) + 1);
    }
    int tmp,len = dep[n] - dep[rt];
    if (mx >= len) tmp = len;
    else tmp = mx + (len - mx + 1) / 2;
    ans = tmp;
    //cout << ans << endl;
    for (int i = dep[rt] + 1; i < dep[n]; i++){
        int u = st[i];
        //printf("[%d]\n",u);
        int d = dep[n] - dep[rt] - 3 * (i - dep[rt]),mx = 0;
        if (d <= 0) break;
        Redge(u) if (!tag[to = ed[k].to]){
```

```
            mx = max(mx,dfs2(to,u) + 1);
        }
        if (mx >= d) tmp = d;
        else tmp = mx + (d - mx + 1) / 2;
        //cout << d << ' ' << tmp << ' ' << i << ' ' << dep[rt] << endl;
        ans = max(ans,tmp + i - dep[rt]);
    }
    printf("%d\n",ans);
}
int main(){
    n = read(); t = read();
    for (int i = 1; i < n; i++) build(read(),read());
    work();
    return 0;
}
```

训练实况

训练总结

wxg: 这场时间分配有点问题□E应该早点写□B没有调出来也是遗憾

hxm:很遗憾没有调出B

fyh:开局不错，但是J我又读错题了，就沿着错误的方向就去想一直没有想出来。另外□n=500是可以过n^3的。