**#1. 交易资金**

**题目背景**

小 L 想要收购小 F 的原神账号。

**题目内容**

小 L 不想花太多钱，于是他决定让小 F 去猜他想出多少钱。

如果小 F 猜的钱大于答案，小 L 就会认为小 F 贪得无厌，并将预算减少 k 。

当然，小 L 的预算不会小于 0 ，减到 0 就不减少了。

如果小 F 猜的钱小于等于答案，小 L 就会同意这个价格。

如果小 L 初始的预算在 [1,*n*] 之间随机，问在最优策略下，小 F 能得到多少钱？

**输入格式**

两个整数 *n*,*k* ，含义如题目描述。

**输出格式**

一个整数 *ans* ，表示小 F 最多能期望得到多少钱。

可以证明 *ans*×*n* 是一个整数，所以你只需要输出 *ans*×*n* 。

**样例 1 输入**

1145 191

**样例 1 输出**

378490

**样例 2 输入**

5 3

**样例 2 输出**

9

**子任务**

| **子任务名** | **评分方式** | **时间限制** | **内存限制** | **说明** | **分数** |
| --- | --- | --- | --- | --- | --- |
| 默认子任务 | 求和 | 1000 ms | 512 MB | 共 10 个测试点 | 100 |

**提示**

对于 30% 的数据，*n*≤8

对于 60% 的数据，*n*≤3000

对于另外 10% 的数据，*k*=0

对于另外 20% 的数据，*k*=*n*

对于 100% 的数据，*k*≤*n*≤5×106,*n*≥1,*k*≥0

**#2. 异或**

**题目内容**

给定一颗大小为 *n* 的树，你需要给每个点赋一个 [0,*m*] 中的整数权值，满足对于任意一个点，删掉这个点以及与它相邻的边后，形成的若干个连通块内的异或和两两不等。

**输入格式**

第一行两个数 *n*,*m* ，表示树的节点数量和权值的上界。

接下来 *n*−1 行，每行两个数 *u*,*v* ，表示一条树上的边。

**输出格式**

一行 *n* 个整数，依次表示节点 1 到节点 *n* 的权值。

如果有多个解，输出任意可行解即可。

**样例 1 输入**

5 10

1 2

2 3

3 4

4 5

**样例 1 输出**

1 2 4 8 10

**子任务**

| **子任务名** | **评分方式** | **时间限制** | **内存限制** | **说明** | **分数** |
| --- | --- | --- | --- | --- | --- |
| 默认子任务 | 求和 | 1000 ms | 512 MB | 共 10 个测试点 | 100 |

**提示**

对于 30% 的数据，*m*≥1018,*n*≤5000

对于 60% 的数据，*m*≥4×*n*,*n*≤5000

对于 70% 的数据，*m*≥2×*n*

对于另外 20% 的数据，*n*≤8

对于 100% 的数据， *n*≤106,*n*−2≤*m*≤263−1

**#3. 机房惨案**

**题目背景**

快要NOIP了！小 L 复习了一下 dijkstra 的模板，然后出去上厕所了。

这时小 F 发现他没锁电脑，于是把他的模板稍微修改了一下。现在他想知道这份新的程序通过样例数据的概率是多少。

**题目内容**

小L正在解决的题目是需要求出一个有向图中从 1 号点到其他每个点的最短路径，即单源最短路。

小 F 把代码修改成了每次随机选取一个未扩展过的点进行扩展，即钦定一个 1∼*n* 的排列一次扩展。显然这个算法是有误的。

这是小 F 修改过后的代码：

vector<pair<int,int>> G[N];//vector存图，first表示边的终点，second表示边权

bool vis[N];//当前节点是否已经扩展过

int dis[N];//答案数组，dis[i]表示从 1 到 i 的最短路

int p[N];//排列

void min\_path()

{

memset(dis,0x3f,sizeof(dis));

dis[1]=0;

for(int i=1;i<=n;++i)p[i]=i;

random\_shuffle(p+1,p+n+1);

for(int i=1;i<=n;++i)

{

int nw=p[i];

vis[nw]=1;

for(auto &[v,w]:G[nw])

if(dis[v]>dis[u]+w)

dis[v]=dis[u]+w;

}

}

请你求出在所有排列中，有多少排列能求出正确结果。

**输入格式**

第一行一个整数 *n* ，表示节点个数

接下来 *n* 行，每行 *n* 个数。第 *i* 行 *j* 列的数 *ai*,*j*​ 表示 *i*→*j* 这条有向边的长度。

保证 *ai*,*i*​=0

**输出格式**

一个整数，表示答案。

**样例 1 输入**

4

0 810 2 3

2 0 514 3

2 1919 0 1

3 114 1 0

**样例 1 输出**

12

**子任务**

| **子任务名** | **评分方式** | **时间限制** | **内存限制** | **说明** | **分数** |
| --- | --- | --- | --- | --- | --- |
| 默认子任务 | 求和 | 1500 ms | 1024 MB | 共 10 个测试点 | 100 |

**提示**

对于 100% 的数据，*n*≤15，*ai*,*j*​≤10000

| **测试点编号** | **特殊性质** |
| --- | --- |
| 1∼2 | *n*≤7 |
| 3∼5 | *n*≤10 |
| 6∼7 | *ai*,*j*​≤1 |
| 8∼10 | 无 |



