The 2015 ACM-ICPC Asia Beijing Regional Contest

比赛网址

训练结果

- 时间:2020-08-24 13:00-16:00
- rank:50/564
- 完成情况：5/6/11

题解

A - Xiongnu’s Land

题解

```

```

C - Today Is a Rainy Day

题意

给了两个字符串，只包含1-6的数字，你每次可以把一个数或者一类数变成另一个数，问你一个串转为另一个串的最小操作数

题解

发现最优肯定是先改一类数，先用 bfs 求出原来1-6分别变成什么数的最小操作次数，最后求一下变完之后还要改几次即可。

```
 #include<iostream>
#include<cstdio>
#include<algorithm>
#include<cstring>
#include<queue>
#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;
}
int n,m,dp[6*6*6*6*6*6],num[7],nxt[7],a[6][6];
queue<int> q;
char s[1005],t[1005];
void solve()
{
    memset(dp,-1,sizeof(dp));
    int s=0;
    for(int i=5;i>=0;i--)
        s=s*6+i;
    dp[s]=0;q.push(s);
    while(!q.empty())
    {
        int u=q.front();q.pop();
        int uu=u;
        for(int i=0;i<6;i++)
            num[i]=uu%6,uu/=6;
        for(int i=0;i<=5;i++)
            for(int j=0;j<=5;j++)
            {
                if(i==j)  continue;
                int v=0;
                for(int k=5;k>=0;k--)
                {
                    if(num[k]==i)
                        nxt[k]=j;
                    else nxt[k]=num[k];
                    v=v*6+nxt[k];
                }
                if(dp[v]==-1)
                    dp[v]=dp[u]+1,q.push(v);
            }
    }
}

int main()
{
    solve();
    while(scanf("%s",t)!=EOF)
    {
        scanf("%s",s);
        n=strlen(s);
        int ans=100005;
        memset(a,0,sizeof(a));
        for(int i=0;i<n;i++)
            a[s[i]-'1'][t[i]-'1']++;
```

```
        for(int i=0;i<6*6*6*6*6*6;i++)
        {
            if(dp[i]==-1) continue;
            int res=dp[i];int s=i;
//          if(dp[i]<=0) cout<<i<<" ";
            for(int j=0;j<6;j++)
                num[j]=s%6,s/=6;
            for(int j=0;j<=5;j++)
                for(int k=0;k<=5;k++)
                    if(num[j]!=k)
                        res+=a[j][k];
            ans=min(res,ans);
        }
        printf("%d\n",ans);
    }
    return 0;
}
```

D - Kejin Game

题意

有一棵技能图（保证是DAG□□习得一个技能需要花费a_i个价钱，以及要把所有前置技能学会，当然你可以直接花费b_i通过氪金将这个技能直接习得。问目标技能的最小花费。

题解

最小割。割掉这条边便表示进行此次花费，如果一个点断流了就表示这个点已经学得。于是我们考虑这样的建图：把每个技能拆成两个点(a,a')□源点向每个点的一号点连容量为a_i的边，对于DAG上的边(u,v,w),u'向v连w的边□u向u'连b_u□汇点就是目标技能点的二号点。

```
#include<iostream>
#include<cstdio>
#include<cmath>
#include<queue>
#include<cstring>
#include<algorithm>
#define LL long long int
#define Redge(u) for (int k = h[u],to; k; k = ed[k].nxt)
#define REP(i,n) for (int i = 1; i <= (n); i++)
#define BUG(s,n) for (int i = 1; i <= (n); i++) cout<<s[i]<<' '; puts("");
#define cls(s) memset(s,0,sizeof(s))
#define eps  1e-9
using namespace std;
const int maxn=(510)<<1,maxm=10010*4,INF = 1000000000;
inline int read(){
```

```
    int out = 0,flag = 1; char c = getchar();
    while (c < 48 || c > 57){if (c == '-') flag = -1; c = getchar();}
    while (c >= 48 && c <= 57){out = (out << 3) + (out << 1) + c - 48; c =
getchar();}
    return out * flag;
}
int h[maxn],ne = 2;
struct EDGE{int to,nxt; int f;}ed[maxm];
int n,m,a,b,c,s;
LL ke[maxn],val[maxn];
inline void build(int u,int v,int f){
    ed[ne] = (EDGE){v,h[u],f}; h[u] = ne++;
    ed[ne] = (EDGE){u,h[v],0}; h[v] = ne++;
}
int d[maxn],vis[maxn],cur[maxn],S,T;
bool bfs(){
    cls(d); cls(vis); vis[S] = true;
    queue<int> q; q.push(S);
    int u;
    while (!q.empty()){
        u = q.front(); q.pop();
        Redge(u) if (fabs(ed[k].f) > eps && !vis[to = ed[k].to]){
            d[to] = d[u] + 1; vis[to] = true;
            if (to == T) return true;
            q.push(to);
        }
    }
    return vis[T];
}
LL dfs(int u,int minf){
    if (u == T || abs(minf) < eps) return minf;
    int f,flow = 0; int to;
    if (cur[u] == -1) cur[u] = h[u];
    for (int& k = cur[u]; k; k = ed[k].nxt){
        if (d[to = ed[k].to] == d[u] + 1 && abs(f =
dfs(to,min(minf,ed[k].f))) >= eps){
            ed[k].f -= f; ed[k ^ 1].f += f;
            flow += f; minf -= f;
            if (abs(minf) < 0) break;
        }
    }
    return flow;
}
LL maxflow(){
    LL flow = 0;
    while (bfs()){
        memset(cur,-1,sizeof(cur));
        flow += dfs(S,INF);
    }
    return flow;
```

```
}
int main(){
    for(int Case=read();Case;Case--)
    {
        ne=2;
        cls(h);
        n=read();m=read();s=read();
        S=2*n+1;T=s+n;
        while(m--)
        {
            a=read(),b=read(),c=read();
            build(a+n,b,c);
        }
        for(int i=1;i<=n;i++)val[i]=read(),build(S,i,val[i]);
        for(int i=1;i<=n;i++)ke[i]=read(),build(i,i+n,ke[i]);
        printf("%lld\n",maxflow());
    }
    return 0;
}
```

G - Mysterious Antiques in Sackler Museum

题意

给四个矩形的长与宽，问是否能选出三个矩形，拼出一个更大的矩形。

题解

分类讨论，一种是三个并排放，一种是俩纵向放，另一个横向放。

```
#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;
}
int a[4][2],b[4],n;
int judge(int x,int y,int z)
{
```

```
    for(int i=0;i<2;i++)
        for(int j=0;j<2;j++)
            for(int k=0;k<2;k++)
            {
                if(a[x][i]==a[y][j]&&a[y][j]==a[z][k])return 1;
                if(a[x][i]==a[y][j]+a[z][k]&&a[y][j^1]==a[z][k^1])return 1;
            }
    return 0;
}
int main()
{
    for(int T=read();T;T--)
    {
        for(int i=0;i<4;i++)scanf("%d%d",&a[i][0],&a[i][1]);
        for(int i=0;i<4;i++)b[i]=i;
        bool ok=0;
        do{
            if(judge(b[0],b[1],b[2])){ok=1;break;}
        }while(next_permutation(b,b+4));
        if(ok)puts("Yes");
        else puts("No");
    }
    return 0;
}
```

J - Osu! Master

题意

签到题

K - A Math Problem

题意

$$ f[1]=1 \\ 3f[n]f[2n+1]=f[2n](1+3f[n]) \\ f(2n)<6f(n) \\ g[i]=\sum_{k=0}^{mod-1} [f[k]\%mod=i] \\ 求g[0]XOR\ldotsXOR g[mod-1] $$

题解

推推式子发现$f[n]$便是把n写成二进制，只不过二进制上每一位权值都变成了3而已。

数位DP□$dp[k][rest][left]$表示当前模的数是第几个，当前在第$rest$位，模数剩余$left$的方案数，暴力转移即可。

训练实况

训练总结

wxg: 开始把D当成dp题耽误了很多时间

hxm:

fyh:一开始写G题发现讨论有点恶心，当时有点混乱，冷静下来之后才想到用next_permutation减少代码量。