Содержание

Пример работы с файлами.

Если вы не умеете читать/выводить данные, или открывать файлы, воспользуйтесь примерами. `http://acm.math.spbu.ru/~sk1/algo/sum/`

Пример работы с файлами.

В некоторых задачах большой ввод и вывод. Про ввод-вывод в C++:
`http://acm.math.spbu.ru/~sk1/algo/input-output/cpp_common.html`
Имеет смысл пользоваться супер быстрым вводом-выводом. Две версии:
`http://acm.math.spbu.ru/~sk1/algo/input-output/io_export.cpp.html`
`http://acm.math.spbu.ru/~sk1/algo/input-output/fread_write_export.cpp.html`

Выделение памяти.

В некоторых задачах нужен STL, который активно использует динамическую память (set-ы, map-ы) переопределение стандартного аллокатора ускорит вашу программу:
`http://acm.math.spbu.ru/~sk1/algo/memory.cpp.html`

Обязательные задачи

Задача 9A. Отметки на подмножествах [0.4 sec, 256 mb]

Рассмотрим множество $\mathcal{S}$, состоящее из n элементов — натуральных чисел 1, 2, ..., n.

Сперва отметим несколько подмножеств $\mathcal{S}$, а также все подмножества этих подмножеств.

Затем снимем все отметки, если они есть, с нескольких подмножеств $\mathcal{S}$, а также со всех их подмножеств.

Найдите количество отмеченных подмножеств после всех этих операций.

Формат входных данных

В первой строке входного файла заданы через пробел три целых числа n, x и y. Следующие x строк содержат описания подмножеств, отмеченных на первом шаге, по одному на строке; также были отмечены все подмножества этих подмножеств. Наконец, последние y строк входного файла содержат описания подмножеств, с которых сняли отметки на втором шаге, по одному на строке; также были сняты отметки со всех их подмножеств. Описание каждого подмножества имеет вид $k\ a_1\ a_2\ \ldots\ a_k$, где k — количество элементов данного подмножества ($0 \leqslant k \leqslant n$), а a_i — сами элементы (a_i попарно различны, $1 \leqslant a_i \leqslant n$). Элементы могут быть перечислены в любом порядке.

Формат выходных данных

В первой строке выходного файла выведите одно число — количество отмеченных подмножеств после всех описанных операций.

Ограничения

- $1 \leqslant n \leqslant 10$
- $0 \leqslant x, y \leqslant 1000$

Примеры

marked.in	marked.out
1 1 1 1 1 0	1
2 0 1 2 2 1	0
3 2 1 2 1 2 2 2 3 2 1 3	3

Пояснения к примерам

В первом примере на первом шаге ставится отметка на подмножество {1} и на пустое подмножество, на втором шаге с пустого подмножества снимается отметка.

Во втором примере отметок нет.

В третьем примере на первом шаге отмеченными оказываются следующие шесть подмножеств: {}, {1}, {2}, {3}, {1, 2} и {2, 3}. На втором шаге снимаются отметки с трёх подмножеств {}, {1} и {3}.

Задача 9B. Отметки на подмножествах 2 [0.4 sec, 256 mb]

Условие такое же, как у задачи marked.
Файлы: `marked2.in`, `marked2.out`

Ограничения

- $1 \leqslant n \leqslant 20$
- $0 \leqslant x, y \leqslant 1000$

Задача 9C. Сумма «случайных» чисел [0.4 sec, 256 mb]

На столе лежат n шариков, на каждом шарике написано натуральное число. Валя и Витя используют эти шарики, чтобы получать «случайные» числа. Процедура получения «случайного» числа следующая. Сначала Валя берёт со стола некоторое непустое подмножество шариков; при этом необходимо, чтобы на столе остался хотя бы один шарик. Затем Витя также берёт какое-то непустое подмножество шариков, оставшихся на столе; после этого шариков на столе может не остаться. Наконец, Валя и Витя вычисляют «случайное» число $r = a \bmod b$, где a — это сумма чисел на шариках Вали, а b — сумма чисел на шариках Вити; $a \bmod b$ — это остаток от деления a на b. После этого все шарики возвращаются на стол.

Предположим, что Валя выбрала каждое допустимое подмножество шариков, и для каждого из них Витя выбрал по одному разу все допустимые подмножества оставшихся шариков. Найдите сумму всех «случайных» чисел, которые получились при этом.

Формат входных данных

В первой строке входного файла задано целое число n. Вторая строка содержит n целых чисел s_1, s_2, …, s_n через пробел; s_i — это число, написанное на i-м шарике.

Формат выходных данных

В первой строке выходного файла выведите одно число — сумму всех получившихся у Вали и Вити «случайных» чисел.

Ограничения

- $2 \leqslant n \leqslant 10$
- $1 \leqslant s_i \leqslant 1000$

Примеры

modsum.in	modsum.out
2 1 1	0
3 1 1 1	3
3 3 1 2	8

Пояснения к примерам

В первом примере у Вали и Вити получаются числа $a = b = 1$, а $1 \bmod 1 = 0$.

Во втором примере числа, отличные от нуля, получаются, только когда Валя берёт один любой шарик, а Витя — оба оставшихся.

В третьем примере суммируются следующие числа:

$$\begin{array}{lll} 0 = 3 \bmod 1 & 1 = 3 \bmod 2 & 0 = 3 \bmod (1+2) \\ 1 = 1 \bmod 3 & 1 = 1 \bmod 2 & 1 = 1 \bmod (3+2) \\ 2 = 2 \bmod 3 & 0 = 2 \bmod 1 & 2 = 2 \bmod (3+1) \\ 0 = (3+1) \bmod 2 & 0 = (3+2) \bmod 1 & 0 = (1+2) \bmod 3 \end{array}$$

Задача 9D. Сумма «случайных» чисел 2 [0.6 sec, 256 mb]

Условие такое же, как у задачи modsum.
Файлы: `modsum2.in`, `modsum2.out`

Ограничения

- $2 \leqslant n \leqslant 16$
- $1 \leqslant s_i \leqslant 1\,000\,000$

Задача 9E. Коммивояжёр возвращается! [2 sec, 256 mb]

Коммивояжёр возвращается в систему Альфы Центавра! Население системы с нетерпением ждёт его прибытия — каждый хочет приобрести что-нибудь с далёких планет!

Как обычно, коммивояжёр хочет минимизировать транспортные расходы. Он выбирает начальную планету, прилетает туда на межгалактическом корабле, после чего посещает все остальные планеты системы в порядке, минимизирующем суммарную стоимость посещения, и на другом межгалактическом корабле улетает обратно. Естественно, коммивояжёр не хочет летать ни на какую планету дважды.

Найдите оптимальный маршрут для коммивояжёра. Массы больше не могут ждать!

Формат входных данных

В системе Альфы Центавра n планет. Это число записано в первой строке входного файла ($1 \leqslant n \leqslant 19$). Следующие n строк содержат по n чисел каждая: j-ое число на i-ой из этих строк — стоимость перемещения a_{ij} от i-ой планеты до j-ой. Числа в каждой строке разделены пробелами. Числа a_{ii} не несут полезной информации. Все числа во входном файле положительны и не превосходят 10^8.

Формат выходных данных

В первой строке выходного файла выведите минимальную суммарную стоимость посещения всех планет. Во второй строке выведите n чисел через пробел — номера планет системы в порядке их посещения. Если оптимальных маршрутов несколько, можно вывести любой из них.

Пример

salesman.in	salesman.out
3 8 1 6 3 5 7 4 9 2	5 3 1 2

Замечание

Это очень простая задача. Решение за $\mathcal{O}(2^n n^2)$ без проблем зайдёт по времени.

Задача 9F. Гиперкуб [0.4 sec, 256 mb]

Гиперкуб — это обобщение понятия трёхмерного куба на N измерений. Нуль-мерным гиперкубом является точка, одномерным — отрезок, двумерным — квадрат. В общем же случае N-мерный гиперкуб — это правильный N-мерный многогранник, каждая из $2 \cdot N$ граней которого является $(N-1)$-мерным гиперкубом. Например, для $N = 2$ квадрат — это правильный многоугольник, каждая из $2 \cdot 2 = 4$ сторон которого — отрезок, то есть одномерный гиперкуб. Отметим, что N-мерный гиперкуб имеет 2^N вершин.

Старшеклассник Петя долго разбирался, что же такое гиперкуб, но наконец понял, как этот объект устроен, и ему настолько понравилось, что он даже придумал свою собственную игру на гиперкубе. Игра заключается в следующем.

Рассмотрим N-мерный единичный гиперкуб. Расположим его таким образом, чтобы одна из вершин находилась в начале координат — точке $(0, 0, \ldots, 0)$ в N-мерном пространстве, а для любой из остальных вершин каждая координата равнялась бы нулю или единице. В каждой из 2^N вершин запишем по целому неотрицательному числу. Игрок начинает свой путь в начале координат. За один ход можно переместиться из текущей вершины по любому ребру при условии, что сумма координат новой вершины строго больше суммы координат старой. Игра заканчивается, когда игрок попадает в вершину $(1, 1, \ldots, 1)$, имеющую максимальную сумму координат — N. Результат игры — сумма чисел во всех посещённых игроком вершинах. Цель игры — пройти по гиперкубу таким образом, чтобы эта сумма (количество очков за игру) оказалась как можно больше.

Петя довольно быстро понял, что между двумя вершинами гиперкуба A и B ребро есть тогда и только тогда, когда все координаты этих вершин $(A_1, A_2, \ldots, A_N)$ и $(B_1, B_2, \ldots, B_N)$ совпадают, кроме одной, которая равна нулю у одной из вершин (скажем, A) и единице у другой (B). Поскольку при этом $A_1 + A_2 + \ldots + A_N + 1 = B_1 + B_2 + \ldots + B_N$, то по такому ребру можно перемещаться из A в B, но не наоборот. Однако, сыграв в свою игру, Петя не может с уверенностью сказать, является ли полученная им сумма максимальной или можно на данном гиперкубе сыграть по-другому и набрать больше очков.

Напишите программу, которая по данному гиперкубу находит максимальную сумму, которую можно получить, сыграв в эту игру.

Формат входных данных

В первой строке входного файла записано число N ($1 \leqslant N \leqslant 10$) — размерность гиперкуба. В следующих 2^N строках содержится по одному числу в каждой; в $(k+2)$-ой строке записано C_k ($0 \leqslant C_k \leqslant 1000$) — число в вершине с номером k.

Номер вершины вычисляется так: вершина A с координатами $(A_1, A_2, \ldots, A_N)$ имеет номер, равный $A_1 \cdot 2^{N-1} + A_2 \cdot 2^{N-2} + \ldots + A_{N-1} \cdot 2 + A_N$, то есть координаты просто интерпретируются как двоичная запись номера вершины. В этой нумерации начальная вершина имеет номер 0, а конечная — номер $2^N - 1$.

Формат выходных данных

В выходной файл выведите одно число — максимальную сумму, которую можно получить при игре на данном гиперкубе.

Пример

cube.in	cube.out
3 1 2 3 4 5 6 7 8	21

Пояснение к примеру

Наш маршрут таков:

- вершина 0 (число 1, координаты (0, 0, 0)) — начальная
- вершина 4 (число 5, координаты (1, 0, 0))
- вершина 6 (число 7, координаты (1, 1, 0))
- вершина 7 (число 8, координаты (1, 1, 1)) — конечная

Наше количество очков: $1 + 5 + 7 + 8 = 21$.
Любой другой маршрут с соблюдением правил игры даёт меньшее количество очков.

Задача 9G. Самый длинный путь 22 [0.7 sec, 256 mb]

В данном ориентированном графе найдите самый длинный путь такой, что каждая вершина графа встречается в нём не более одного раза.

Формат входных данных

В первой строке входного файла заданы через пробел два целых числа n и m ($1 \leqslant n \leqslant 22$, $0 \leqslant m \leqslant 1000$). В следующих m строках заданы рёбра графа в формате u_i v_i — номера начальной и конечной вершин ребра i, соответственно. Граф может содержать петли и кратные рёбра.

Формат выходных данных

В первой строке выходного файла выведите длину искомого пути l. Во второй строке выведите $l + 1$ число через пробел — вершины пути в порядке обхода. Если оптимальных ответов несколько, можно вывести любой из них.

Примеры

path.in	path.out
3 3 1 2 2 3 3 1	2 1 2 3
4 6 1 2 2 1 2 3 2 4 3 2 4 2	2 1 2 4
5 3 3 2 2 2 1 5	1 3 2

Замечание

Предполагается решение за $\mathcal{O}(2^n n)$. Перебор с отсечениями работает ещё быстрее ;)

Задача 9H. Сумма всего подряд [0.5 sec, 256 mb]

Дан случайный граф. Нужно для каждого множества вершин A посчитать $f(A)$, количество независимых подмножеств вершин B: $B \subseteq A$. Множество вершин B называется независимым, если в графе нет ребра, оба конца которого лежат в множестве B.

Формат входных данных

На первой строке число вершин n ($1 \leqslant n \leqslant 23$) и число ребер $m \geqslant 1$.
Следующие m строк содержат пары чисел от 1 до n — ребра графа.
В графе нет ни петель, ни кратных ребер.

Формат выходных данных

Каждому множеству A можно сопоставить целое число $b(A)$, двоичная запись которого соответствует наличию элементов в множестве A. Пример: $n = 5, A = \{1, 2, 5\}, b(A) = 2^0 + 2^1 + 2^4 = 19$. Выведите $\sum\limits_A f(A)2^{b(A)} \mod (10^9 + 7)$

subsetsum.in	subsetsum.out
3 1 1 2	1221

Пояснение к примеру

Независимыми являются множества вершин $\{\}, \{1\}, \{2\}, \{3\}, \{1, 3\}, \{2, 3\}$.

$A = \{\}$ $f(A) = 1$ $b(A) = 0$
$A = \{1\}$ $f(A) = 2$ $b(A) = 2^0 = 1$
$A = \{2\}$ $f(A) = 2$ $b(A) = 2^1 = 2$
$A = \{1, 2\}$ $f(A) = 3$ $b(A) = 2^0 + 2^1 = 3$
$A = \{3\}$ $f(A) = 2$ $b(A) = 2^2 = 4$
$A = \{1, 3\}$ $f(A) = 4$ $b(A) = 2^0 + 2^2 = 5$
$A = \{2, 3\}$ $f(A) = 4$ $b(A) = 2^1 + 2^2 = 6$
$A = \{1, 2, 3\}$ $f(A) = 6$ $b(A) = 2^0 + 2^1 + 2^2 = 7$

$1 \cdot 2^0 + 2 \cdot 2^1 + 2 \cdot 2^2 + 3 \cdot 2^3 + 2 \cdot 2^4 + 4 \cdot 2^5 + 4 \cdot 2^6 + 6 \cdot 2^7 = 1221$

Замечание

Предполагается решение за $\mathcal{O}(2^n)$.

Задача 9I. Крышки [1.5 sec, 256 mb]

У программиста Петрова есть хобби — собирать крышки от пивных бутылок. Ничего странного, он знает еще с сотню программистов, которые очень уважают пиво. Да, они тоже собирают крышки. Не все из них, конечно, но некоторые. Если честно, то часть своих крышек он просто купил, уже без бутылок. Да, это не совсем спортивно, зато коллекция теперь более солидная. Одна вот беда — не хватает ему для полноты коллекции еще нескольких редких крышек. Он даже нашел в Интернете программистов, которые согласны продать их ему. Некоторые даже продают крышки сразу наборами, с большой скидкой. Почему продают, да еще со скидкой? А вы попробуйте объяснить жене, какая польза от пивных крышек. Она же не программист. Осталось выбрать оптимальные предложения. Если объяснить жене зачем надо хранить крышки еще возможно, то почему на них надо тратить деньги — точно не поймет. Поэтому надо купить как можно дешевле. Петров выписал на бумажку все варианты и задумался. Купить сразу все не получится, никаких денег не хватит. Тогда надо купить самые необходимые, но подешевле. Да уж, без программы тут не обойдешься...

Формат входных данных

В первой строке записано число N — количество недостающих Петрову крышек ($1 \leqslant N \leqslant 20$). Далее идет N строк — цена, за которую можно купить эту крышку, если покупать ее отдельно. В следующей строке стоит число M ($0 \leqslant M \leqslant 100$) — количество предложений по продаже наборов, содержащих нужные ему крышки. Далее идет M строк описывающих эти наборы. В каждой строке первое число — цена набора, второе — количество крышек в наборе, далее перечислены номера крышек (каждый номер от 1 до N), которые в этот набор входят. Номера в наборе не повторяются. Все цены — положительные числа, не превосходящие 2000. В последней строке перечислен минимальный набор крышек, который Петров намерен купить в любом случае.

Формат выходных данных

Выведите минимальную сумму, необходимую Петрову, чтобы купить все крышки из приведённого в последней строке набора.

Примеры

bottletaps.in	bottletaps.out
4 10 11 12 13 3 17 2 1 3 25 3 2 3 4 15 2 3 4 3 1 3 4	25

Замечание

$\mathcal{O}(2^N M)$ пройдёт. Переборное решение работает быстрее ;)

Задача 9J. Быстрое пересечение множеств [1.5 sec, 256 mb]

Даны N множеств. Множества занумерованы целыми числами от 1 до N. Для каждого множества $i = 1..N$ нужно найти такое множество $j = 1..N, j \neq i$, что их непохожесть минимальна. Непохожестью двух множеств A и B называется количество элементов, присутствующих ровно в одном из множеств A и B.

Формат входных данных

На первой строке целое число N от 2 до 10^4 — количество множеств. Далее собственно множества. Каждое множество задается следующим образом: сперва целое число k от 0 до 32 — размер множества, далее k целых чисел от 0 до 31 — элементы множества. Все элементы множества различны.

Формат выходных данных

Выведите N строк, в i-й строке выведите номер j — номер множества, которое вы считаете наименее непохожим на i-е), и собственно "непохожесть" данных множеств. Если для некоторого i существует несколько оптимальных j, выведите любое.

Пример

intersectsets.in	intersectsets.out
6 6 1 2 3 4 5 6 4 1 2 3 4 4 1 2 3 4 6 0 1 2 3 4 5 4 31 30 29 28 3 1 30 31	2 2 3 0 2 0 1 2 6 3 5 3

Замечание

Вы же помните, что с маленькими множествами все операции делаются за $\mathcal{O}(1)$?

Дополнительные задачи

Задача 9K. Справедливый дележ [1.5 sec, 256 mb]

— Я хотел честно, — сказал Балаганов, собирая деньги с кровати, — по справедливости.

В коробке от сигарет «Кавказ», отнятой у Корейко, было всего $1 \le N \le 15$ купюр, каждая номиналом $1 \leqslant a_i \leqslant 10^3$. Разделить надо было на $1 \le K \le 100$ частей, причём известно, что общая сумма делится на K. Метод деления «по справедливости» следующий: если разделить поровну не получается, то делить следует так, чтобы среднеквадратичное отклонение было минимальным.

Среднеквадратичное отклонение для данного способа дележа определяется следующим образом. Пусть i-й человек получил сумму денег, равную S_i. Обозначим за M среднее арифметическое сумм денег, полученных каждым: $M = (\sum_{i=1}^{K} S_i)/K$. Тогда среднеквадратичное отклонение можно вычислить так: $\sigma = \sqrt{(\sum_{i=1}^{K}(S_i - M)^2)/K}$.

— И как? — поинтересовался Остап.

— Сложно... — вздохнул Шура.

Попробуйте и вы разделить деньги «по справедливости».

Формат входных данных

В первой строке входного файла задано N — количество купюр и K — количество участников дележа. Далее, в следующей строке, заданы N целых чисел a_i — номиналы купюр.

Формат выходных данных

В первой строке выведите искомое среднеквадратичное отклонение с точностью до шести знаков после десятичной точки, в следующей строке выведите N чисел от 1 до K — номер участника дележа, которому досталась i-я купюра. Если оптимальных решений несколько, разрешается выводить любое.

Пример

fair.in	fair.out
4 3 1 2 3 6	1.414214 2 2 1 3

Замечание

Существует решение за $\mathcal{O}(3^n)$, но за $\mathcal{O}(3^n n)$ тоже зайдёт.

Задача 9L. Covering Points [2 sec, 256 mb]

Сегодня, Вася отправляется в научно-исследовательский институт сфер и кругов (RISC). Этот институт изучает все задачи, связанные со сферами и кругами, и теперь Васю просят что-то сделать с кругами.

Даны N точек на плоскости, необходимо покрыть их все K кругами минимально возможного радиуса. Кругам разрешено касаться и пересекаться друг с другом. Не могли бы вы помочь Васе справиться с заданием?

Формат входных данных

Входной файл состоит из одного или нескольких тестов. Каждый тест начинается со строки, содержащей два целых числа N и K ($1 \leqslant N \leqslant 15$, $1 \leqslant K \leqslant N$), а затем N строк, каждая из которых описывает одну точку её целыми координатами x_i и y_i. Точки могут совпадать. Все координаты не превышают 1000 по абсолютной величине. Ввод завершается тестом $N = K = 0$, который учитывать не нужно. Общая сумма чисел N по всем тестам в одном входном файле не превышает 150.

Формат выходных данных

Для каждого теста сначала выведете минимальный радиус, затем приведите пример покрытия. Если есть несколько решений — выводите любое. Радиусы нужно вывести с точностью не менее 6 знаков после запятой. Чекер будет осуществлять проверку "попала ли точка в круг" с точностью 10^{-6}. Мы рекомендуем выводить числа с максимальной точностью.

Заметьте, что круг с нулевым радиусом — точка.

Примеры

cover.in

```
3 2
0 0
0 1
0 2
3 1
0 0
0 1
1 0
0 0
```

cover.out

```
Case 1: The minimal possible radius is 0.5
  circle 1 at (0.0, 0.5)
  circle 2 at (0.0, 2.0)

Case 2: The minimal possible radius is 0.7071067811865476
  circle 1 at (0.5, 0.5)
```

Задача 9M. Отметки на подмножествах 3 [0.7 sec, 256 mb]

Условие такое же, как у задачи marked.
Файлы: `marked3.in`, `marked3.out`

Ограничения

- $1 \leqslant n \leqslant 26$
- $0 \leqslant x, y \leqslant 1000$

Задача 9N. Сумма «случайных» чисел 3 [1.5 sec, 256 mb]

Условие такое же, как у задачи modsum.
Файлы: `modsum3.in`, `modsum3.out`

Ограничения

- $2 \leqslant n \leqslant 18$
- $1 \leqslant s_i \leqslant 10$
- Сумма всех s_i не превосходит 50.

Задача 9O. Игра «Жизнь» [2 sec, 256 mb]

Что наша игра? Жизнь!

Дана клетчатая доска $W \times H$. Каждая клетка исходно является либо чёрной, либо белой. Клетки называются соседними, если у них есть общая сторона.

Повторяется следующая процедура:

- Находится белая клетка, у которой не менее двух чёрных соседей. Если такой клетки нет, процесс завершается.
- Найденная клетка перекрашивается в чёрный цвет.

Напишите программу, которая вычислит количество различных раскрасок доски, которые могут получиться по завершении процесса.

Поскольку ответ может быть большим, достаточно найти его остаток по модулю 10^{18}.

Формат входных данных

В первой строке ввода записаны размеры доски: два целых числа W и H ($1 \leqslant W, H \leqslant 13$). В следующих H строках записано по W символов, описывающих саму доску:

- Символ «`B`» обозначает, что клетка в начальной раскраске была чёрной.
- Символ «`.`» обозначает, что исходный цвет клетки неизвестен, то есть он может быть как чёрным, так и белым.

Формат выходных данных

Выведите одно целое число: остаток количества различных возможных конечных раскрасок по модулю 10^{18}.

Пример

life.in	life.out
3 4 B.B .B.	3

Пояснение к примеру

В данном примере возможны следующие конечные раскраски (здесь символом «`W`» обозначаются белые клетки):

```
BBB   BBB   BBB
BBB   BBB   BBB
WWW   BBB   BBB
WWW   WWW   BBB
```