День первый (15.02.2014 г.) Контест Евгения Капуна

Об авторе...

Капун Евгений Дмитриевич, родился 2 октября 1989 года в Санкт-Петербурге, в 2006 году закончил ФТШ, а в 2012 году стал магистром кафедры компьютерных технологий факультета информационных технологий и программирования НИУ ИТМО.

Основные достижения:

- 1 место на финале ACM ICPC 2009 и 2012 (в составе команды ИТМО).
- 4 место на ACM NEERC 2009 и 2010, 1 место на ACM NEERC 2011 (в составе команды ИТМО).
- 2 место на финале Russian Code Cup 2011, 5 место на финале Russian Code Cup 2012.
- 2 место на финале Яндекс.Алгоритма 2013.
- 1 место на Чемпионате Урала 2009, 2010 и 2012 (в составе команды ИТМО).
- 1 место на Открытой Всесибирской Олимпиаде по программированию имени И. В. Потоссина 2008, 2009 и 2010 (в составе команды ИТМО).
- 2 место на турнире по программированию ICL 2011 (в составе команды ИТМО).
- 3 место на чемпионате по программированию КРОК-2013.
- 4 место в V открытом кубке имени Е. В. Панкратьева по программированию и 3 место в VII, VIII, X и XI кубках (в составе команды ИТМО), 2 место в XIII кубке (в составе команды Petr Team).

Теоретический материал. Теория вероятностей

Определения

Теория вероятностей рассматривает некоторый случайный процесс (или совокупность случайных процессов), называемый опытом. Возможные результаты этого опыта называются исходами. Множество исходов (обычно его обозначают буквой Ω) — множество, на котором определена мера, такая, что мера всего множества равна 1. Если говорить простым языком, то мера — это функция, которая сопоставляет некоторый неотрицательный вес каждому элементу множества, а за меру подмножества принимается сумма мер его элементов. Мера каждого исхода называется его вероятностью и указывает, насколько ожидаем тот или иной исход.

Событие — подмножество множества исходов. Вероятность события — его вес. Вероятность события A обозначается $P(A)$.

Пример

Пусть множество исходов состоит из двух элементов, и каждому из них соответствует вероятность $\frac{1}{2}$. Тогда существует четыре различных события: пустое подмножество, два подмножества из одного элемента и всё множество. В элементарной теории вероятностей часто встречаются конечные множества исходов, в которых все исходы равновероятны.

Сведение задач по теории вероятностей к задачам по комбинаторике

Часто задачи по теории вероятностей можно свести к задачам по комбинаторике. Обычно в этих задачах требуется найти вероятность того, что результат некоторого случайного выбора, имеющего несколько равновероятных исходов, принадлежит некоторому подмножеству. В этом случае ответом будет размер подмножества, делёный на размер всего множества возможных исходов.

Пример. Задача: найти вероятность того, что 10 наугад выбранных чисел от 1 до 10 будут различными. Решение: всего существует 10^{10} способов выбрать 10 чисел от 1 до 10, и при выборе наугад вероятности всех этих способов совпадают. Из них, в 10! способах все числа будут различными. Ответ: $\frac{10!}{10^{10}}$.

Вероятностная динамика

При решении задач по теории вероятностей часто используется динамика, в которой значением является вероятность (или несколько вероятностей). Будем называть такую динамику вероятностной. Обычно в таких задачах переходы между состояниями необратимы (пройдя состояние, нельзя в него вернуться), а в итоге происходит переход или в “хорошее” состояние (вероятность 1), или в “плохое” (вероятность 0).

Такие задачи удобно решать с конца, при этом значением динамики в некотором состоянии является некоторая функция $P(x)$ (вероятность события x), если начинать из этого состояния. Чтобы посчитать это значение для некоторого состояния, нужно знать его для всех состояний, в которые из него можно перейти. Формула обычно имеет вид $D_i = \sum_j P_{ij} D_j$, где D_i — значения динамики, а P_{ij} — вероятности перехода.

Примером использования вероятностной динамики является такая игра: в каждом раунде первый игрок выигрывает с вероятностью p, а второй — с вероятностью $q = 1 - p$. Если выигрывает первый игрок, то второй отдаёт ему одну монету, если же выигрывает второй, то монету отдаёт первый игрок. Игра заканчивается, когда у одного из игроков не остаётся денег. Если у первого игрока изначально n_1 монет, а у второго n_2, то вероятность выигрыша первого игрока равна $\frac{p^{n_1+n_2}-p^{n_2}q^{n_1}}{p^{n_1+n_2}-q^{n_1+n_2}}$, а второго — $\frac{q^{n_1+n_2}-p^{n_2}q^{n_1}}{q^{n_1+n_2}-p^{n_1+n_2}}$. Этот результат известен как теорема Гюйгенса (кстати, если $p = q = \frac{1}{2}$, то всё намного проще — вероятности равны $\frac{n_1}{n_1+n_2}$ и $\frac{n_2}{n_1+n_2}$ соответственно).

Чтобы доказать это, можно воспользоваться вероятностной динамикой, однако, в отличии от обычного случая, здесь переходы обратимы, поэтому просто вычислить вероятности по порядку не получится. Вместо этого вероятностная динамика даёт систему линейных уравнений, которые можно решить. Таким образом, если переходы необратимы, значения можно просто вычислять друг за другом, а если обратимы, то придётся решать систему уравнений.

Условные вероятности. Формула Байеса

Условная вероятность — это вероятность того, что произошло некоторое событие, но только если рассматривать исходы, при которых произошло какое-то другое событие. Формально $P(A|B)$ — условная вероятность того, что событие A произошло при условии события B, — равна $\frac{P(A\cap B)}{P(B)}$. Чтобы это определение было осмысленным, необходимо, чтобы вероятность события B была ненулевой.

Как следствие,

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A).$$

Если $\bigsqcup_i A_i = \Omega$ — разбиение множества исходов на события A_i, то $P(B) = \sum_i P(A_i \cap B) = \sum_i P(B|A_i)P(A_i)$. Отсюда следует, что $P(A_i|B) = \frac{P(A_i\cap B)}{P(B)} = \frac{P(B|A_i)P(A_i)}{\sum_j P(B|A_j)P(A_j)}$. Этот результат известен как формула Байеса и очень полезен тем, что позволяет "разворачивать" условные вероятности.

Независимые события

События A и B называются независимыми, если $P(A \cap B) = P(A)P(B)$. Если $P(B) > 0$, то это то же самое, что $P(A|B) = P(A)$. Это значит, что знание того, что произошло некоторое событие B, не влияет на наше представление о вероятности события A. Таким образом, эти события в некотором смысле никак не влияют друг на друга. Вероятностную независимость можно определить и для большего числа событий, причём независимость множества событий — более сильное утверждение, чем попарная независимость всех этих событий (и даже чем независимость всех собственных подмножеств).

Случайные величины. Математическое ожидание

Случайная величина — это величина, которая может принимать различные значения в зависимости от исхода. Это значит, что каждому возможному значению случайной величины можно сопоставить вероятность. Простейший пример: величина X, которая принимает значение 0 с вероятностью $\frac{1}{2}$ и значение 1 с вероятностью $\frac{1}{2}$. Записывают так: $P(X = 0) = P(X = 1) = \frac{1}{2}$. Соответствие между множеством значений случайной величины и их вероятностями называется распределением случайной величины.

Пример распределения случайной величины — геометрическое распределение. Оно принимает параметр p, и если величина X имеет такое распределение, то $P(X = i) = qp^i$, где $q = 1 - p$, для всех целых $i \geq 0$. Заметим, что это распределение дискретно — величина может принимать только целые значения. Несложно проверить, что сумма вероятностей по всем возможным значениям i равна 1.

Геометрическое распределение часто встречается на практике как распределение числа неудачных попыток перед первой удачной в испытании, в котором вероятность успеха всегда одинаковая (и равна p). Например, если подбрасывать монетку, то число орлов до первой решки имеет геометрическое распределение с параметром $p = \frac{1}{2}$.

Случайные величины можно складывать, умножать и вычислять результат других математических действий над ними. На значение функции от нескольких величин может влиять вероятность определённых комбинаций их значений. Например, если две случайные величины всегда равны, то их разность всегда равна нулю. Две величины X и Y называются независимыми, если события $X = x$ и $Y = y$ независимы для всех x и y.

Математическое ожидание случайной величины — в некотором смысле её среднее значение. Формально $E(X) = \sum\limits_x xP(X = x)$, где x пробегает все возможные значения X. Математическое ожидание суммы случайных величин равно сумме их математических ожиданий, а математическое ожидание произведения случайной величины и константы равно произведению математического ожидания этой же величины и этой же константы: $E(X + Y) = E(X) + E(Y)$ и $E(cX) = cE(X)$. Если случайные величины независимы, то произведение их математических ожиданий равно математическому ожиданию их произведения, но в общем случае это неверно.

Вероятностная динамика для математического ожидания

Во многих задачах требуется найти математическое ожидание числа шагов до наступления некоторого события. В таких задачах можно использовать динамику, в которой значением является искомое математическое ожидание при условии, что в качестве начального используется текущее состояние. Формула пересчета обычно выглядит так: $D_i = 1 + \sum\limits_j P_{ij} D_j$. Как и в обычной вероятностной динамике, значения вычисляются с конца, а если переходы обратимы, то приходится решать систему линейных уравнений.

Пример: лягушка начинает с точки с координатой 0 и за один шаг из точки с координатой i с равной вероятностью прыгает в точку с координатой $i+1$ или $i+2$. Найти, за сколько шагов (в среднем) она попадёт в точку, координата которой не меньше n. Решение: состояние динамики — координата лягушки, значение — искомая величина, но при условии, что лягушка начинает путь из текущей точки. Формулы: для $i \geq n$ $D_i = 0$, иначе $D_i = 1 + \frac{D_{i+1}+D_{i+2}}{2}$.

Кроме вариантов с необратимыми и обратимыми переходами есть промежуточный вариант, когда переходы необратимы, за исключением возможных переходов из некоторых состояний в себя. В этом случае нужно действовать так же, как и с необратимыми переходами, но на каждом шаге решать линейное уравнение с одной неизвестной.

Цепи Маркова

Цепь Маркова — случайный процесс, в котором очередное состояние зависит только от предыдущего состояния и, возможно, от порядкового номера. Если зависимости от номера нет, то цепь называется однородной. Именно такие цепи обычно встречаются в задачах. Более того, обычно у них конечное число состояний, а иногда даже можно явно построить матрицу переходов.

Матрица переходов $M = \|p_{ij}\|$, где p_{ij} — вероятность перехода из состояния i в состояние j и для всех i $\sum\limits_j p_{ij} = 1$. Матрица переходов в цепи Маркова всегда является стохастической — все её элементы неотрицательны и сумма элементов в каждой строке равна единице. Также представляет интерес граф переходов — ориентированный граф, в котором есть ребро из вершины i в вершину j, если $p_{ij} > 0$. Любая возможная последовательность состояний цепи Маркова соответствует пути в графе переходов, и, наоборот, любой путь в графе переходов соответствует возможной последовательности состояний в цепи.

Стационарное распределение — распределение состояний цепи, которое не изменяется со временем, то есть если распределение некоторого состояния является стационарным, то все последующие состояния будут распределены так же. Если вероятности состояний заданы вектором π, то $\pi M = \pi$. Часто, но не всегда, распределение в цепи со временем стремится к стационарному.

Состояния в цепи Маркова можно классифицировать по достижимости. Состояние j достижимо из состояния i, если система может из состояния i перейти в состояние j за конечное число шагов (записывают $i \to j$). Состояния i и j коммуницируют (записывают $i \leftrightarrow j$), если $i \to j$ и $j \to i$. Это отношение порождает классы эквивалентности, называемые неразложимыми классами. Если вся цепь представляет собой один такой класс, то её также называют неразложимой.

Состояние называется периодическим, если, выйдя из него, вернуться можно только через число шагов, кратное некоторому целому числу, большему единицы. В противном случае состояние называют апериодическим. Можно доказать, что в неразложимом классе все состояния или апериодические, или периодические с одинаковым периодом.

Оказывается, если цепь Маркова неразложима и апериодична, то у неё есть единственное стационарное распределение, к которому она будет стремиться из любого начального распределения. Этот результат известен как теорема Фробениуса — Перрона. Если цепь неразложима и периодична, то всё равно распределения состояний, номера которых сравнимы между собой по модулю периода, имеют предел (например, если период равен двум, то распределения чётных и нечётных состояний имеют предел, но эти пределы могут не совпадать). Эти пределы могут зависеть от начального распределения.

Существует несколько приёмов работы с цепями Маркова. Например, можно использовать алгоритм быстрого возведения матрицы в степень, чтобы быстро считать распределение состояний через большое число шагов. Чтобы найти стационарное распределение, нужно решить систему уравнений $\pi M = \pi$ (по одному уравнению для каждой координаты), в которой вместо одного из уравнений вставлено уравнение $\sum p_i = 1$ (заменить можно любое из уравнений).

Задачи и разборы

Задача A. Трудный путь (Юниорская лига)

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Вася хорошо выпил и теперь, когда он добрался до своей улицы, он полностью потерял чувство направления. Поскольку он не помнит, с какой стороны его дом, он выбирает направление наобум. Более того, на каждом перекрёстке он с вероятностью 50% продолжает идти вперёд, а иначе разворачивается и идёт назад. Он настолько потерял связь с реальностью, что может даже пройти мимо своего дома и не заметить этого!

Пройдя N кварталов, Вася засыпает прямо на улице. Проснувшись, он задаётся вопросом: какой у него был шанс заснуть рядом с домом? Ведь от перекрёстка, от которого он начал свой путь, до перекрёстка рядом с домом Васи всего M кварталов. Помогите ему.

Ограничения

$1 \le N \le 1000$
$0 \le M \le 1000$

Формат входного файла

Первая строка входного файла содержит два целых числа: N и M.

Формат выходного файла

Выведите единственное число — вероятность Васи заснуть на перекрёстке рядом со своим домом. Выведите ответ с абсолютной погрешностью не более 10^{-7}.

Пример

stdin	stdout
1 1	0.5
10 20	0.0
1000 100	0.0001694

Разбор задачи A. Трудный путь

Будем представлять путь Васи как строку из нулей и единиц, где ноль обозначает квартал, пройденный в одном направлении, а единица — в другом. Тогда, если $N = 1$, то очевидно, что два возможных пути — 0 и 1 — равновероятны. На каждом следующем шаге к пути с равной вероятностью дописывается 0 и 1, так что после N шагов получается 2^N возможных путей, и все они равновероятны.

Пусть в пути K единиц. Несложно заметить, что точка, где закончится путь, отстоит от начальной на $|N-2K|$ кварталов, причём $0 \le K \le N$. Значит, если $M > N$ или M и N имеют разную чётность, ответ равен нулю, так как такой точке не соответствует ни один путь. Иначе $K = \frac{N+M}{2}$ (будем считать, что единица — это квартал, пройденный в направлении дома) и ответ равен $\frac{C_N^K}{2^N}$ — числу допустимых путей, деленному на общее число путей.

Значение C_N^K легко вычисляется с помощью факториалов. Однако, если производить вычисления с использованием типа double, то может произойти переполнение. Чтобы этого избежать, можно использовать тип long double или вместо самих значений производить операции над их логарифмами.

Задача B. Случайное совпадение (Юниорская лига)

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Васе скучно, и он от нечего делать взял N кубиков, пронумерованных числами от 1 до N, перемешал их и расставил в случайном порядке. После этого он несколько раз перемешал кубики и расставил их в том порядке, в котором они оказались. После каждого перемешивания он смотрел, не получилась ли та же самая перестановка, что и перестановка, получившаяся после первого перемешивания. Но кубиков было много, искомая перестановка всё никак не получалась, и Васе опять стало скучно. Но Вася подумал и решил получить искомую перестановку другим способом. Он расставил кубики по порядку от 1 до N и решил выбрать часть кубиков, перемешать их и вернуть на те же места, но, возможно, в другом порядке. Он решил повторять эти действия до тех пор, пока не получится перестановка, которую он запомнил в самом начале, но перед этим ему хочется узнать, через какое время, в среднем, он сможет этого добиться? Вася хочет получить эту перестановку как можно быстрее и выбирает кубики, которые он будет перемешивать, соответственно.

Ограничения

$1 \le N \le 1000$

Формат входного файла

Первая строка входного файла содержит целое число N.

Вторая строка содержит N целых чисел от 1 до N — перестановка, которую хочет получить Вася.

Формат выходного файла

Выведите единственное число — среднее количество перемешиваний, после которых Вася получит искомую перестановку.

Пример

stdin	stdout
3 2 1 3	2.0
3 3 1 2	3.0

Пояснение

В первом примере Вася каждый раз выбирает для перемешивания первые два кубика. С вероятностью $\frac{1}{2}$ после перемешивания кубики окажутся переставленными, и на этом процесс закончится. В среднем это произойдёт после двух перемешиваний.

Разбор задачи B. Случайное совпадение

Пусть текущая перестановка равна $\{P_i\}_{i=1}^{N}$, и для очередного перемешивания Вася выбрал множество из n кубиков, находящихся в позициях $\{i_k\}_{k=1}^{n}$. Поскольку все перестановки кубиков равновероятны, после перемешивания каждый из n перемешиваемых кубиков с вероятностью $\frac{1}{n}$ окажется на каждой из n позиций. Из этих позиций в лучшем случае одна будет совпадать с позицией этого кубика в искомой перестановке. Будем считать, что $X_k = 1$, если кубик из позиции i_k после перемешивания попал на позицию в искомой перестановке, иначе $X_k = 0$. В этом случае у тех кубиков, у которых позиция в искомой перестановке находится среди позиций перемешиваемых кубиков, $E(X_k) = \frac{1}{n}$, у остальных же кубиков $E(X_k) = 0$. Но $\sum_{k=1}^{n} X_k$ — это число кубиков (из перемешиваемых), которые после перемешивания оказались на целевых позициях, и $E(\sum_{k=1}^{n} X_k) = \sum_{k=1}^{n} E(X_k) \leq 1$. Значит, из всех перемешиваемых кубиков в среднем не более одного окажется на целевой позиции, а равенство достигается, когда для каждого из перемешиваемых кубиков его целевая позиция также входит в множество позиций перемешиваемых кубиков.

Значит, за каждое перемешивание число кубиков, стоящих на своих местах, можно увеличить, в среднем, не более чем на один. Как увеличить ровно на один? Оказывается, для этого достаточно каждый раз перемешивать ровно те кубики, которые стоят не на своих местах. До перемешивания среди них ноль кубиков стоят на своих местах, кроме того, для каждого из кубиков в перемешиваваемое множество входит и кубик на позиции, где он должен стоять (так как он тоже стоит не на месте). Значит, после перемешивания в среднем ровно один из этих кубиков будет стоять на своём месте, то есть на один больше, чем до перемешивания.

Докажем, что если n кубиков стоят не на своих местах, то ответ (назовём его A_n) равен n. Доказывать будем по индукции. Если $n = 0$, то все кубики стоят на своих местах и ничего делать не надо. Иначе перемешаем те кубики, которые стоят не на своих местах. Пусть после этого m кубиков стоят не на своих местах, причём $0 \le m \le n$, так как кубики, уже стоящие на своих местах, при перемешивании не затронуты. Тогда, согласно доказанному ранее, $E(m) = n - 1$, то есть

$$P(m = 1) + 2P(m = 2) + \ldots + nP(m = n) = n - 1.$$

Но тогда

$$A_n = 1 + A_0 P(m = 0) + A_1 P(m = 1) + \ldots + A_{n-1} P(m = n-1) + A_n P(m = n).$$

По предположению индукции для всех $m < n$ $A_m = m$, из чего получается, что $A_n = n$. То, что такая стратегия оптимальна, доказывается аналогично: если на очередном шаге сделать выбор по-другому, то A_n получится не меньше, чем при данной стратегии.

Задача C. Полный набор (Юниорская лига)

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Вася продолжает умирать от скуки. Чтобы хоть как-то развлечься, он взял N кубиков, пронумерованных от 1 до N, перемешал их и взял K из них наобум, после чего записал их номера и вернул их в общую кучу. Затем он повторил эти действия: снова перемешал, снова взял K кубиков, и так далее. И теперь у него возник вопрос: сколько раз нужно так сделать, чтобы каждый кубик был взят хотя бы по одному разу?

Ограничения

$1 \le N \le 1000$
$1 \le K \le N$

Формат входного файла

Первая строка входного файла содержит два целых числа: N и K.

Формат выходного файла

Выведите единственное число — среднее количество итераций до того, как каждый кубик будет взят хотя бы по разу. Выведите ответ с относительной погрешностью не более 10^{-7}.

Пример

stdin	stdout
5 1	11.4166667
5 5	1.0

Разбор задачи C. Полный набор

Поскольку все кубики равнозначны, в каждый момент времени нас интересует только то, сколько кубиков было взято хотя бы по одному разу. Будем реализовывать вероятностную динамику, где состояние — количество уже взятых когда-либо кубиков, а значение — среднее количество итераций, начиная с этого момента, до того, как будут взяты все кубики.

Динамика считается с конца. Для N значение динамики равно 0, поскольку все кубики уже взяты. Пусть взято i кубиков, тогда на следующей итерации может быть взято $i+j$ кубиков, где $0 \le j \le K$ и $K \le i+j \le N$. Вероятность того, что после i будет $i+j$, считается так: всего есть C_N^K способов выбрать K кубиков из N, и все они равновероятны. Из них $C_i^{K-j}C_{N-i}^{j}$ способов приведут к тому, что на следующей итерации будет взято ровно $i+j$ кубиков. Таким образом, вероятность этого равна $\frac{C_i^{K-j}C_{N-i}^{j}}{C_N^K}$.

Теперь несложно написать формулу для перехода: если $i < k$, то $D_i = 1 + \frac{\sum_{K-i \le j \le \min(N-i,K)} C_i^{K-j}C_{N-i}^{j}D_{i+j}}{C_N^K}$, если же $i \ge k$, то $D_i = \frac{C_N^K + \sum_{1 \le j \le \min(N-i,K)} C_i^{K-j}C_{N-i}^{j}D_{i+j}}{C_N^K - C_i^K}$ где D_i — значение динамики в состоянии i, а $D_N = 0$. Ответом задачи будет значение D_0.

Задача D. Питание

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Вася решил отправиться в путешествие. Сейчас он летит на самолёте, где как раз начинают раздавать обед. Обед бывает двух видов — мясной и рыбный, и бортпроводники спрашивают у каждого из пассажиров, какой вид обеда они предпочитают.

Но Вася знает, что каждого вида обеда в отдельности не хватит на всех пассажиров. Если обеды одного вида заканчиваются, бортпроводники перестают спрашивать пассажиров об их предпочтениях и просто дают им то, что осталось. Вася очень хочет получить рыбный обед. Он знает, что на самолёт, на котором летят N пассажиров, взято N_1 мясных и N_2 рыбных обедов. Кроме того, он посмотрел, в каком порядке раздают обеды, и заметил, что он I-й по счёту. Также он предусмотрительно раздобыл статистику, из которой узнал, что рыбные обеды предпочитают $P\%$ пассажиров. Теперь ему не терпится узнать: какой у него шанс получить рыбный обед?

Ограничения

$0 \leq N_1 \leq 1000$
$0 \leq N_2 \leq 1000$
$1 \leq N \leq N_1 + N_2$
$1 \leq I \leq N$
$0 \leq P \leq 100$

Формат входного файла

Первая строка входного файла содержит пять целых чисел: N, N_1, N_2, I и P.

Формат выходного файла

Выведите единственное число — вероятность Васи получить рыбный обед. Выведите ответ с абсолютной погрешностью не более 10^{-7}.

Пример

stdin	stdout
10 5 5 10 50	0.5
10 8 7 8 60	0.9720064

Разбор задачи D. Питание

Условие задачи специально усложнено тем, что некоторые пассажиры не всегда делают выбор, а также ненужным значением N. Действительно, Васе совершенно неважно, сколько пассажиров после него будут заказывать обед. Более того, будем считать, что все пассажиры (все — это $I - 1$ пассажиров перед Васей, остальные его не интересуют) сделали выбор, просто некоторые из них не имеют шанса его высказать. Теперь заметим, что рыба достанется Васе тогда и только тогда, когда из пассажиров впереди него рыбу выбрали менее чем N_2 пассажира. Действительно, если перед Васей рыбу выбрали хотя бы N_2 пассажира, то она точно закончится и Васе не достанется, иначе рассмотрим два случая: мясо перед Васей закончилось или не закончилось. В первом случае рыба не может закончиться, так как $N_1 + N_2 \geq N$, а во втором каждому достанется то, что он выбрал, а рыбу хочет менее, чем N_2 пассажира, поэтому Васе она достанется.

Таким образом, осталось найти вероятность того, что из первых $I-1$ пассажиров рыбу выбрали меньше, чем N_2. Заметим, что количество пассажиров, выбравших рыбу, подчиняется биномиальному распределению. Если $N_2 \geq I$, то эта вероятность равна 1, иначе нужно просуммировать вероятности для значений от 0 до $N_2 - 1$:

$$P_a = \sum_{i=0}^{N_2-1} P^i (1-P)^{I-1-i} C_{I-1}^i$$

(здесь считаем, что $0 \leq P \leq 1$, P_a — искомая вероятность).

Задача E. Секретный код

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Вернувшись из путешествия, Вася был неприятно удивлен: на двери подъезда его дома установлен новый кодовый замок. Вася не может попасть к себе домой! Кодовый замок содержит N дисков, каждый из которых может находиться в одном из M положений. Ровно одна комбинация является подходящей. Внимательно осмотрев диски, Вася по отпечаткам пальцев и царапинам определил вероятность каждого из положений для каждого диска. Теперь у Васи есть K попыток подобрать код: если он не успеет, то бдительные соседи вызовут полицию, и Васе придётся долго доказывать, что он не вор, а просто пытается попасть домой. Помогите Васе посчитать максимальную вероятность оказаться дома, а не в полиции.

Ограничения

$1 \leq N \leq 100$
$1 \leq M \leq 20$
$1 \leq K \leq 100$
$0 \leq P_{ij} \leq 100$

Формат входного файла

Первая строка входного файла содержит три целых числа: N, M и K.

Следующие N строк содержат по M целых чисел каждая: j-е число i-й строки (P_{ij}) — вероятность того, что i-й диск в подходящей комбинации находится в положении j. Гарантируется, что $\sum_{j=1}^{M} P_{ij} = 100$.

Формат выходного файла

Выведите единственное число — вероятность Васи успеть подобрать код. Выведите ответ с абсолютной погрешностью не более 10^{-7}.

Пример

stdin	stdout
2 2 1 50 50 10 90	0.45
3 5 4 10 15 20 25 30 1 2 3 4 90 100 0 0 0 0	0.81

Разбор задачи E. Секретный код

Очевидно, что оптимальная стратегия для Васи — попробовать K наиболее вероятных комбинаций. Вероятность того, что некоторая комбинация является допустимой, равна произведению соответствующих вероятностей для каждого диска. Как узнать K самых вероятных комбинаций? Заметим, что если комбинация входит в K самых вероятных, то комбинация из первых $N-1$ дисков тоже входит в K самых вероятных среди комбинаций из первых $N-1$ дисков: иначе можно заменить комбинацию из первых $N-1$ дисков на более вероятную, что увеличит вероятность комбинации из N дисков, таким образом, чтобы она не совпала ни с одной из более вероятных комбинаций. Аналогично, комбинация из первых $N-2$ дисков также входит в K самых вероятных, как и комбинация из первых $N-3$, и т. д.

Чтобы посчитать ответ, начнём с комбинации из нуля дисков. Такая комбинация ровно одна. Теперь на каждом шаге будем дописывать ко всем комбинациям все возможные значения следующей цифры, после чего оставлять только K наиболее вероятных. Если для поддержания K наиболее вероятных комбинаций использовать биномиальную кучу, то просеивание KM комбинаций может быть сделано за время $O(KM \log K)$. Всего нужно сделать N шагов, что делается за время $O(NKM \log K)$. Осталось только просуммировать вероятности.

Задача F. Dura Lex

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Вася обожает заказывать новые гаджеты из-за границы. К сожалению, для Васи, недавно ввели новые правила таможенного контроля, согласно которым для получения каждого гаджета Васе нужно получить N справок. Получить каждую справку непросто — чтобы получить i-ю справку, нужно отстоять в очереди D_i дней, причём одновременно можно стоять в очереди не более чем за одной справкой. И что самое обидное, в выдаче i-й справки отказывают с вероятностью $P_i\%$, причём совершенно случайно, и более того, если в выдаче справки отказано, то все предыдущие справки автоматически аннулируются, и всё приходится начинать сначала. Хоть одно радует — справки можно получать в любом порядке.

Вася хочет получить новый гаджет во что бы то ни стало. Он будет пытаться собрать все справки, пока не добьётся успеха. Помогите ему выбрать порядок получения справок таким образом, чтобы минимизировать среднее время, за которое он соберёт их все.

Ограничения

$1 \leq N \leq 10^5$
$0 \leq D_i \leq 1000$
$0 \leq P_i < 100$

Формат входного файла

Первая строка входного файла содержит целое число N.
Следующие N строк содержат по два целых числа каждая: D_i и P_i.

Формат выходного файла

Выведите N целых чисел от 1 до N — искомую перестановку. Если оптимальных перестановок несколько, выведите лексикографически минимальную.

Пример

stdin	stdout
3 1 10 10 10 5 90	3 1 2
3 2 20 3 30 4 40	1 2 3

Разбор задачи F. Dura Lex

Пусть Вася пытается получить i-ю справку. Вероятность получить её с первого раза равна $1 - P_i$, со второго — $P_i(1 - P_i)$ (нужно, чтобы отказали в первый раз, но не во второй), с третьего — $P_i^2(1 - P_i)$ и т. д. Несложно заметить, что количество попыток подчиняется геометрическому распределению и среднее количество попыток до успешного получения справки равно $\frac{1}{1-P_i}$.

Пусть Вася пытается получить i-ю справку, а затем j-ю. Ему нужно в среднем $\frac{1}{1-P_i}$ попыток, чтобы получить i-ю справку, но каждое получение i-й справки даёт ему всего одну попытку для получения j-й справки, а таких попыток нужно в среднем $\frac{1}{1-P_j}$, так что всего попыток будет $\frac{1}{(1-P_i)(1-P_j)}$. В каждой из этих попыток Вася попытается получить i-ю справку, а в $\frac{1}{1-P_j}$ из них — ещё и j-ю, что в сумме занимает, в среднем, $\frac{D_i}{(1-P_i)(1-P_j)} + \frac{D_j}{1-P_j}$ дней.

Пусть Вася пытается получить i_1-ю справку, затем i_2-ю, и так далее, затем i_N. Применяя рассуждения, аналогичные выше, можно узнать, что это в среднем это займёт $\frac{D_{i_1}}{(1-P_{i_1})(1-P_{i_2})...(1-P_{i_N})} + \frac{D_{i_2}}{(1-P_{i_2})...(1-P_{i_N})} + \ldots + \frac{D_{i_N}}{1-P_{i_N}}$ дней. Если переставить две соседние справки — i_j-ю и i_{j+1}-ю — то это время увеличится на $\frac{D_{i_{j+1}}P_{i_j} - D_{i_j}P_{i_{j+1}}}{(1-P_{i_j})...(1-P_{i_N})}$. Соответственно, нам выгодно делать такую замену, если $D_{i_{j+1}}P_{i_j} < D_{i_j}P_{i_{j+1}}$. Можно доказать, что это отношение транзитивно и что такую замену выгодно делать не только для соседних справок.

Соответственно, решение — отсортировать справки, сравнивая их способом, указанным выше. Чтобы получить лексикографически минимальную перестановку, эквивалентные справки нужно выводить по порядку. Небольшая проблема состоит в том, что справки, для которых $D_i = P_i = 0$, не влияют на результат и могут быть вставлены в любое место списка. Предлагается выносить их в отдельный список и в конце объединять с остальными.

Задача G. Путь к знаниям

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Вася учится в университете и каждый день ходит туда пешком. Город, в котором живёт Вася, представляет собой неориентированный граф. Вася решил использовать научный подход в выборе дороги, поэтому он изучил карту города и нашёл все кратчайшие маршруты от дома до университета. Теперь каждый раз, когда Вася идёт в университет или обратно, он выбирает один из маршрутов, причём каждый маршрут выбирается с равной вероятностью.

Через несколько дней Вася заметил, что через некоторые перекрёстки он ходит чаще, чем через другие. Он решил посчитать, сколько раз в день он в среднем проходит через каждый перекрёсток. Но, поскольку он занят учёбой, он поручил это сделать вам.

Ограничения

$1 \le N \le 10^5$
$0 \le M \le 10^5$
$1 \le A_i, B_i \le N$
$1 \le L_i \le 10000$

Формат входного файла

Первая строка входного файла содержит два целых числа N и M — количество перекрёстков и дорог в городе, где живёт Вася.

Каждая из следующих M строк соответствует одной улице и содержит три целых числа A_i, B_i и L_i — номера перекрёстков, которые соединяет улица, и её длину в километрах.

Дом Васи находится рядом с первым перекрёстком, а университет — рядом с N-м. Гарантируется, что от дома Васи можно дойти до университета по дорогам.

Формат выходного файла

Выведите N чисел — среднее количество проходов в день через перекрёстки с первого по N-й. Выводите числа с абсолютной погрешностью не более 10^{-7}. Не забывайте, что Вася проходит по городу два раза в день — в университет и обратно.

Пример

stdin	stdout

`3 3`
`1 2 1`
`2 3 1`
`1 3 1`
	`2.0 0.0 2.0`

`4 4`
`1 2 1`
`1 3 1`
`2 4 1`
`3 4 1`
	`2.0 1.0 1.0 2.0`

Разбор задачи G. Путь к знаниям

Эта задача не столько на теорию вероятностей, сколько на умение вычислять количество кратчайших путей, проходящих через некоторую вершину. Вначале нужно посчитать расстояние до каждой из вершин от первой и N-й вершины. Затем, если сумма этих расстояний для какой-то вершины больше, чем расстояние между первой и N-й вершиной, то такая вершина не лежит ни на одном кратчайшем пути. Иначе число кратчайших путей равно произведению числа кратчайших путей от этой вершины до первой вершины и до N-й вершины. Эти числа легко считаются с помощью динамики по графу, можно даже это делать параллельно с подсчётом расстояний. Осталось только поделить число кратчайших путей, проходящих через вершину, на общее число кратчайших путей и умножить результат на два.

Задача H. Гранит науки

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

В университете, где учится Вася, начинается новый семестр. В новом семестре Васе предстоят занятия по N предметам, причём по каждому предмету занятия будут проходить каждый день. Перед началом семестра Вася узнал, сколько всего занятий планируется по каждому из предметов: по i-му предмету планируется N_i занятий для всех i от одного до N. Кроме того, от студентов старших курсов Вася узнал трудность каждого из занятий: число H_{ij} для всех i и j обозначает трудность j-го занятия по i-му предмету.

Многие преподаватели ещё не вернулись из отпусков, поэтому занятия по некоторым предметам начнутся не сразу. Если точнее, то первый день занятий по каждому из предметов выбирается равновероятно из первых M учебных дней. Может даже получиться так, что в первый учебный день вообще не будет занятий. После того как занятия по какому-то предмету начинаются, они проходят регулярно, по одному занятию в день, пока не будут проведены все N_i занятий.

Во время учёбы Вася устаёт. Вася посчитал, что за один день его усталость равна квадрату суммарной трудности всех занятий в этот день. Чтобы лучше понять, что ему предстоит, Вася хочет узнать, чему будет равна его суммарная усталость за весь семестр. Поскольку Вася ещё не знает, когда именно начнутся занятия, его интересует среднее значение.

Ограничения

$1 \le N \le 500$
$1 \le M \le 500$
$1 \le N_i \le 500$
$0 \le H_{ij} \le 1000$

Формат входного файла

Первая строка входного файла содержит два целых числа: N и M.

Каждая из следующих N строк соответствует одному предмету и содержит целое число N_i — число занятий по этому предмету, — и N_i целых чисел H_{ij} — трудность каждого из этих занятий.

Формат выходного файла

Выведите единственное число — среднюю суммарную усталость Васи за семестр. Выведите ответ с абсолютной или относительной погрешностью не более 10^{-7}.

Пример

stdin	stdout
2 1 1 1 1 2	9.0
3 3 1 3 2 1 1 3 0 0 0	14.3333333

Разбор задачи H. Гранит науки

Пусть i-е занятие начинается в S_i-й день, а усталость Васи в i-й день равна T_i, тогда $T_i = (\sum_j H_{j(i-S_j+1)})^2$, где суммирование происходит по всем j, для которых значение $H_{j(i-S_j+1)}$ определено. Нам необходимо найти $E(\sum_i T_i) = E(\sum_i(\sum_j H_{j(i-S_j+1)})^2) = \sum_i \sum_{j_1} E(H_{j_1(i-S_{j_1}+1)} \sum_{j_2} H_{j_2(i-S_{j_2}+1)})$. Если из суммы по j_2 вынести слагаемое, где $j_2 = j_1$, то оставшаяся часть будет независима от S_{j_1}, поэтому можно будет заменить среднее произведение произведением средних: $\sum_i \sum_{j_1} E(H_{j_1(i-S_{j_1}+1)} \sum_{j_2} H_{j_2(i-S_{j_2}+1)}) = \sum_i \sum_{j_1}(E(H^2_{j_1(i-S_{j_1}+1)}) + E(H_{j_1(i-S_{j_1}+1)}) \sum_{j_2 \neq j_1} E(H_{j_2(i-S_{j_2}+1)}))$.

Осталось научиться вычислять $E(H_{j(i-S_j+1)})$ и $E(H^2_{j(i-S_j+1)})$ для всех i и j. Они равны $\frac{\sum_{S_j=1}^{M} H_{j(i-S_j+1)}}{M}$ и $\frac{\sum_{S_j=1}^{M} H^2_{j(i-S_j+1)}}{M}$ соответственно. Используя скользящие суммы, можно вычислить эти значения за $O(\sum N_i)$. Чтобы быстро вычислять суммы по $j_2 \neq j_1$, можно из суммы по всем j вычитать значение для j_1. Таким образом можно вычислить ответ за $O(MN + \sum N_i)$.

Задача I. Вероятный диагноз

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

В качестве курсовой работы Вася проектирует информационную систему больницы. Сейчас ему нужно написать компонент, который ставит предварительный диагноз на основе статистических данных.

В программу загружена информация об N болезнях и M возможных симптомах. Для каждой болезни известна вероятность того, что у больного будет проявляться каждый из симптомов. Аналогичная информация известна и для здоровых людей. Вася предполагает, что каждый из пациентов болен не более чем одной болезнью, а также то, что если зафиксировать болезнь (или её отсутствие), то различные симптомы будут проявляться независимо друг от друга.

В программу загружается информация о K пациентах. Для каждого из пациентов известно, что некоторые из симптомов у него проявляются, а некоторые нет, про некоторые же из симптомов ничего не известно. Нужно определить вероятность того, что он болен каждой из известных болезней.

Ограничения

$1 \leq N \leq 200$

$1 \leq M \leq 200$
$1 \leq K \leq 200$

Формат входного файла

Первая строка входного файла содержит два целых числа: N и M.

Следующая строка содержит M чисел — вероятность проявления каждого из симптомов у здорового человека.

Следующие N строк содержат информацию о болезнях. Каждая из них содержит $M+1$ чисел: вероятность того, что человек болен этой болезнью (сумма этих вероятностей по всем болезням на превосходит 100%), и вероятность проявления каждого из симптомов, если человек болен этой болезнью.

Следующая строка содержит единственное целое число K.

Следующие K строк содержат информацию о пациентах: каждая из них содержит строку длины M, каждый из символов которой равен `+`, если соответствующий симптом у пациента проявляется, `-`, если не проявляется, или `?`, если о наличии этого симптома нет сведений.

Все вероятности во входном файле указаны в процентах, это числа от 0 до 100 ровно с двумя знаками после точки. Гарантируется, что комбинация симптомов для каждого пациента имеет ненулевую вероятность.

Формат выходного файла

Выведите K строк — по одной на каждого пациента. Каждая строка должна содержать N чисел — вероятности того, что соответствующий пациент болен каждой из болезней. Выводите вероятности как числа от 0 до 1 с абсолютной погрешностью не более 10^{-7}.

Пример

stdin	stdout
1 2 10.00 10.00 40.00 90.00 90.00 3 ?? ++ +-	0.4000000 0.9818182 0.4000000
2 3 10.00 30.00 50.00 10.00 50.00 50.00 50.00 1.00 20.00 30.00 20.00 4 --? ++? ??- ---	0.0422797 0.0094707 0.4780115 0.0114723 0.0994036 0.0159046 0.0420408 0.0150674

Разбор задачи I. Вероятный диагноз

Задача эта, в общем-то, на применение формулы Байеса. Нам известна априорная вероятность того, что пациент болен каждой из болезней, а условная вероятность того, что пациент имеет именно такие симптомы, получается путём перемножения вероятностей для отдельных симптомов (для тех симптомов, которые проявляются, используется вероятность того, что соответствующий симптом проявляется, для тех, которые не проявляются, соответственно, используется вероятность того, что симптом не проявляется). Задача состоит в том, чтобы найти условную вероятность того, что больной болен определённой болезнью по симптомам. Вероятности для каждого пациента вычисляются за $O(MN)$, таким образом общая сложность получается $O(MNK)$.

Задача J. Зоологический эксперимент

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Вася занимается в зоологическом кружке и ставит там эксперименты над шушпанчиками. В одном из экспериментов он помещает двух шушпанчиков в лабиринт, который представляет собой неориентированный граф. Каждую секунду каждый из шушпанчиков выбирает одну из вершин лабиринта, смежную с текущей, и прыгает туда. Шушпанчики выбирают каждую из смежных вершин с равной вероятностью. Хотя они и находятся в одном лабиринте, они никак не реагируют друг на друга и движутся совершенно независимо. По крайней мере, Вася так считает. Чтобы проверить эту гипотезу, он решил измерить, какую часть времени, в среднем, шушпанчики проводят в одной и той же вершине. Чтобы избежать погрешности, Вася усредняет долю секунд, которую шушпанчики находились в одной вершине, за продолжительный период времени. Также Вася считает, что несмотря на то, что шушпанчики никак не реагируют друг на друга, они выдерживают ритм с такой точностью, что прыгали всё это время совершенно синхронно. Необходимо сделать теоретический расчёт этой величины.

Ограничения

$2 \le N \le 100$
$1 \le M \le 10000$

Формат входного файла

Первая строка входного файла содержит два целых числа: N и M — количество вершин и рёбер в лабиринте.

Каждая из следующих M строк содержит два целых числа — номера вершин (от 1 до N), соединённых ребром. Гарантируется, что в графе нет петель, параллельных рёбер и изолированных вершин.

Следующая строка содержит два целых числа от 1 до N — номера вершин, в которые были изначально помещены шушпанчики. Эти номера могут совпадать. Шушпанчики начинают движение одновременно.

Формат выходного файла

Выведите единственное число — долю времени, которую шушпанчики должны проводить в одной и той же вершине, если Васины предположения верны.

Пример

stdin	stdout
5 5 1 2 2 3 3 4 4 5 1 5 1 2	0.2

Разбор задачи J. Зоологический эксперимент

Эта задача призвана показать различные виды состояний в цепях Маркова. В начале следует проверить некоторые особые случаи. Во-первых, если шушпанчики начинают путь в различных компонентах связности, то ответ равен нулю. В дальнейшем будем рассматривать только ту компоненту, в которой находятся шушпанчики. Во-вторых, если эта компонента двудольная и шушпанчики находятся в различных долях, ответ также равен нулю.

Дальнейшее решение зависит от того, является ли компонента с шушпанчиками двудольной. Если нет, то несложно доказать, что соответствующая цепь Маркова является неразложимой и апериодической, а значит, у неё есть единственное стационарное состояние и система стремится к нему из любого начального состояния. Если же компонента является двусвязной, то цепь, хотя и остаётся неразложимой, будет периодической с периодом 2. В этом случае есть смысл разбить цепь на две: одну для чётных состояний, а другую для нечётных. Переходы внутри этих цепей будут делаться сразу на два шага. Дальше нужно и в том, и в другом случае найти стационарное состояние, например, решив систему уравнений методом Гаусса. Затем нужно просуммировать квадраты вероятностей и, если цепей две, взять среднее арифметическое (поскольку половину времени шушпанчики проводят в одной цепи, а половину в другой).

Задача K. Игра (Высшая лига)

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Вася встретил приятеля, который предложил ему сыграть на коллекционные карты. Игра проходит в несколько раундов, и Вася рассчитал, что в каждом раунде он может выиграть с вероятностью $P\%$. Если он выигрывает раунд, то приятель отдаёт ему N_1 коллекционных карт, иначе Вася отдаёт приятелю N_2 коллекционных карт. Когда проигравшему нечем платить, игра заканчивается, и тот, у кого закончились карты, считается проигравшим всю игру. У Васи M_1 коллекционных карт, и он знает, что у приятеля M_2 коллекционных карт. Чтобы решить, стоит ли играть, Вася хочет посчитать вероятность своего выигрыша при таких условиях, но ему это не под силу. Помогите ему.

Ограничения

$1 \leq N_1, N_2, M_1, M_2 \leq 50$
$0 \leq P \leq 100$

Формат входного файла

Первая строка входного файла содержит пять целых чисел: N_1, N_2, M_1, M_2 и P.

Формат выходного файла

Выведите единственное число — вероятность Васи выиграть всю игру. Выведите ответ с абсолютной погрешностью не более 10^{-7}.

Пример

stdin	stdout
`1 1 10 10 50`	`0.5`
`2 1 10 10 40`	`0.8943674`

Разбор задачи К. Игра

Хотя задача и кажется сложной, решается она достаточно просто — с помощью вероятностной динамики. В качестве состояния можно использовать, например, количество денег у Васи, тогда количество денег у приятеля определяется однозначно. Значение — вероятность выигрыша. Каждому состоянию соответствует линейное уравнение, которое соответствует тому, что после очередного хода средняя вероятность выигрыша не изменится. Систему уравнений можно решить методом Гаусса. Сложность решения $O((N_1 + N_2)^3)$.

Задача L. Опасная игра (Высшая лига)

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Вася попал в серьёзную передрягу. Он задолжал мафии N долларов, и время расплаты почти подошло. Вася хорошо знает, чем грозит несвоевременный возврат долга, поэтому он хочет собрать N долларов любой ценой.

Однако, на данный момент у него есть только M долларов. Приятель, который узнал о проблемах Васи, предложил Васе игру — в этот раз на деньги. У Васи есть K попыток сделать ставку на любую сумму от нуля до той суммы, которая у него есть — не обязательно даже целую. С вероятностью $P\%$ Вася получает ставку в двойном объёме, иначе не получает ничего. Васе не важно, сколько денег у него останется в итоге, главное, чтобы было N долларов, чтобы расплатиться. Вася, конечно же, играет оптимально. Узнайте, с какой вероятностью ему повезёт и он расплатится деньгами.

Ограничения

$0 \le N \le 10^9$
$0 \le M \le 10^9$
$1 \le K \le 12$
$0 \le P \le 100$

Формат входного файла

Первая строка входного файла содержит четыре целых числа: N, M, K и P.

Формат выходного файла

Выведите единственное число — вероятность собрать достаточную сумму. Выведите ответ с абсолютной погрешностью не более 10^{-7}.

Пример

stdin	stdout
`1000000 750000 1 10`	`0.1`
`1000000 750000 2 10`	`0.19`

Разбор задачи L. Опасная игра

Сначала может показаться, что эту задачу невозможно решить, поскольку в ней бесконечно много состояний — ставки то могут быть и дробными.

Оказывается, что ответ задачи — кусочно-постоянная функция от M. Будем решать задачу с конца. Если $K = 0$, то если $M < N$, ответ равен нулю, иначе единице. Если $K = 1$, то отрезок делится на две части: если $0 \leq M < \frac{N}{2}$, то ответ равен нулю, если же $\frac{N}{2} \leq M < N$, то ответ равен P. Если же $M \geq N$, то ответ, как и раньше, равен единице.

Оказывается, эту закономерность можно обобщить. Для произвольного K отрезок от нуля до N делится на 2^K равных отрезков, и на каждом из них ответ постоянен. Доказывается это методом индукции по K: действительно, после каждой попытки можно считать, что начинается новая игра с K, меньшим на единицу. Отрезков, соответственно, будет в два раза меньше. Сделав ставку, Вася может перейти из одного отрезка в текущем состоянии в два отрезка в следующем — в один в случае выигрыша, а в другой в случае проигрыша. Пара отрезков зависит от текущего значения N и от ставки, сделанной Васей. Заметим, что из всех точек одного и того же отрезка в текущем состоянии достижимы одни и те же пары отрезков в следующем состоянии. Поскольку различные точки отрезков в следующем состоянии дают одинаковый выигрыш по предположению индукции, для отрезков в текущем состоянии это тоже будет верно.

Из доказательства получается и решение задачи: для каждого K и каждого отрезка будем перебирать все возможные переходы (их порядка 2^K). Так как состояний тоже порядка 2^K, общая сложность получается 2^{2K}.

Задача M. Хеш-таблица (Высшая лига)

Вход:	`stdin`
Выход:	`stdout`
Ограничение по времени:	1 с
Ограничение по памяти:	256 Мб

Недавно Вася узнал про новую структуру данных: хеш-таблицу с открытой адресацией. Хеш-таблица с открытой адресацией состоит из N ячеек, пронумерованных от 1 до N. Каждая из них может быть или свободна, или хранить какое-то значение. При вставке нового значения от него вычисляется хеш — случайное число от 1 до N (назовём его h) — после чего, если ячейка под номером h свободна, то значение записывается в нее. Иначе, если ячейка $h + 1$ свободна, то значение вставляется в неё, иначе проверяются ячейки $h + 2$, $h + 3$ и так далее. Если поиск дошёл до N-й ячейки, и она тоже занята, поиск продолжается с ячейки под номером 1. Таким образом, пока в таблице есть свободные места, значение так или иначе будет добавлено.

Каждая проверка ячейки, которая оказалась занята, называется коллизией. Например, если значение, хеш которого равен h, оказалось записанным в ячейку $h+2$, то при его вставке произошли две коллизии. Коллизии замедляют работу хеш-таблицы, поэтому Вася решил узнать, сколько коллизий будет в среднем, если в пустую хеш-таблицу вставить M различных значений. А вычислять, как всегда, вам.

Ограничения

$1 \le N \le 100$
$0 \le M \le N$

Формат входного файла

Первая строка входного файла содержит два целых числа: N и M.

Формат выходного файла

Выведите единственное число — среднее количество коллизий. Выведите ответ с абсолютной или относительной погрешностью не более 10^{-7}.

Пример

stdin	stdout
5 1	0.0
10 2	0.1

Разбор задачи M. Хеш-таблица

В условии задачи описана хеш-таблица, которая циклически замкнута. Рассмотрим разомкнутую хеш-таблицу. В ней, если произошла коллизия в последней ячейке, будем говорить, что произошло переполнение. Будем решать такую задачу: в разомкнутую хеш таблицу размера N вставили M элементов. Чему равна вероятность того, что переполнения не произойдёт, и сколько при этом в среднем будет коллизий?

Вначале рассмотрим случай, когда $M < N$. Чтобы найти вероятность, разделим одно событие “не произошло переполнение” на несколько отдельных событий, i-е из которых имеет вид “не произошло переполнения и в итоге первая свободная ячейка имеет индекс i”. Эти события несовместимы, поэтому вероятность исходного события равна сумме вероятностей частей. Среднее количество коллизий также легко вычисляется как взвешенное среднее. Теперь заметим, что событие “не произошло переполнения и в итоге первая свободная ячейка имеет индекс i” выполняется тогда и только тогда, когда в первые $i-1$ ячеек попало ровно $i-1$ хешей, в “подтаблице” из первых $i-1$ ячеек не произошло переполнения, в i-ю ячейку не попало ни одного хеша, и в “подтаблице” из последних $N-i$ ячеек, в которую попало $M-i+1$ хешей, также не произошло переполнения. Таким образом, задача сводится к подзадачам с меньшим N. Вероятность того, что запросы лягут нужным образом, легко решается комбинаторными методами и равна $\frac{C_M^{i-1}(i-1)^{i-1}(N-i)^{M-i+1}}{N^M}$.

Если $N = M$, то перебирать нужно не первую свободную ячейку (таковых нет), а последнюю занятую. И опять задача делится на две с меньшими N. Чтобы свести исходную задачу (с замкнутой хеш-таблицей) к задаче с разомкнутой хеш-таблицей, нужно опять же, в зависимости от того, равны ли N и M или нет, перебрать или последнюю занятую ячейку, или первую и последнюю свободную ячейки.

Поскольку состояние динамики квадратичное, а количество переходов линейное, итоговая сложность $O(N^3)$.