[2019 Multi-University Training Contest 1]

比赛网址

训练详情

- 时间:2020-5-17 13:00~18:00
- rank:' 93/506 '
- 完成情况：' 7/10/13 '

题解

A Rush Hour Puzzle

题意

一个大家都玩过的游戏，求步数

题解

solved by fyh

把6*6哈希压成一个状态，然后直接bfs暴力搜 代码写的过于丑，但是基本都是复制粘贴的。

```
#include<bits/stdc++.h>
#include<map>
using namespace std;
#define mem(a,b) memset(a,b,sizeof(a))
typedef long long LL;
typedef unsigned long long ULL;
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;
}
struct State
{
    int st[10][10];
```

```
    State() {mem(st,0);}
    ULL Hash()
    {
        ULL res=0;
        for(int i=0;i<6;i++)
            for(int j=0;j<6;j++)
                res=(res+st[i][j])*107;
        return res;
    }
};
map<ULL,int> vis;
map<ULL,int> dis;
queue<State> Q;
int main()
{
    State init;
    for(int i=0;i<6;i++)
        for(int j=0;j<6;j++)
            init.st[i][j]=read();
    Q.push(init);
    ULL hinit=init.Hash();
    vis[hinit]=1;
    dis[hinit]=0;
    while(Q.size())
    {
        State now=Q.front();Q.pop();
        ULL hnow=now.Hash();
        if(dis[hnow]>8)return puts("-1"),0;
        if(now.st[2][5]==1 && now.st[2][4]==1)return
printf("%d\n",dis[hnow]+2),0;
        for(int i=0;i<6;i++)
            for(int j=0;j<6;j++)
            {
                if(now.st[i][j] && j+1<6 && now.st[i][j]==now.st[i][j+1] &&
(now.st[i][j+1]!=now.st[i][j+2] || j+2>=6) && (now.st[i][j-1]!=now.st[i][j]
|| j<=0))//横2车
                {
                    if(j>0 && !now.st[i][j-1])//左移
                    {
                        State tmp=now;
                        tmp.st[i][j-1]=now.st[i][j];
                        tmp.st[i][j+1]=0;
                        ULL htmp=tmp.Hash();
                        if(!vis[htmp])
                        {
                            dis[htmp]=dis[hnow]+1;
                            vis[htmp]=1;
                            Q.push(tmp);
                        }
                    }
```

```
                    if(j+2<6 && !now.st[i][j+2])
                    {
                        State tmp=now;
                        tmp.st[i][j+2]=now.st[i][j];
                        tmp.st[i][j]=0;
                        ULL htmp=tmp.Hash();
                        if(!vis[htmp])
                        {
                            dis[htmp]=dis[hnow]+1;
                            vis[htmp]=1;
                            Q.push(tmp);
                        }
                    }//右移
                }
                else if(now.st[i][j] && j+2<6 &&
now.st[i][j]==now.st[i][j+1] && now.st[i][j+1]==now.st[i][j+2])//横3车
                {
                    if(j>0 && !now.st[i][j-1])//左移
                    {

                        State tmp=now;
                        tmp.st[i][j-1]=now.st[i][j];
                        tmp.st[i][j+2]=0;
                        ULL htmp=tmp.Hash();
                        if(!vis[htmp])
                        {
                            dis[htmp]=dis[hnow]+1;
                            vis[htmp]=1;
                            Q.push(tmp);
                        }
                    }
                    if(j+3<6 && !now.st[i][j+3])
                    {
                        State tmp=now;
                        tmp.st[i][j+3]=now.st[i][j];
                        tmp.st[i][j]=0;
                        ULL htmp=tmp.Hash();
                        if(!vis[htmp])
                        {
                            dis[htmp]=dis[hnow]+1;
                            vis[htmp]=1;
                            Q.push(tmp);
                        }
                    }//右移
                }
                else if(now.st[i][j] && i+1<6 &&
now.st[i][j]==now.st[i+1][j] && (now.st[i+1][j]!=now.st[i+2][j] || i+2>=6)
&& (now.st[i-1][j]!=now.st[i][j] || i<=0))//竖2车
                {
                    if(i>0 && !now.st[i-1][j])//上移
                    {
```

```
                        State tmp=now;
                        tmp.st[i-1][j]=now.st[i][j];
                        tmp.st[i+1][j]=0;
                        ULL htmp=tmp.Hash();
                        if(!vis[htmp])
                        {
                            dis[htmp]=dis[hnow]+1;
                            vis[htmp]=1;
                            Q.push(tmp);
                        }
                    }
                    if(i+2<6 && !now.st[i+2][j])
                    {
                        State tmp=now;
                        tmp.st[i+2][j]=now.st[i][j];
                        tmp.st[i][j]=0;
                        ULL htmp=tmp.Hash();
                        if(!vis[htmp])
                        {
                            dis[htmp]=dis[hnow]+1;
                            vis[htmp]=1;
                            Q.push(tmp);
                        }
                    }//右移
                }
                else if(now.st[i][j] && i+2<6 &&
now.st[i][j]==now.st[i+1][j] && now.st[i+1][j]==now.st[i+2][j])//竖3车
                {
                    if(i>0 && !now.st[i-1][j])//上移
                    {
                        State tmp=now;
                        tmp.st[i-1][j]=now.st[i][j];
                        tmp.st[i+2][j]=0;
                        ULL htmp=tmp.Hash();
                        if(!vis[htmp])
                        {
                            dis[htmp]=dis[hnow]+1;
                            vis[htmp]=1;
                            Q.push(tmp);
                        }
                    }
                    if(i+3<6 && !now.st[i+3][j])
                    {
                        State tmp=now;
                        tmp.st[i+3][j]=now.st[i][j];
                        tmp.st[i][j]=0;
                        ULL htmp=tmp.Hash();
                        if(!vis[htmp])
                        {
                            dis[htmp]=dis[hnow]+1;
```

```
                            vis[htmp]=1;
                            Q.push(tmp);
                        }
                    }//右移
                }
            }
    }
    puts("-1");
    return 0;
}
```

B The Power Monitor System

题意

一棵树$n\leq10^5$初始点和边都是白的，目标是将所有的边和点都染黑。操作代价为1，可以将一个节点和节点的所有出边全部染黑，服从以下三个规则：

- rule1:如果一条边被染黑，两个端点也被染黑。
- rule2:如果一条边两个点都被染黑，该边被染黑。
- rule3:一个点的度数$k>1$且$k-1$条边都被染黑，则剩下一条也被染黑

题解

补题 by fyh

很可怕的一个树形DP

设计状态$dp[u][0/1/2][0/1]$第二维度的0表示该点由儿子传递过来，1表示自己染，2表示由父亲传递过来。第三维度的0表示不使用规则3进行染色，1表示使用规则3进行染色。

那么就是转移了

$dp[u][1][0]=\sum min(dp[v][0/1/2][0/1])$ 这个转移很显然，这个点都染上色了，子节点就可以xjb选了

$dp[u][1][1]$显然离谱，我们就不管他了

$dp[u][2][0]=\sum dp[v][0][1]$ 不通过规则3染色，故要所有的儿子都是通过规则3生成的，否则(u,v)该条边会被染黑

$dp[u][2][1]$为在$dp[u][2][0]$下选择一颗子树，将其替换成$min\{dp[v][2][0/1]\}$表示(u,v)这条边是由规则3生成染黑的。

$dp[u][0][0]$ 为至少有一棵子树为$dp[v][1/0][0]$,(前者是儿子直接染黑，后者是儿子被染黑，而且儿子除了父边之外的所有边都是0,通过规则3使得u被染上的。)+其他子树取$min(00,01,10)$

$dp[u][0][1]$为在$dp[u][0][0]$ 下的其他子树中选择一颗子树，将其替换成$min\{dp[v][2][0/1]\}$

就是感觉这个DP在第三维度定义的很玄学，于是我们的zzh鸽鸽就想到了把状态放在边上的做法…真的太强了。

```
#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;
vector<int> G[maxn];
int n,u,v,dp[maxn][3][2];
void dfs(int now,int fa)
{
    for(int i=0;i<3;i++)for(int j=0;j<2;j++)dp[now][i][j]=maxn;
    dp[now][1][0]=1;dp[now][2][0]=0;
    if(G[now].size()==1 && fa)return;
    dp[now][0][0]=dp[now][0][1]=dp[now][2][0]=dp[now][2][1]=0;
    int choice[2],tp1[2][2]={0,maxn,maxn,maxn};//choice[]
    choice[0]=0;choice[1]=maxn;
    for(auto &v: G[now])
    {
        if(v==fa)continue;
        dfs(v,now);
        int tmp=min(dp[v][1][0],min(dp[v][0][0],dp[v][0][1]));
choice[1]=min(choice[1]+tmp,choice[0]+min(dp[v][0][0],dp[v][1][0])),choice[
0]+=tmp;
tp1[1][1]=min(tp1[1][1]+tmp,min(tp1[0][1]+min(dp[v][2][0],dp[v][2][1]),tp1[
1][0]+min(dp[v][0][0],dp[v][1][0])));
tp1[0][1]=min(tp1[0][1]+tmp,tp1[0][0]+min(dp[v][0][0],dp[v][1][0]));
tp1[1][0]=min(tp1[1][0]+tmp,tp1[0][0]+min(dp[v][2][0],dp[v][2][1]));
        tp1[0][0]+=tmp;

dp[now][1][0]+=min(min(dp[v][0][0],dp[v][0][1]),min(dp[v][1][0],min(dp[v][2
][0],dp[v][2][1])));
dp[now][2][1]=min(dp[now][2][0]+min(dp[v][2][0],dp[v][2][1]),dp[now][2][1]+
dp[v][0][1]);
        dp[now][2][0]+=dp[v][0][1];
    }
    dp[now][0][0]=choice[1],dp[now][0][1]=tp1[1][1];
}
```

```
int main()
{
    n=read();
    for(int
i=1;i<n;i++)u=read(),v=read(),G[u].push_back(v),G[v].push_back(u);
    dfs(1,0);
    printf("%d\n",min(dp[1][1][0],min(dp[1][0][0],dp[1][0][1])));
    return 0;
}
```

C Are They All Integers?

solved by hxm

题意

大水题

题解

略

D Tapioka

solved by wxg

题意

删掉一个字符串序列中的指定的字符串

题解

无敌水题，略

E The League of Sequence Designers

solved by wxg&hxm

题意

给定一个数字序列，求最大的连续和乘以长度。

现在给出一个错误的做法，错误做法仅计算最大连续和来更新答案

求构造出一个长度大于l小于2000的序列，使得错误答案和正确答案相差正好为k

题解

分析发现，只要在最大连续和的串前加上一个负数，那么这个做法就会出错，现在要构造出两种答案相差k

由于最大长度为1999，不妨就构造长度为1999的序列，第一位为-1，剩下为非负数，和为a则有

$(a - 1) \times 1999 - a \times 1998 = k$

即$a = k + 1999$算出a后，分配给剩下每一位即可

```
#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 K,L;
int ans[maxn];
void work(){
    int sum = 1999 + K;
```

```
    printf("1999\n");
    printf("-1");
    for (int i = 1; i <= 1997; i++) printf(" %d",sum / 1998);
    printf(" %d\n",sum - sum / 1998 * 1997);
}
int main(){
    int T = read();
    while (T--){
        K = read(); L = read();
        if (L >= 2000) puts("-1");
        else work();
    }
    return 0;
}
```

H Mining a

solved by fyh&hxm

题意

$\frac{1}{n} = \frac{1}{a Xor b} + \frac{1}{b}$

给定n□b是任意的，求最大的a使等式成立

题解

化简得$b = n + \frac{n^2}{a Xor b - n}$

枚举分母即可

```
#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;
}
LL n,a,b,N,ans;
int main(){
    int T =read();
    while (T--){
        ans = 0;
        n = read(); N = n * n;
        LL t;
        for (int i = 1; i < n; i++){
            if (N % i == 0){
                //1
                t = i;
                b = N / t + n;
                a = (t + n) ^ b;
                ans = max(ans,a);
                //2
                t = N / i;
                b = N / t + n;
                a = (t + n) ^ b;
                ans = max(ans,a);
            }
        }
        printf("%I64d\n",ans);
    }
    return 0;
}
```

J Automatic Control Machine

solved by hxm

题意

给定若干个集合，使用最少的集合并成全集

题解

bitset状压dp一下就好了

```

```

K Automatic Control Machine

solved by wxg

题意

合并果子，数据还特小，题解略

J Largest Quadrilateral

题意

求平面点组成的最大四边形面积$n\leq5000$

题解

补题by fyh

去除重复点，求出凸包,然后旋转卡壳枚举每个点的最远点，然后暴力计算四边形面积

退化情况有：

凸包上只有三个点，那么这个四边形一定是一个凹四边形，计算出最小三角形面积然后用凸包面积减去小三角形面积就可以。

凸包上只有两个点：（再**的见）

训练实况

0~15min hxm wxg横扫**K,C,D**三个签到题

15min~1h20min 口糊出**A**,fyh开写**A**,wxg,hxm讨论**E**,并讨论出来

1h20min~1h28min **A** 本机编译错误，哈希出现问题 hxm开**E**,并A

1h28min~1h??min 想出**H**,写**H** 并A

2h~2h14min hxmA**J**,其他人在读题

2h14min~2h36min fyhA**A**,

2h36min~ 读完所有题并确认全不可做，结束比赛

训练总结

像**A**这种代码量较大的题要尽量往后放，否则会增加罚时

尽早放弃

改进

From:
https://wiki.cvbbacm.com/ - **CVBB ACM Team**

Permanent link:
https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:die_java:front_page_springtraining6&rev=1590165967

Last update: **2020/05/23 00:46**