A 选择

暂无。

B 签到题!!!

显然最优情况是a石头对b剪刀，a剪刀对b布，a布对b石头，这三种情况相互独立，互不干扰，所以答案就是`min(a石头，b剪刀) + min(a剪刀，b布) + min(a布，b石头)`。

时间复杂度：`O(1)`

C 爬山

思路：

从编号**x**到编号**y**的山，体力变化为：

(h[y]+h[y-1])*(h[y]-h[y-1])+(h[y-1]+h[y-2])*(h[y-1]-h[y-2])+...+(h[x+1]+h[x])*(h[x+1]-h[x])

=h[y]^2-h[y-1]^2+h[y-1]^2+-h[y-2]^2+...+h[x+1]^2+-h[x]^2

=h[y]^2-h[x]^2

关键代码：

```
while(q--) {
    int x,y;
    scanf("%d%d",&x,&y);
    printf("%d\n",h[y]*h[y]-h[x]*h[x]);
}
```

D 混元形意下棋掌门人

这题如果硬想的话应该有很多种思路，这里只讲找规律的做法。

我们设dp[i][j]表示棋子一开始在<i,j>马老师是输还是赢，0表示输，1表示赢，现在已知dp[0][0]为0，那么就可以在小棋盘里(比如20x20)进行必胜必败态转换打表找规律。

时间复杂度:`o(1)`

```
for(int i=0; i=20; ++i){  //打表找规律
        for(int j=0; j<=20; ++j){
            if(i==0 && j==0) continue;
            bool findlose=false;
            if(!dp[i-1][j] && i-1>=0) findlose=true;
            if(!dp[i][j-1] && j-1>=0) findlose=true;
            if(!dp[i-2][j] && i-2>=0) findlose=true;
            if(!dp[i][j-2] && j-2>=0) findlose=true;
            if(findlose) dp[i][j]=1;
        }
    }
    for(int i=0; i<=20; ++i){
        for(int j=0; j<=20; ++j){
            cout<<dp[i][j]<<' ';
        } cout<<'\n';
    }
```

E 跳跳棋

思路:

一维dp,设f[x]为以编号**x**方格为起点的游戏回合数，a[x]为到达**x**方格后下回合要往右前进的步数。

状态转移方程:

若x<=n,f[x]=f[x+a[x]]+1;(当前方格游戏回合数为下一回合所在方格游戏回合数+1)。

若x>n,f[x]=0;(离开棋盘，游戏结束)

期望值计算:

(f[1]+f[2]+...+f[x])/n;

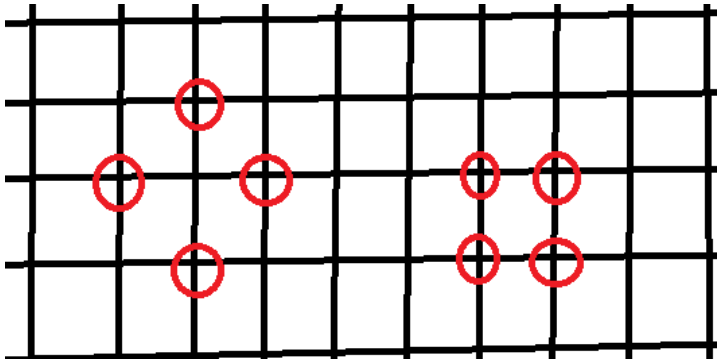
关键代码:

```
ll sum=0;
for(int i=n;i>=1;i--) {
    if(i+a[i]>n) f[i]=1;
    else f[i]=f[i+a[i]]+1;
    sum+=f[i];
}
if(sum-(sum/n)*n<=(sum/n+1)*n-sum) {
    printf("%lld\n",sum/n);
}else printf("%lld\n",sum/n+1);
```

F 导弹袭击

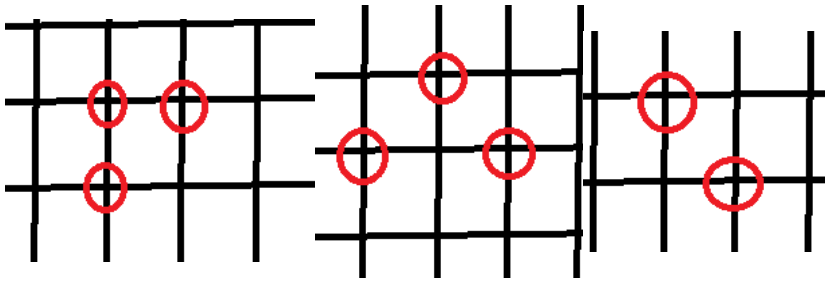
思路：

通过观察发现，能够同时被轰炸到的坐标存在以下两种情况：

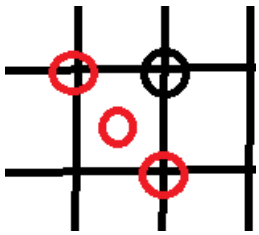


即能被同时轰炸到的坐标必然为以上两种坐标集的子集。

例如：



对于任意一个坐标，以以下三个位置为轰炸中心即可包括上面的两种情况：



所以仅需枚举这些位置，判断能被轰炸到的点的数量，取最大值即可。

关键代码：

```
ll cal(int x1,int y1,int x2,int y2) {
    ll tmpx=(x1-x2),tmpy=(y1-y2);
    return tmpx*tmpx+tmpy*tmpy;
}
```

```
while(T--) {
    int n;
    cnt=0;
    scanf("%d",&n);
    for(int i=1;i<=n;i++) {
        scanf("%d%d",&x[i],&y[i]);
        x[i]*=2,y[i]*=2;
```

```
            _x[++cnt]=x[i]-2,_y[cnt]=y[i];
            _x[++cnt]=x[i],_y[cnt]=y[i]-2;
            _x[++cnt]=x[i]-1,_y[cnt]=y[i]-1;
        }
        int ans=0;
        for(int i=1;i<=cnt;i++) {
            int sum=0;
            for(int j=1;j<=n;j++) {
                if(cal(_x[i],_y[i],x[j],y[j])<=4) sum++;
            }
            ans=max(ans,sum);
        }
        printf("%d\n",ans);
    }
```

G 签到题???

预备知识：深度优先搜索

做法：

1. 数学公式
2. 网络流
3. 全排列枚举所有情况（用深度优先搜索实现）

因为不赢的情况一共就六种<a石头，b石头>,<a石头，b布>,<a剪刀，b剪刀>,<a剪刀，b石头>,<a布，b布>,<a布，b剪刀>，假设他们分别为a,b,c,d,e,f操作，那么他们之间的顺序是无所谓的，比如依次做acbcb操作和依次做accbb是一样的，所以就可以对这六种情况进行全排列，然后分别贪心减掉两两之间的最小值加入答案，维护答案即可。

时间复杂度:`O(6!)`

H 一个人的探险

预备知识：欧拉回路性质，floyd最短路，状压dp

首先对于欧拉回路：所有点的度数都为偶数。因为所有点至少经过一次，那么可以把题意转换成最少加多少条边使得图满足以上结论。
而加边的目的是为了把奇度数转化为偶度数，先floyd一下得到全源最短路。dp[i]表示状态i下度数为偶数的最小花费，因为n<=15,想到状压dp，挑两个奇度数的点转移即可。

时间复杂度:`O(2^n*n*n)`