概率与数学期望

山东省实验中学
宁华

一、概率

- 概率，又称或然率、机会率、机率（几率）或可能性，它是概率论的基本概念。概率是对随机事件发生的可能性的度量，一般以一个在0到1之间的实数表示一个事件发生的可能性大小。越接近1，该事件更可能发生；越接近0，则该事件更不可能发生，其是客观论证，而非主观验证。

基本概念

- 样本点
- 样本空间
- 随机事件
- 随机变量

例

- 罐中有12粒围棋子，其中8粒白子，4粒黑子，从中任取3粒，求取到的都是白子的概率是多少？

- 12粒围棋子从中任取3粒的总取法数是C(12,3)
- 取到3粒的都是白子的情况是C(8,3)

- ∴概率
- C(8,3)
- P = ———————— = 14/55
- C(12,3)

例

- 三人斗地主，大小王在同一家的概率是多少？（一副牌54张：地主20张牌，农民各17张牌）。

方法1

- 假设大小王在地主家：P1 = C(52,18)/C(54,20)
- 假设大小王在某个农民家：P2 = C(52,15)/C(54,17)

- 则大小王在同一家的概率：P = P1 + 2*P2

方法2

- 假设大小王在地主家：P1 = 20/54*19/53
- 假设大小王在某个农民家：P2 = 17/54*16/53

- 则大小王在同一家的概率：P = P1 + 2*P2

方法3

- 54张牌可以看做54个位置，一个位置可以放入一张牌。每一个农民对应17个位置，地主对应17个位置加3个地主牌的位置。
- 大小王占两个位置，总的情况为C(54,2)。
- 则一个农民同时抓到大小王意味着大小王放入该农民对应的17个位置，共C(17,2)。
- 对于地主，大小王可以同时在3张地主牌中，为C(3,2)；可以不在地主牌中,则和农民一样为C(17,2)；可以有一张在地主牌中，另一张在前面的17个位置中，为C(17,1)*C(3,1)。
- 故P=(2*C(17,2) + (C(3,2)+C(17,2)+C(17,1) * C(3,1))) / C(54,2)

例

- 一副扑克54张牌，现在分成3份，每份18张，问大小王出现在同一份中的概率是多少？

方法1：

- 此处不妨假设3份是有序的。

- 扑克牌54张分成3等份，先从54张牌中选18张，再从剩余36张选18张，最后再从18中拿18张，因此共有
- M = C(54, 18) * C(36, 18) * C(18, 18) 种分法。
- 其中大小王出现在同一份中的分法是把两张大小王放在3份中的1份，再从剩余52张中选16张放入大小王所在的那份，再从剩余36张选18张，最后再从18中拿18张，因此共有：
- N = C(3, 1) * C(52, 16) * C(36, 18) * C(18, 18) 种。
- 因此所求概率为: P = N / M = 17 / 53

方法2

- 假设有1、2、3三组，我们先求大王小王在同在1组的概率：
- P1 = C(16,52)/C(54,18)

- 1，2，3组都有可能选择，所以大小王同时出现在一份中的概率为：
- P=3*P1

方法3：

- 设三份分别为A、B、C。
- 大小王之一肯定在其中一份中，假设在A中，概率为1/3。
- A中剩余17张牌中可能含有另一张王，而B、C中的18张牌也可能含有另一张王，因此A份中含有另一张王的概率是17 / (17 + 18 + 18) = 17 / 53
- 也因此可知，A份中同时含有大小王的概率为:
- (1 / 3) * (17 / 53)
- B中C中同时含有大小王的概率同A，因此所求概率为:
- 3 * ((1 / 3) * (17 / 53)) = 17 / 53

方法4：（同方法3，引入条件概率的概念）

- 假设有1、2、3三组，我们先求大王小王在同在1组的概率：
- (1)大王在1组P(B)：1/3
- (2)大王已经在1组的条件下小王在1组P(A|B)：17/53（因为把大王放在1组用掉了一个空位）
- 那么大王小王同在1组的概率P：P(A,B) = P(B)*P(A|B) = 1/3 * 17/53

- 1，2，3组都有可能选择：3 * 18/54 * 17/53 = 17/53
- 所以大小王同时出现在一份中的概率为 17/53。

二、数学期望

- 由来：
- 在17世纪，有一个赌徒向法国著名数学家帕斯卡挑战，给他出了一道题目：甲乙两个人赌博，他们两人获胜的机率相等，比赛规则是先胜三局者为赢家，赢家可以获得100法郎的奖励。当比赛进行到第四局的时候，甲胜了两局，乙胜了一局，这时由于某些原因中止了比赛，那么如何分配这100法郎才比较公平？

- 用概率论的知识，不难得知，甲获胜的可能性大，甲赢了第四局，或输掉了第四局却赢了第五局，概率为1/2+(1/2)*(1/2)=3/4。分析乙获胜的可能性，乙赢了第四局和第五局，概率为(1/2)*(1/2)=1/4。由此引出了甲的期望所得值为100*3/4=75法郎，乙的期望所得值为100*1/4=25法郎。这个故事里出现了“期望”这个词，数学期望由此而来。

例子

- 假设有1000根钢筋，其中200根长0.98米；300根长0.99米；400根长1米；100根长1.01米，求其平均长度。

例子

- 解：

$$\frac{0.98\times200+0.99\times300+1\times400+1.01\times100}{1000}$$

$$=0.98\times\frac{200}{1000}+0.99\times\frac{300}{1000}+1\times\frac{400}{1000}+1.01\times\frac{100}{1000}$$

- 而200/1000、300/1000、400/1000、100/1000分别是从1000根钢筋中，任意抽取一根，它的尺寸分别为0.98、0.99、1、1.01的概率。
- 由此可以引出下面的随机变量的数学期望的定义。

数学期望的定义

- 离散型随机变量：如果随机变量只取得有限个值或无穷个能按一定次序一一列出的值，其值域为一个或若干个有限或无限区间，这样的随机变量称为离散型随机变量。

- 离散型随机变量的一切可能的取值Xi与对应的概率p(Xi)乘积之和称为该离散型随机变量的数学期望(若该求和绝对收敛），记为E(X)，即：

$$E(X)=\sum_{k=1}^{+\infty}x_k p_k$$

数学期望的本质

- 数学期望的本质——加权平均。
- 它是简单算术平均的一种推广，**它是一个数不再是随机变量。**

公式

- 离散型随机变量X的取值为X1,X2,...,Xn,
- p(X1),p(X2),...,p(Xn)为X对应取值的概率，可理解为数据X1,X2,...,Xn出现的频率f(Xi)，则：

$$E(X) = X1 * p(X1) + X2 * p(X2) + \ldots + Xn * p(Xn)$$
$$= X1 * f(X1) + X2 * f(X2) + \ldots + Xn * f(Xn)$$

例

- 某城市有10万个家庭，没有孩子的家庭有1000个，有一个孩子的家庭有9万个，有两个孩子的家庭有6000个，有3个孩子的家庭有3000个。
- 则此城市中任一个家庭中孩子的数目是一个随机变量，记为X。它可取值0，1，2，3。
- 其中，X取0的概率为0.01，取1的概率为0.9，取2的概率为0.06，取3的概率为0.03。
- 则，X的数学期望

$$E(X)=0*0.01+1*0.9+2*0.06+3*0.03=1.11$$

- 即此城市一个家庭平均有小孩1.11个。

例

- 甲、乙两选手进行打靶，击中环数为X_1，X_2，它们的分布律为

X_1	7	8	9	10
p_k	0.2	0.3	0.4	0.1

X_2	7	8	9	10
p_k	0.3	0.5	0.1	0.1

- 试评定他们的成绩好坏。

- 解：我们计算两者的数学期望，可知
- $E(X_1) = 7 \times 0.2 + 8 \times 0.3 + 9 \times 0.4 + 10 \times 0.1$ = 8.4
- $E(X_2) = 7 \times 0.3 + 8 \times 0.5 + 9 \times 0.1 + 10 \times 0.1$ = 8.0
- 即甲的成绩略好于乙的成绩。

例

- 按规定，某车站每天 8:00 ~ 9:00, 9:00 ~ 10:00 都恰有一辆客车到站，但到站的时刻是随机的，且两者到站的时间相互独立。其规律为：

到站时刻	8:10 9:10	8:30 9:30	8:50 9:50
概率	$\frac{1}{6}$	$\frac{3}{6}$	$\frac{2}{6}$

- 问题：
- (i) 一旅客 8:00 到车站，求他候车时间的数学期望。
- (ii) 一旅客 8:20 到车站，求他候车时间的数学期望。

到站时刻	8:10 9:10	8:30 9:30	8:50 9:50
概率	$\frac{1}{6}$	$\frac{3}{6}$	$\frac{2}{6}$

- 解
- (i) 一旅客 8:00 到车站，求他候车时间的数学期望。
- 候车时间X的分布律为

X	10	30	50
P_k	$\frac{1}{6}$	$\frac{3}{6}$	$\frac{2}{6}$

- 候车时间的数学期望为：$E(X)=10\times\frac{1}{6}+30\times\frac{3}{6}+50\times\frac{2}{6}$ $=33.33$(分钟).

到站时刻	8:10 9:10	8:30 9:30	8:50 9:50
概率	$\frac{1}{6}$	$\frac{3}{6}$	$\frac{2}{6}$

- 解
- (ii) 一旅客 8:20 到车站，求他候车时间的数学期望。
- 候车时间X的分布律为

X	10	30	50	70	90
p_k	$\frac{3}{6}$	$\frac{2}{6}$	$\frac{1}{6}\times\frac{1}{6}$	$\frac{1}{6}\times\frac{3}{6}$	$\frac{1}{6}\times\frac{2}{6}$

- 候车时间的数学期望为：

$$E(X)=10\times\frac{3}{6}+30\times\frac{2}{6}+50\times\frac{1}{6}\times\frac{1}{6}+70\times\frac{1}{6}\times\frac{3}{6}+90\times\frac{1}{6}\times\frac{2}{6}\ =27.22(\text{分钟}).$$

例 BSOJ 3851 计算概率

- 题目描述
- 小明有n个长度不一的小木棍，这些木棍的长度都是正整数。小明的父亲想和小明做一个游戏。他规定一个整数长度l，让小明闭着眼睛从n个木棍中随便拿出两个。如果两个木棍的长度总和小于等于l，则小明胜，否则小明的父亲胜。小明想知道他胜出的概率究竟有多大。

- 输入
- 输入包含两行。第一行为两个整数n和l，其中n和l都不超过100000。第二行包含n个整数，分别为n个木棍的长度。
- 输出
- 输出包含一个实数，小明胜出的概率，保留两位小数。
- 样例输入
- 4 5
- 1 2 3 4
- 样例输出
- 0.67

```
sort(a+1,a+1+n);
int ans=0;
for(int i=2;i<=n;i++)
{
	int t=upper_bound(a+1,a+i,l-a[i])-a;
	ans+=t-1;
}
printf("%.2f\n",2.0*ans/n/(n-1));
```

```
•	sort(a+1,a+1+n);
•	int ans=0;
•	for(int i=1;i<n;i++)
•	{
•		int t=upper_bound(a+i+1,a+n+1,l-a[i])-a;
•		ans+=t-1-i;
•	}
•	printf("%.2f\n",2.0*ans/n/(n-1));
```

双指针

双指针

- sort(a+1,a+n+1);
- int i=1,j=n;
- while(i<j)
- {
-     if(a[i]+a[j]<=l)
-     {
-         cnt+=j-i;
-         i++;
-     }
-     else j--;
- }
- long long tot=n*(n-1)/2;
- printf("%.2f\n",(double)cnt/tot);

超级蚯蚓

- 题目限制
- 1000 ms 128 M
- 题目描述
- 生物学家们利用基因工程制造了一种超级蚯蚓，与原品种可以一分为二的特性相反，我们将两条这种超级蚯蚓的头或尾端接触，他们的头或尾会连接起来。
- 实验室中现在有n条这样的超级蚯蚓，现在重复n次以下操作：随机抽出两条超级蚯蚓，使它们的头或尾接触。可以想象，这样n次之后將不再有条状蚯蚓，n条超级蚯蚓连接成了一些环。那么有多大概率刚好所有这些超级蚯蚓只形成了一个环？

- 输入格式
- 仅一行，包含一个整数n (2<=n<=1000)。
- 输出格式
- 输出一行，为刚好成环的概率。
- 输入样例
- input example1:
- 2
- input example2:
- 5
- input example3:
- 3

- 输出样例
- output example1:
- 0.666667
- output example2:
- 0.406349
- output example3:
- 0.533333
- 样例解释
- 假设n=2,有2条超级蚯蚓，它们共有四个头/尾端，假设编号为ABCD，那么第一次选择AB或者CD，最终不能成一个环，除此之外选择AC AD BC BD，最终都能成一个环，成环概率为4/(2+4)=2/3=0.666667

- 数据范围
- 对于25%的数据，2<=n<=10
- 对于50%的数据，2<=n<=100
- 对于100%的数据，2<=n<=1000

题解

- 最后只得到一个环，当且仅当（除最后一次外）每一次合并操作都不会出现环。
- 当有n条非环形蚯蚓时，合并出现一个环的概率为1/(2n-1)，则不会出现一个环的概率为 1 − 1/(2n-1) 。
- 记f(n)为n条非环状蚯蚓合并后只有一个环的概率，则有：
- f(n) = f(n-1) * (1-1/(2n-1))   n>1
- f(1) = 1

排队

- 【题目描述】
- 在成都某中学有m个男生与n个女生排队，这个学校的女生比较古怪，从某个位置（包含这个位置）开始往前数，男生的数量超过了女生的数量，女生会感觉不安全，于是会大叫起来，为了构建和谐校园，安排队伍时应该避免这样的情况。请你计算出不会引发尖叫的排队方案的概率。（排队方案不同定义：当且仅当某个某个位置人不一样，如男生A、男生B ，与男生B、男生A ，2个排列是不同方案）
- 【输入格式】
- 第一行1个整数, 表示测试数据的组数。
- 每个数据 有两个数 N,M（N个女生，M个男生）
- 【输出格式】
- 对于每组数据,输出一个实数（保留到小数点后 6 位）

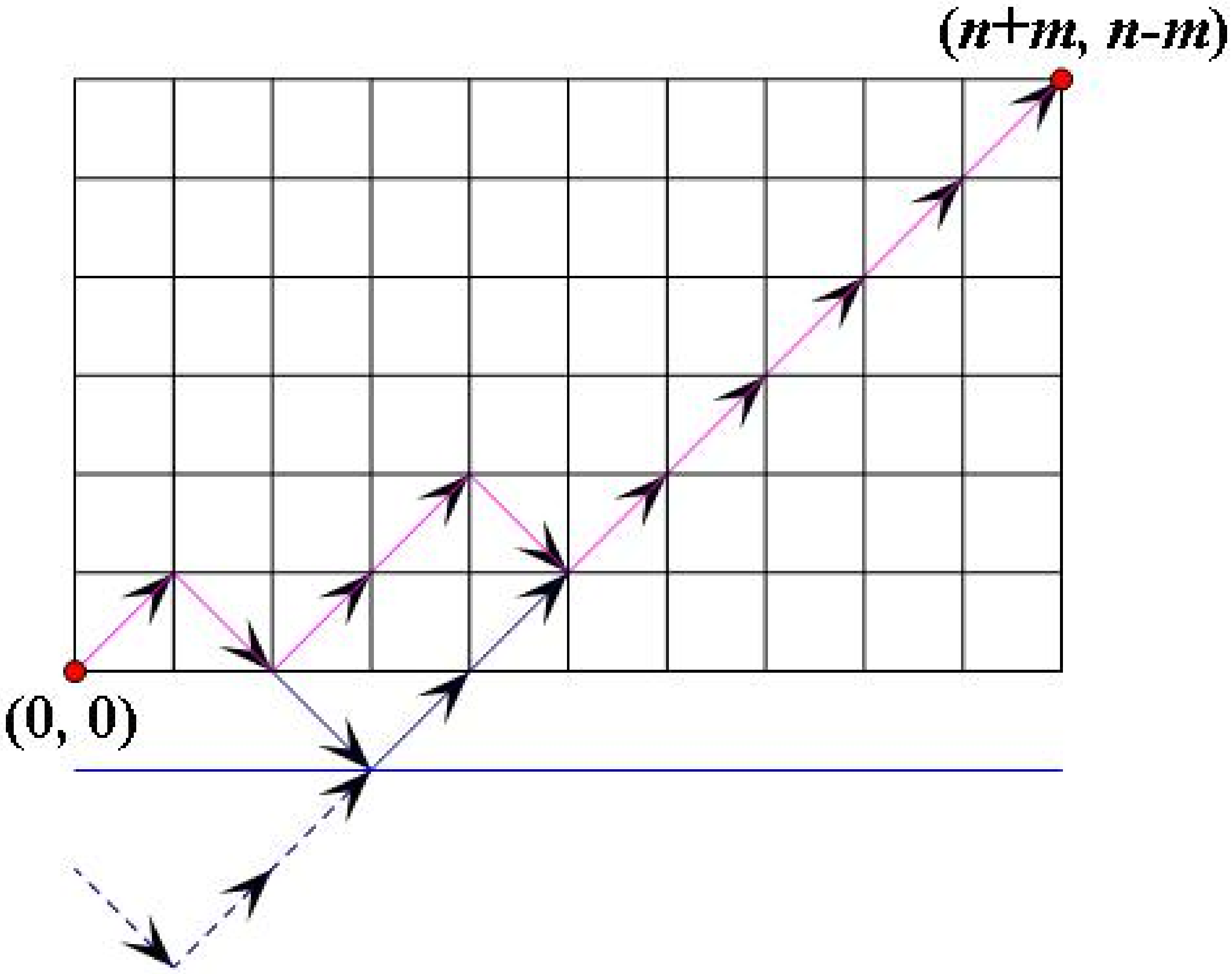
- 【样例输入】
- 3
- 1 0
- 0 1
- 1 1
- 【样例输出】
- 1.000000
- 0.000000
- 0.500000
- 【备注】
- 30%的数据： (测试组数<=10),(0<=N,M<=1000).
- 100%的数据： (测试组数=9008)， (0<=N,M<=20000).

题目分析

- 这个题可以考虑数形结合的思想。
- 将问题看作在二维平面上行走。加入一个女生看成移动 (1, 1)，加入一个男生看成移动 (1, -1)，那么最后落在 (n+m, n-m) 处。

- 所以在移动过程中，一旦纵坐标小于0，那么就一定对应一种非法方案，这时可将其看做从 (0, -2) 出发到 (n+m, n-m) 处，那么向上就多走了一步，向下少走了一步，最后合法方案数即为C(m+n, m) - C(m+n, m-1)，最后答案就是 1 - m/(n+1)。（注意答案最后可能小于0，所以需取ans与0二者较大值）

[SCOI2010] 生成字符串



一类关于概率的dp问题

- 概率dp

例 火山喷发 2000ms 262144K

- https://nanti.jisuanke.com/t/16862
- 火山喷发对所有附近的生物具有毁灭性的影响。在本题中，我们希望用数值来模拟这一过程。
- 在环境里有n个生物分别具有A1,A2,… ,An点生命值，一次火山喷发总计M轮，每轮造成1点伤害，等概率地分给所有存活的生物，即如果目前有K个活着的生物，每个生物受到这点伤害的概率是1/K。如果一个生物的生命值减为0，它会立即死去，此后都不会再占用受到伤害的概率。如果没有生物存活，那么将没有生物会受到伤害。
- 现在你的任务是，给定n,M和全部生物的生命值，问每个生物火山喷发后依然存活的概率。

- 输入格式
- 第一行两个正整数n和M。
- 第二行n个正整数A1,...,An。
- 输出格式
- n行，第i行一个数表示第i个生物存活下来的概率，保留小数点后六位。
- 数据范围与约定
- 对于10%的数据 N=1。
- 对于30%的数据 N=2。
- 对于全部数据 N ≤4，M≤120, Ai≤50。

- 样例输入1
- 1 2
- 1
- 样例输出1
- 0.000000
- 样例输入2
- 3 15
- 2 12 2
- 样例输出2
- 0.001684
- 0.996632
- 0.001684

- 注意到最多只有4种生物
- 设 f[i][j][k][p] 表示生命值为 i，j，k，p 时的概率

- 目标?
- 比如：
- 求第一种生物最后存活的概率
- ∑ f[x][j][k][p] 1<=x<=a[1], 0<=j<=a[2], 0<=k<=a[3], 0<=p<=a[4]
- 或者：
- 1 - 第一种生物不存活的概率
- = 1 - ∑ f[0][j][k][p] 0<=j<=a[2], 0<=k<=a[3], 0<=p<=a[4]

```
f[a[1]][a[2]][a[3]][a[4]]=1;
for (l=0;l<=m;l++)
{
    for (i=0;i<=a[1];i++)
    {
        for (j=0;j<=a[2];j++)
        {
            for (k=0;k<=a[3];k++)
            {
                int p=a[4]-(l-(a[1]-i)-(a[2]-j)-(a[3]-k));
                if (p<0||p>a[4]) continue;
                if (f[i][j][k][p]==0) continue;
                cnt=0;
                if (i) cnt++;if (k) cnt++;
                if (j) cnt++;if (p) cnt++;
                if (cnt==0) continue;
                if (i) f[i-1][j][k][p]+=f[i][j][k][p]/cnt;
                if (j) f[i][j-1][k][p]+=f[i][j][k][p]/cnt;
                if (k) f[i][j][k-1][p]+=f[i][j][k][p]/cnt;
                if (p) f[i][j][k][p-1]+=f[i][j][k][p]/cnt;
            }
        }
    }
}
```

NOIP 2016 换教室

- Description
- 对于刚上大学的牛牛来说,他面临的第一个问题是如何根据实际情况申请合适的课程。在可以选择的课程中,有2n节课程安排在n个时间段上。在第i(1≤i≤n)个时间段上,两节内容相同的课程同时在不同的地点进行,其中,牛牛预先被安排在教室ci上课,而另一节课程在教室di进行。在不提交任何申请的情况下,学生们需要按时间段的顺序依次完成所有的n节安排好的课程。如果学生想更换第i节课程的教室,则需要提出申请。若申请通过,学生就可以在第i个时间段去教室di上课,否则仍然在教室ci上课。由于更换教室的需求太多,申请不一定能获得通过。通过计算,牛牛发现申请更换第i节课程的教室时,申请被通过的概率是一个已知的实数ki,并且对于不同课程的申请,被通过的概率是互相独立的。学校规定,所有的申请只能在学期开始前一次性提交,并且每个人只能选择至多m节课程进行申请。这意味着牛牛必须一次性决定是否申请更换每节课的教室,而不能根据某些课程的申请结果来决定其他课程是否申请;牛牛可以申请自己最希望更换教室的m门课程,也可以不用完这m个申请的机会,甚至可以一门课程都不申请。因为不同的课程可能会被安排在不同的教室进行,所以牛牛需要利用课间时间从一间教室赶到另一间教室。

- 牛牛所在的大学有v个教室,有e条道路。每条道路连接两间教室,并且是可以双向通行的。由于道路的长度和拥堵程度不同,通过不同的道路耗费的体力可能会有所不同。当第i(1≤i≤n-1)节课结束后,牛牛就会从这节课的教室出发,选择一条耗费体力最少的路径前往下一节课的教室。现在牛牛想知道,申请哪几门课程可以使他因在教室间移动耗费的体力值的总和的期望值最小,请你帮他求出这个最小值。

- Input
- 第一行四个整数n,m,v,e。n表示这个学期内的时间段的数量;m表示牛牛最多可以申请更换多少节课程的教室;
- v表示牛牛学校里教室的数量;e表示牛牛的学校里道路的数量。
- 第二行n个正整数,第i(1≤i≤n)个正整数表示c,,即第i个时间段牛牛被安排上课的教室;保证1≤ci≤v。
- 第三行n个正整数,第i(1≤i≤n)个正整数表示di,即第i个时间段另一间上同样课程的教室;保证1≤di≤v。
- 第四行n个实数,第i(1≤i≤n)个实数表示ki,即牛牛申请在第i个时间段更换教室获得通过的概率。保证0≤ki≤1。
- 接下来e行,每行三个正整数aj,bj,wj,表示有一条双向道路连接教室aj,bj,通过这条道路需要耗费的体力值是Wj;
- 保证1≤aj,bj≤v,1≤wj≤100。
- 保证1≤n≤2000,0≤m≤2000,1≤v≤300,0≤e≤90000。
- 保证通过学校里的道路,从任何一间教室出发,都能到达其他所有的教室。
- 保证输入的实数最多包含3位小数。

- Output
- 输出一行,包含一个实数,四舍五入精确到小数点后恰好2位,表示答案。你的
- 输出必须和标准输出完全一样才算正确。
- 测试数据保证四舍五入后的答案和准确答案的差的绝对值不大于4*10^-3。(如果你不知道什么是浮点误差,这段话
- 可以理解为:对于大多数的算法,你可以正常地使用浮点数类型而不用对它进行特殊的处理)

- Sample Input
- 3 2 3 3
- 2 1 2
- 1 2 1
- 0.8 0.2 0.5
- 1 2 5
- 1 3 3
- 2 3 1

- Sample Output
- 2.80

- 设 f[i][j][0/1] 表示 前 i 节课共提出 j 个申请，第i节课是否申请 的最少的体力值的期望。
- 例如 f[5][3][1] 表示 前 5 节课共提出 3 个申请，第5节课申请换教室
- 目标：
- `double ans=f[n][0][0];`
- `for(int i=1;i<=m;i++)`
- `ans=min(ans,min(f[n][i][0],f[n][i][1]));`
- `printf("%.2lf",ans);`

- 那么，状态是如何转移的呢？

```
if(第i节课不申请)
{
    if(第i-1节课申请)
    {
        if(第i-1节课申请失败)
            f[i][j][0]=f[i-1][j][1]+dist[c[i-1]][c[i]]*(1.0-p[i-1]);  /*上一个原教室到当前原教室的距离*失败的几率*/
        else
            f[i][j][0]=f[i-1][j][1]+dist[d[i-1]][c[i]]*p[i-1];  /*上一个更换的教室到当前原教室的距离*成功的几率*/
    }
    else
        f[i][j][0]=f[i-1][j][0]+dist[c[i-1]][c[i]];  /*上一个原教室到当前原教室的距离*/
}
else
{
    if(第i节课申请成功)
    {
        if(第i-1节课申请)
        {
            if(第i-1节课申请成功)
                f[i][j][1]=f[i-1][j-1][1]+dist[d[i-1]][d[i]]*p[i]*p[i-1];  /*上一个更换的教室到当前更换的教室的距离*都成功的几率*/
            else
                f[i][j][1]=f[i-1][j-1][1]+dist[c[i-1]][d[i]]*p[i]*(1-p[i-1]);  /*上一个原教室到当前更换的教室的距离*上一个失败，当前成功的几率*/
        }
        else
            f[i][j][1]=f[i-1][j-1][0]+dist[c[i-1]][d[i]]*p[i];  /*上一个原教室到当前更换的教室的距离*当前成功的几率*/
    }
    else
    {
        if(第i-1节课申请)
        {
            if(第i-1节课申请成功)
                f[i][j][1]=f[i-1][j-1][1]+dist[d[i-1]][c[i]]*(1-p[i])*p[i-1];  /*上一个原教室到当前更换的教室的距离*上一个成功，当前失败*/
            else
                f[i][j][1]=f[i-1][j-1][1]+dist[c[i-1]][c[i]]*(1-p[i])*(1-p[i-1]);  /*上一个原教室到当前更换的教室的距离*都失败的几率*/
        }
        else
            f[i][j][1]=f[i-1][j-1][0]+dist[c[i-1]][c[i]]*(1-p[i]);  /*上一个原教室到当前更换的教室的距离*当前失败的几率*/
    }
}
```

```
memset(f,0x3f,sizeof(f));
f[1][0][0]=f[1][1][1]=0;
for(int i=2;i<=n;i++)
{
    f[i][0][0]=f[i-1][0][0]+dist[c[i-1]][c[i]];
    for(int j=1;j<=min(m,i);j++)
    {
        // 这节不申请 求f[i][j][0]
        f[i][j][0]=f[i-1][j][1]+dist[c[i-1]][c[i]]*(1.0-p[i-1])+dist[d[i-1]][c[i]]*p[i-1]; //这节不申请 上节申请
        f[i][j][0]=min(f[i][j][0],f[i-1][j][0]+dist[c[i-1]][c[i]]);  // 这节不申请 上节也不申请

        // 这节申请 求f[i][j][1]
        // 上节不申请
        f[i][j][1]=f[i-1][j-1][0]+dist[c[i-1]][c[i]]*(1.0-p[i])+dist[c[i-1]][d[i]]*p[i]; // 上节不申请 这节申请
        // 上节申请
        double t=f[i-1][j-1][1]+dist[c[i-1]][c[i]]*(1.0-p[i-1])*(1.0-p[i]);  // 这节申请不成 上节申请不成
        t+=dist[d[i-1]][c[i]]*p[i-1]*(1.0-p[i]);  // 这节申请不成 上节申请成
        t+=dist[c[i-1]][d[i]]*(1.0-p[i-1])*p[i];  // 这节申请成 上节申请不成
        t+=dist[d[i-1]][d[i]]*p[i-1]*p[i];          // 这节申请成 上节申请成
        f[i][j][1]=min(f[i][j][1],t);
    }
}
```