周赛二题解

A、简单的概率

出题人：颜集源

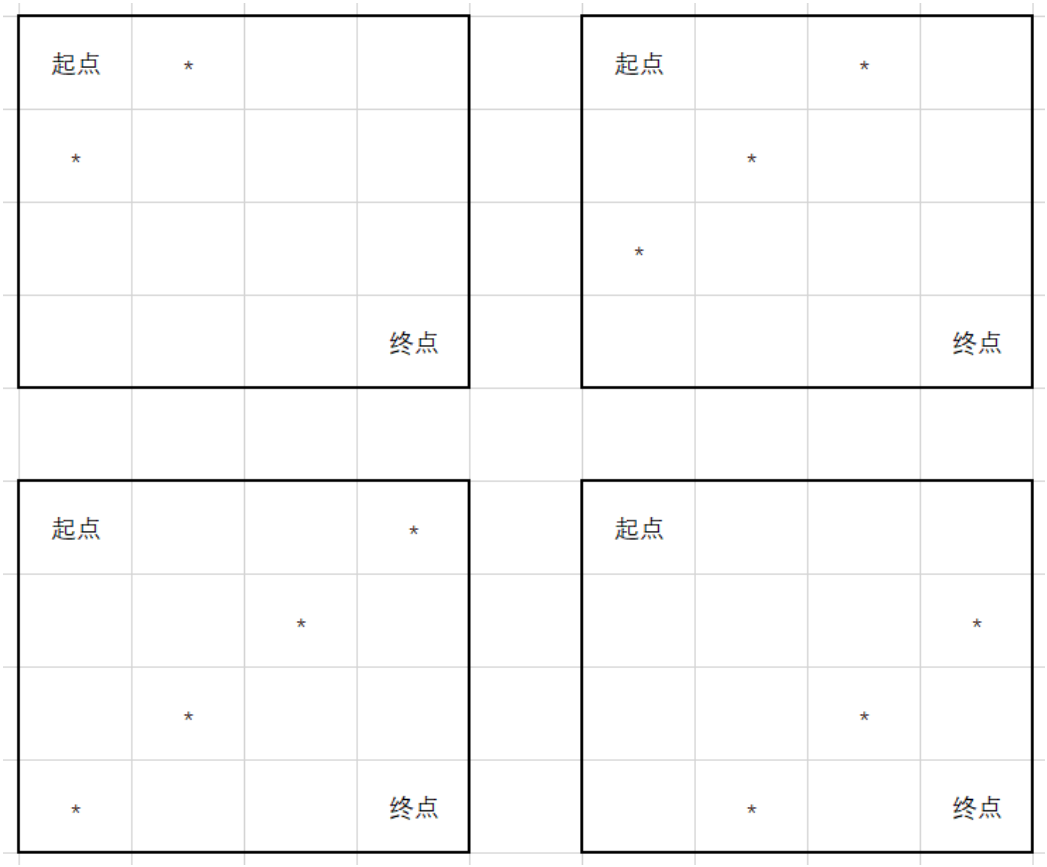
每个数字被抽到的概率是一样多的，所以我们只需要用三重for循环代表三次抽取抽到的数字，然后判断其中大于X的数量有几次即可。

```
#include <stdio.h>
int main()
{
    int x;
    scanf("%d",&x);
    int i,j,k;
    double sum=0;
    for(i=1;i<=100;i++)
        for(j=1;j<=100;j++)
            for(k=1;k<=100;k++)
            {
                if(i+j+k>x) sum++;
            }
    double ans=sum/1000000.0*100;
    printf("%.1lf%%\n",ans);
    return 0;
}
```

B、YJY倒了

出题人：向晨

迷宫是一个方阵，从起点无法到达终点的情况，只有斜边的路完全被堵上的情况。



以4*4的迷宫为例，以上是无法到达终点的情况中的4种。

可以证明，只要斜边不完全被堵死，其他的情况都可以从起点到达终点。

而对于方阵而言，同一斜边满足行数+列数=一个固定数，且每个不同的斜边这个固定数都不同，只要统计每条斜边上的障碍数量，再判断它有没有完全堵死即可。

也可以从一个角落往斜边不断延伸，看看有没有障碍连过去，如果一直连到底，就是堵死了。

```
#include <stdio.h>
int lines[200010]={0};
int getCount(int sum,int n){//获取这个固定数
    return sum>n ? n+n+1-sum : sum-1;
}
int main()
{
    int n,x,i;
    scanf("%d",&n);
    for(i=1;i<=n;i++){
        scanf("%d",&x);
        x+=i;//行+列
        lines[x]++;//统计斜行上障碍数量
        if(lines[x]&&lines[x]>=getCount(x,n)){//斜行完全被堵死了
            printf("NO");
            break;
        }
    }
    if(i>n){
        if(lines[2]||lines[n+n]){//起点和终点有障碍的话
```

```
            printf("NO");
        }
        else printf("YES");
    }
    return 0;
}
```

C、Disco Elysium III

出题人：李泽政

对于每种技能，思维的选择都是独立的。

而哈里最多能装n个技能。

所以对于某个技能，我们将所有的思维按该技能加成的大小降序排序，然后选取前n个的和作为答案。需要注意，技能的加成可能是负数，前n个里我们只要正数的和，负数加成就不要装备了。

```
#include <stdio.h>
#include <string.h>
#define ll long long
//using namespace std;
const int maxn=1e6+10;
char s[105][105];
char in[105];
int add[105][105];
int cnt[105];
int main()
{
    int t,m,n;
    scanf("%d%d%d",&t,&m,&n);
    for(int i=1;i<=t;i++) scanf("%s",s[i]);
    for(int i=1;i<=m;i++)
    {
        int c;
        scanf("%d",&c);
        for(int j=1;j<=c;j++)
        {
            scanf("%s",in);
            for(int k=1;k<=t;k++)
            {
                if(strcmp(in,s[k])==0)
                {
                    int d;
                    scanf("%d",&d);
                    add[k][++cnt[k]]=d;
                    break;
                }
            }
        }
```

```
    }
    for(int i=1;i<=t;i++)
    {
        for(int j=1;j<cnt[i];j++)//冒泡排序
            for(int k=1;k<=cnt[i]-j;k++)
            {
                if(add[i][k]<add[i][k+1])
                {
                    int temp=add[i][k];
                    add[i][k]=add[i][k+1];
                    add[i][k+1]=temp;
                }
            }
        ll ans=0;
        for(int j=1;j<=n;j++)
            if(add[i][j]>0) ans+=add[i][j];
        printf("%lld\n",ans);
    }
    return 0;
}
```

D、方形卡片游戏

出题人：高言峰

如果桌面上有 n 张卡片未被拿开时小G存在必胜的策略，则称此时为一个必胜态；否则如果对手存在必胜的策略，则称此时为必败态。

那么我们考察卡片数为 $n-1, n-4, n-9, \ldots$ 的状态。显然，如果在这之中存在一个必败态，那么此时就是必胜态；而如果它们全都是必胜态，那么此时就是必败态。

例如，0 是必败态，1 因为可以变成必败态 0 所以是必胜态，2 因为只能变成必胜态 1 所以是必败态.....等等。

这可以用递推的方法来做，因为询问较多，建议先 $O(n\sqrt{n})$ 预处理出答案，再 $O(1)$ 查询。

```
#include <stdio.h>
#define MAXN 100005
int dp[MAXN];
int main()
{
    for (int i = 1; i < MAXN; ++i)
        for (int j = 1, k = 1; j <= i; k += 2, j += k)
            if (!dp[i - j])//能转移到必败态
                dp[i] = 1;//那就是必胜态
    int t, n;
    scanf("%d", &t);
    while (t--)
    {
        scanf("%d", &n);
        printf("%s\n", dp[n] ? "Win" : "Lose");
```

```
    }
    return 0;
}
```

E、对称轴的数量

出题人：李嘉龙

因为给的全都是矩形，且保证其两边都与坐标轴平行，所以我们可以确定，最多只存在4条对称轴。

可能的四条对称轴，分别是竖的、横的、两条45°斜对角线的。

所以，读入的时候，我们先得到矩形四个端点，然后由此得到最大和最小的x、y坐标，然后根据此得到四条对称轴的位置。

接下来，对于每条对称轴，我们都去枚举每个点，看看对称轴的另一边能不能找到对称的点，如果都可以，那这条对称轴就是满足条件的，答案数+1。

但是还有几个问题：

1、坐标可能是负数：可以在读入的时候将所有点的x、y坐标都+100，相当于将整个图形往右上角平移。

2、对称轴涉及除以2，但是给的坐标点都是整数，所以对称轴可能是0.5：在读入时将所有点的坐标都先乘2，相当于是等比例扩大，不改变答案。

3、重合的矩形边会消失：所以对于所有多次出现的矩形端点，我们都不需要考虑，因为只要一个端点多次出现，就代表他在多个矩形重合的边上，此时该边已经消失，不需要考虑这个点还有没有对称点。

由此就可以完成这题了。

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int vis[500][500];
struct node{int x;int y;};
struct node p[500];
int max(int x,int y){return x>y?x:y;}
int min(int x,int y){return x<y?x:y;}
int cnt;
void adda(int x,int y)
{
    p[++cnt].x=x;
    p[cnt].y=y;
}
int main()
{
    int t;
    scanf("%d",&t);
```

```
    while(t--)
    {
        memset(vis,0,sizeof(vis));
        cnt=0;
        int maxx=-1e5,maxy=-1e5;
        int minx=1e5,miny=1e5;//初始化
        int n;
        scanf("%d",&n);
        for(int i=1;i<=n;i++)
        {
            int x1,y1,x2,y2;
            scanf("%d%d%d%d",&x1,&y1,&x2,&y2);//读入
            x1*=2;y1*=2;x2*=2;y2*=2;//乘2
            x1+=200;y1+=200;x2+=200;y2+=200;//防止负数
            maxx=max(maxx,x1);maxx=max(maxx,x2);
            maxy=max(maxy,y1);maxy=max(maxy,y2);
            minx=min(minx,x1);minx=min(minx,x2);
            miny=min(miny,y1);miny=min(miny,y2);//找到边角点
            vis[x1][y1]++;
            vis[x1][y2]++;
            vis[x2][y1]++;
            vis[x2][y2]++;//每个点出现的位置
            adda(x1,y1);adda(x1,y2);
            adda(x2,y1);adda(x2,y2);

        }
        int dx=(maxx+minx)/2,dy=(maxy+miny)/2;//对称轴
        int f1=1,f2=1,f3=1,f4=1;//四条对称轴是否可行的标记
        for(int i=1;i<=cnt;i++)
        {
            int nx=p[i].x,ny=p[i].y;
            if(vis[nx][ny]>1) continue;//多次出现的点可以无视
            if(2*dy-ny<0) f1=0;//越界代表对称轴另一边必无这个点，可以直接
判不行
            else f1=f1&&vis[nx][2*dy-ny];
            if(2*dx-nx<0) f2=0;
            else f2=f2&&vis[2*dx-nx][ny];//横轴和竖轴
            if(dx-dy+ny<0||dy-dx+nx<0) f3=0;
            else f3=f3&&vis[dx-dy+ny][dy-dx+nx];//对角线1
            if(dx+dy-ny<0||dx+dy-nx<0) f4=0;
            else f4=f4&&vis[dx+dy-ny][dx+dy-nx];//对角线2
        }
        int ans=0;
        if(f1) ans++;
        if(f2) ans++;
        if(f3) ans++;
        if(f4) ans++;
        printf("%d\n",ans);
    }
    return 0;
}
```

总结：

吸取了周赛一的教训，本次周赛二提高了B题和C题的难度，同时降低了D题的难度，让区分度更加明显。

ABCD四题的AC比例基本都在2:1~3:1内，保证每两三个人就能有一个人尝试下一题。

本次的A题，看似是一题求概率的题目，但很快就能想到等概率，用枚举即可。

B题则需要一定的思维量，相比周赛一的B题要更有难度，代码也更加地难写。

C题则是无需什么思维量的字符串处理，难点集中在复杂繁琐的操作，需要细心和集中的专注力。

D题与其说是动态规划，不如说是递推，实际上并无超纲，认真思考还是能想出来的，但是思维量要比B更上一个台阶。

本次的E题也并未超纲，涉及的几何知识全是大学前所学，难度集中在想到对称轴的数量最多只有4，以及后续的若干小问题处理上，与其说是计算几何，不如说是一道大模拟题。