A. Suborrays

所有排列都可以。

```
#include<bits/stdc++.h>
#define ll long long
#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define pii pair<int,int>
using namespace std;
const int N=1e5+10;
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n;
        scanf("%d",&n);
        for(int i=1;i<=n;i++)printf("%d ",i);
        puts("");
    }
    return 0;
}
```

B. Fix You

只要使最下边的行全向右、最右边的列全向下即可。

```
#include<bits/stdc++.h>
#define ll long long
#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define pii pair<int,int>
using namespace std;
const int N=105;
char s[N][N];
int main()
{
```

```
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n,m,ans=0;
        scanf("%d%d",&n,&m);
        for(int i=1;i<=n;i++)scanf("%s",s[i]+1);
        for(int i=1;i<m;i++)if(s[n][i]!='R')ans++;
        for(int i=1;i<n;i++)if(s[i][m]!='D')ans++;
        printf("%d\n",ans);
    }
    return 0;
}
```

C. Cyclic Permutations

长度为 n 的排列，每个位置向前面、后面最近的大于它的数字连边，问有环的方案有多少种。

观察，发现没有环的排列是单峰的，单峰排列 2^{n-1} 种（考虑最大值以外的数字放左边还是右边）。

```
#include<bits/stdc++.h>
#define ll long long
#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define pii pair<int,int>
using namespace std;
const int N=1e6+10;
const ll mod=1e9+7;
ll qpow(ll a,ll n){ll
res=1;while(n){if(n&1)res=res*a%mod;a=a*a%mod,n>>=1;}return res;}
int main()
{
    int n;ll res=1;
    scanf("%d",&n);
    for(int i=1;i<=n;i++)res=res*i%mod;
    printf("%lld\n",(res-qpow(2,n-1)+mod)%mod);
    return 0;
}
```

D. 505

给出 $n\times m$ 的矩阵，要使所有长宽都为偶数的子矩阵 1 的个数都是奇数最少改变多少位置的值。

首先如果 n,m 都不小于 4 是无解的 n 或 m 为 1 直接满足条件。可以讨论一下 n,m 中有一个为 2 或 3 的情况，只需要让所有 2×2 子矩阵维持奇数 1 即可。

$n\times2$ 考虑所有 1×2 长方形，肯定是奇偶交替排列，枚举第一个位置应该是奇数还是偶数即可。

$n\times3$ 对于一个 1×3 矩形，两个 1×2 部分有一个交叠，还是枚举第一个位置两个 1×2 各自的奇偶性，对于某个位置如果两个 1×2 都需改变只改变中间交叠的一格即可。

```
#include<bits/stdc++.h>
#define ll long long
#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define pii pair<int,int>
using namespace std;
const int N=1e6+10;
char s[N];
int a[N][4],n,m,res=N;
int pos(int x,int y){return (x-1)*m+y;}
int main()
{
    scanf("%d%d",&n,&m);
    if(n>=4&&m>=4){puts("-1");return 0;}
    if(n==1||m==1){puts("0");return 0;}
    if(m<4)
    for(int i=1;i<=n;i++)
    {
        scanf("%s",s+1);
        for(int j=1;j<=m;j++)a[i][j]=s[j]-'0';
    }
    else
    {
        for(int i=1;i<=n;i++)
        {
            scanf("%s",s+1);
            for(int j=1;j<=m;j++)a[j][i]=s[j]-'0';
        }
        swap(n,m);
    }
    if(m==2)
    {
        int cnt=0;
        for(int i=1,j=0;i<=n;i++,j^=1)if(((a[i][1]+a[i][2])&1)!=j)cnt++;
```

```
        res=min(res,cnt),cnt=0;
        for(int i=1,j=1;i<=n;i++,j^=1)if(((a[i][1]+a[i][2])&1)!=j)cnt++;
        res=min(res,cnt);
        printf("%d\n",res);
    }
    else if(m==3)
    {
        int cnt=0;
        for(int i=1,j=0,k=0;i<=n;i++,j^=1,k^=1)
        {
            if(((a[i][1]+a[i][2])&1)!=j)cnt++;
            else if(((a[i][2]+a[i][3])&1)!=k)cnt++;
        }
        res=min(res,cnt),cnt=0;
        for(int i=1,j=0,k=1;i<=n;i++,j^=1,k^=1)
        {
            if(((a[i][1]+a[i][2])&1)!=j)cnt++;
            else if(((a[i][2]+a[i][3])&1)!=k)cnt++;
        }
        res=min(res,cnt),cnt=0;
        for(int i=1,j=1,k=0;i<=n;i++,j^=1,k^=1)
        {
            if(((a[i][1]+a[i][2])&1)!=j)cnt++;
            else if(((a[i][2]+a[i][3])&1)!=k)cnt++;
        }
        res=min(res,cnt),cnt=0;
        for(int i=1,j=1,k=1;i<=n;i++,j^=1,k^=1)
        {
            if(((a[i][1]+a[i][2])&1)!=j)cnt++;
            else if(((a[i][2]+a[i][3])&1)!=k)cnt++;
        }
        res=min(res,cnt);
        printf("%d\n",res);
    }
    return 0;
}
```

E. Pairs of Pairs

随便搞一个dfs树，如果有深度大于 $\lceil\frac n2\rceil$ 的点直接输出到根的这条链，如果无，可以对于每个深度值，将同深度的点两两配对，每个深度最多只有一个点未匹配，即未匹配的点数小于 $\lceil\frac n2\rceil$

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

```cpp
#define ll long long
#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define pii pair<int,int>
using namespace std;
const int N=5e5+10,M=1e6+10;
vector<int>g[N];
int q,head[N],cnt,n,m,dep[N],pa[N],f,vis[N];
struct Node{int nxt,to;}edges[M*2];
void addedge(int u,int v){edges[++cnt]=Node{head[u],v},head[u]=cnt;}
void dfs1(int u)
{
    if(dep[u]+1>=q)f=u;else g[dep[u]].pb(u);
    vis[u]=1;
    for(int i=head[u];~i;i=edges[i].nxt)
    {
        int v=edges[i].to;
        if(vis[v])continue;
        dep[v]=dep[u]+1,pa[v]=u,dfs1(v);
    }
}
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        cnt=-1,f=0;
        scanf("%d%d",&n,&m);
        q=ceil(n/2.0)+0.1;
        for(int i=0;i<=n;i++)head[i]=-1,vis[i]=0,vector<int>().swap(g[i]);
        for(int i=1;i<=m;i++)
        {
            int u,v;
            scanf("%d%d",&u,&v);
            addedge(u,v),addedge(v,u);
        }
        dfs1(1);
        if(f)
        {
            puts("PATH");
            printf("%d\n",dep[f]+1);
            while(f)printf("%d ",f),f=pa[f];
            puts("");
        }
        else
        {
            puts("PAIRING");
            printf("%d\n",(q+1)/2);
```

```
            for(int i=0;i<n&&q>0;i++)
            for(int j=0;j+1<g[i].size()&&q>0;j+=2)
            printf("%d %d\n",g[i][j],g[i][j+1]),q-=2;
        }
    }
    return 0;
}
```