

Задача А Монстры

Имя входного файла:

monsters.in

Имя выходного файла:

monsters.out

Максимальное время работы на одном тесте:

2 секунды

Максимальный объем доступной памяти:

8 мегабайт

В одной секретной лаборатории вывели новый вид маленьких монстров, размером чуть больше суслика. В ходе исследований ученые решили поставить следующий эксперимент. В центре комнаты устанавливается прямоугольный стол, поверхность которого разбита на $N \times M$ клеток размера 1×1 . В начальный момент времени на некоторых его клетках располагаются монстры, смотрящие параллельно сторонам стола. По команде экспериментатора монстры начинают двигаться по прямой в ту сторону, в которую они смотрят, доходят до края стола и спрыгивают на пол. Там их собирает лаборант Петя и относит в клетку.

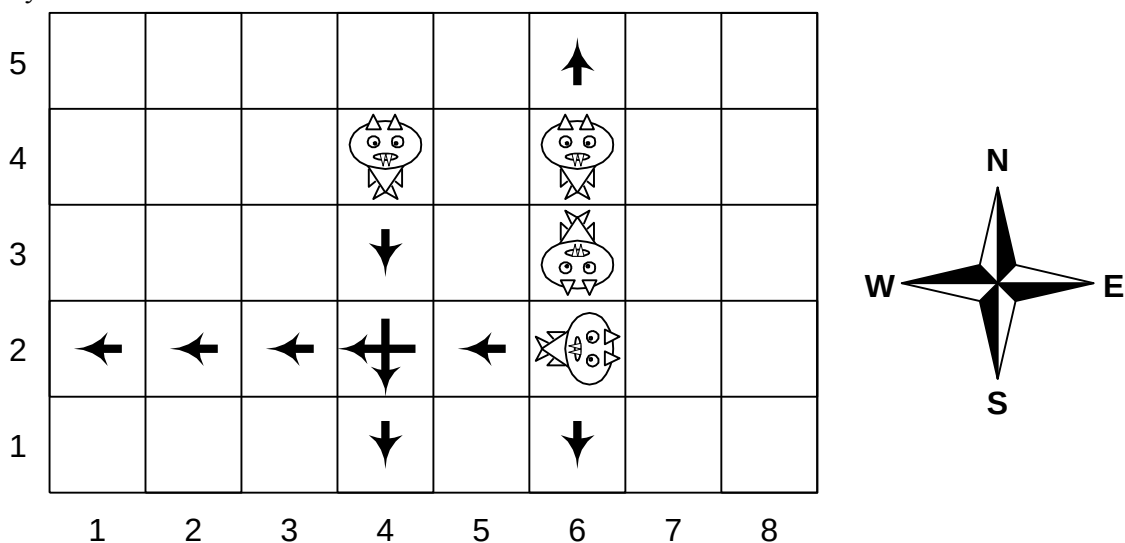


Рисунок 1. Монстры на столе для экспериментов

Поскольку у монстров очень грязные лапки, они оставляют следы на тех клетках, на которых они побывали. Так как отмывать стол придется лаборанту Пете, его заинтересовал вопрос – в каком количестве клеток побывают монстры. Помогите ему решить эту сложную задачу.

Формат входных данных

Первая строка входного файла содержит числа M и N – размеры лабораторного стола ($1 \leq M, N \leq 10^6$). Следующая строка содержит число K – количество монстров ($0 \leq K \leq 10^3$). Следующие K строк содержат описания монстров – два целых числа и один символ из множества $\{N, E, S, W\}$ – начальные координаты и направление соответствующего монстра (соответствие направлений и координат приведено на рисунке 1). Символ отделен от чисел ровно одним пробелом.

Формат выходных данных

В выходной файл выведите единственное число – количество клеток стола, на которых побывают монстры.

Пример

Пример соответствует расположению монстров, приведенному на рисунке 1.

monsters.in	monsters.out
8 5	13
4	
4 4 S	
6 2 W	
6 3 N	
6 4 S	

Задача В Форум

Имя входного файла: forum.in
Имя выходного файла: forum.out
Максимальное время работы на одном тесте: 2 секунды
Максимальный объем доступной памяти: 8 мегабайт

Клуб Юных Хакеров организовал на своем сайте форум. Форум имеет следующую структуру: каждое сообщение либо начинает новую тему, либо является ответом на какое-либо предыдущее сообщение и принадлежит той же теме.

После нескольких месяцев использования своего форума юных хакеров заинтересовал вопрос – какая тема на их форуме наиболее популярна. Помогите им выяснить это.

Формат входных данных

Первая строка входного файла содержит целое число N – количество сообщений в форуме ($1 \leq N \leq 1000$). Следующие строки содержат описание сообщений в хронологическом порядке.

Описание сообщения, которое представляет собой начало новой темы, состоит из трех строк. Первая строка содержит число 0. Вторая строка содержит название темы. Длина названия не превышает 30 символов. Третья строка содержит текст сообщения.

Описание сообщения, которое является ответом на другое сообщение, состоит из двух строк. Первая строка содержит целое число – номер сообщения, ответом на которое оно является. Сообщения нумеруются, начиная с единицы. Ответ всегда появляется позже, чем сообщение, ответом на которое он является. Вторая строка содержит текст сообщения.

Длина всех сообщений не превышает 100 символов.

Формат выходных данных

Выведите в выходной файл название темы, к которой относится наибольшее количество сообщений. Если таких тем несколько, то выведите первую в хронологическом порядке.

Пример

forum.in	forum.out
7 0 Олимпиада по информатике Скоро третья командная олимпиада? 0 Новая компьютерная игра Вышла новая крутая игра! 1 Она пройдет 24 ноября 1 В Санкт-Петербурге и Барнауле 2 Где найти? 4 Примет участие более 50 команд 6 Интересно, какие будут задачи	Олимпиада по информатике

Задача С Три города

Имя входного файла:

three.in

Имя выходного файла:

three.out

Максимальное время работы на одном тесте:

2 секунды

Максимальный объем доступной памяти:

8 мегабайт

В одном государстве имеется N городов. Некоторые города соединены дорогами, причем для любых двух городов A и B выполняется следующее свойство: существует ровно один способ попасть из города A в город B если можно перемещаться только по дорогам и не разрешается проезжать по одной и той же дороге более одного раза.

Недавно президента этой страны заинтересовал вопрос: какие три города являются наиболее удаленными друг от друга. А именно, назовем *взаимной удаленностью* друг от друга трех городов A , B и C минимальное количество дорог, которое необходимо использовать, чтобы доехать от A до B , затем от B до C и затем от C до A (при этом разрешается использовать одну и ту же дорогу в различных путешествиях).

Требуется найти три города, для которых взаимная удаленность друг от друга будет максимальной.

Например, для пяти городов, соединенных дорогами так, как это показано на рисунке 1, три наиболее удаленных друг от друга города – это города 1, 2 и 5 (взаимная удаленность равна $2 + 3 + 3 = 8$), а для городов на рисунке 2 – это любые три города, выбранные из множества $\{1, 2, 4, 5\}$ (удаленность $2 + 2 + 2 = 6$).

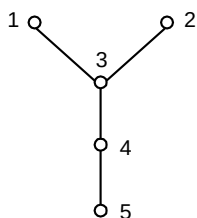


Рисунок 1. Пример городов, соединенных дорогами, решение – 1, 2, 5.

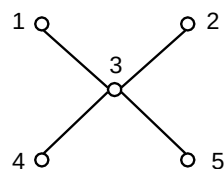


Рисунок 2. Пример городов, соединенных дорогами, решение неоднозначно.

Формат входных данных

Первая строка входного файла содержит число N – количество городов ($3 \leq N \leq 1000$). Следующие N строк содержат описания городов. Описание i -го города сначала содержит K_i – количество городов, с которыми он соединен дорогами ($1 \leq K_i < N$), а затем K_i чисел – номера городов, с которыми он соединен.

Гарантируется, что входные данные корректны – то есть если есть дорога из города A в город B , то есть и дорога из города B в город A , причем для всех пар городов выполняется свойство, указанное в условии задачи.

Формат выходных данных

Выведите в выходной файл три различных числа – номера трех наиболее удаленных друг от друга городов в произвольном порядке. Если решений несколько, выведите любое из них.

Примеры

three.in	three.out
5 1 3 1 3 3 1 2 4 2 3 5 1 4	1 2 5
5 1 3 1 3 4 1 2 4 5 1 3 1 3	1 2 4

Задача D Половинное деление

Имя входного файла: half.in
Имя выходного файла: half.out
Максимальное время работы на одном тесте: 2 секунды
Максимальный объем доступной памяти: 8 мегабайт

Рассмотрим выпуклый многоугольник, вершины которого лежат в точках плоскости с целыми координатами. Требуется разбить его на треугольники с вершинами в точках с целыми координатами, каждый из которых имел бы площадь $\frac{1}{2}$, либо выяснить, что это сделать невозможно.

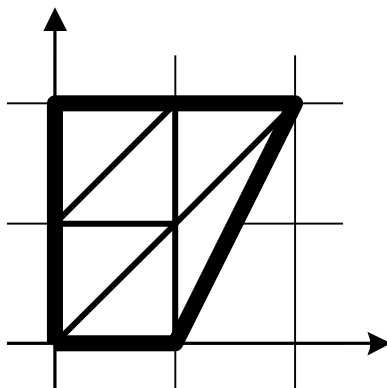


Рисунок 1. Разбиение многоугольника на треугольники с площадью $\frac{1}{2}$.

Формат входных данных

Первая строка входного файла содержит число N – количество вершин многоугольника ($1 \leq N \leq 10$). Следующие N строк содержат координаты вершин многоугольника в порядке обхода их по часовой стрелке. Все координаты – целые неотрицательные числа, не превышающие 10. Никакие три последовательные вершины многоугольника не лежат на одной прямой.

Формат выходных данных

Если выполнить разбиение указанным образом невозможно, выведите в выходной файл единственное число – 0.

В противном случае выведите несколько строк, содержащих по 6 чисел каждая. Количество строк должно быть равно количеству треугольников в найденном разбиении. Числа в каждой строке должны представлять собой координаты вершин соответствующего треугольника – $x_1, y_1, x_2, y_2, x_3, y_3$. Площадь каждого треугольника должна быть $\frac{1}{2}$. Порядок перечисления треугольников и вершин в каждом из треугольников может быть произвольным. Если допустимых разбиений несколько, выведите любое.

Пример

half.in	half.out
4	0 0 1 1 1 0
0 0	0 0 1 1 0 1
0 2	0 1 1 1 1 2
2 2	0 1 1 2 0 2
1 0	1 0 2 2 1 1
	1 1 2 2 1 2

Задача Е Дробь в L^AT_EX-е

Имя входного файла:	frac.in
Имя выходного файла:	frac.out
Максимальное время работы на одном тесте:	2 секунды
Максимальный объем доступной памяти:	8 мегабайт

Издательская система L^AT_EX предназначена для верстки сложных научно-технических текстов с большим количеством формул. Исходный файл для системы L^AT_EX пишется на языке T_EX и представляет собой текст документа, в который включены специальные символы и команды. Специальные символы и команды описывают размещение текста, в частности в математических формулах. Команда представляет собой последовательность латинских букв (регистр важен), перед которой стоит символ “\”. Так, команда \frac предназначена для описания дроби, в которой числитель расположен над знаменателем. Рассмотрим простейшую структуру команды \frac.

Команда \frac имеет два параметра – числитель и знаменатель. Перед самой командой не обязательно ставить пробел. Следом за ключевым словом \frac записываются числитель и знаменатель. Если числитель и знаменатель имеют длину более одного символа, они заключаются в фигурные скобки. Если числитель или знаменатель записываются одной буквой или цифрой, их можно не брать в фигурные скобки. Если числитель записывается одним символом, то он отделяется от \frac хотя бы одним пробелом. Если знаменатель записывается одним символом, то он не отделяется пробелом от числителя. Произвольное ненулевое количество пробелов считается синтаксически эквивалентным одному пробелу. Нельзя разделять пробелами на части ключевое слово \frac.

Дадим также формальное определение выражения для нашей задачи:

<выражение> ::= <элемент> | <элемент><выражение>
<элемент> ::= <дробь> | “{” <выражение> “}” | <другой математический элемент>
<дробь> ::= “\frac” <тело дроби>
<тело дроби> ::= <числитель><знаменатель>
<числитель> ::= <пробелы><непробельный символ> | [<пробелы>] “{” <выражение> “}”
<знаменатель> ::= <непробельный символ> | [<пробелы>] “{” <выражение> “}”
<другой математический элемент> ::= произвольная последовательность печатных символов, не содержащая фигурных скобок и подстроки “\frac”
<пробелы> ::= “ ” | “ ” <пробелы>
<непробельный символ> ::= произвольный печатный символ, за исключением “ ”, “\”, “{” и “}”

Здесь вертикальная черта | означает “или”, заключенная в квадратные скобки часть может отсутствовать, а символы, записанные в кавычках обозначают самих себя. Печатный символ – любой символ с ASCII кодом от 32 (пробел) до 127.

Например, выражение

$$\frac{a+b}{d+1} + \frac{a}{x} - \frac{2}{2+\frac{3}{y}}$$

записывается на языке T_EX как

`\frac{a+b}{d+1}+\frac{a}{x}-\frac{2}{2+\frac{3}{y}}`

Чтобы в печатаемом документе вывести формулу, необходимо вычислить ее высоту для используемого при печати шрифта. Шрифт определяет размеры S – высоту отдельного символа и D – высоту горизонтальной дробной черты. Значения S и D задаются целыми числами. Ваша задача – для заданного выражения на языке T_EX вычислить высоту формулы.

Отметим, что если две дроби принадлежат одному выражению, то их дробные черты записываются на одном уровне, а если нет (например, относятся к числителям или знаменателям различных дробей), это свойство может и не выполняться. Чтобы проиллюстрировать применение этого правила, приведем два примера:

`\frac{a+b}{\frac{cd}}{\frac{ef}{g+h}}`

$$\frac{\frac{a+b}{\frac{c}{d}}}{\frac{f}{g+h}}$$

`\frac{a+b+c}{\frac{\frac{de}{g+h}}{\frac{i+j+k}{\frac{l+m}{\frac{no}}{}}}}`

$$\frac{\frac{a+b+c}{\frac{d}{\frac{e}{g+h}}}}{\frac{i+j+k}{\frac{l+m}{\frac{n}{o}}}}$$

Формат входных данных

В первой строке находятся целые положительные числа S и D ($1 \leq S, D \leq 10000$). Следующая строка содержит описание формулы на T_EX-е, длина строки не более 200 символов. Гарантируется, что формула синтаксически корректна, то есть фигурные скобки образуют правильную скобочную последовательность и строка содержит только печатные символы. Все символы “\”, встречающиеся в строке относятся к некоторой командной последовательности (не обязательно `\frac`), можете считать, что все прочие командные последовательности задают символы, высота которых равна S . Числитель и знаменатель каждой дроби содержат хотя бы по одному символу, вся формула содержит хотя бы один символ.

Формат выходных данных

Выведите в выходной файл единственное число – высоту формулы.

Примеры

<code>frac.in</code>	<code>frac.out</code>
10 2 <code>\frac{a+b}{d+1}+\frac{ax}{-\frac{2}{2+\frac{3}{y}}}</code>	34
10 2 <code>no fractions here</code>	10
10 2 <code>\frac{\alpha}{\beta+\sin{2+x}}</code>	22
10 2 <code>\cos{\frac{\alpha}{b}}</code>	22
10 2 <code>\frac{a}{\sin{a}}</code>	22
10 2 <code>\frac{a+b}{\frac{cd}{\frac{ef}{g+h}}}</code>	46
10 2 <code>\frac{a+b+c}{\frac{\frac{de}{g+h}}{\frac{i+j+k}{\frac{l+m}{\frac{no}}{}}}}</code>	46

Задача F Столица

Имя входного файла: `capital.in`
Имя выходного файла: `capital.out`
Максимальное время работы на одном тесте: 2 секунды
Максимальный объем доступной памяти: 8 мегабайт

В некотором царстве, в некотором государстве было N городов, и все они, судя по главной карте императора, имели целые координаты. В те годы леса были дремучие, дороги же строить умели только параллельно осям координат, так что расстояние между двумя городами определялось как $|x_1 - x_2| + |y_1 - y_2|$.

Император решил построить $N + 1$ -ый город и сделать его столицей своего государства, при этом координаты столицы также должны быть целыми. Место для столицы следует выбрать так, чтобы среднее арифметическое расстояний между столицей и остальными городами было как можно меньше. Однако, разумеется, столицу нельзя строить на месте существующего города.

Нелегкая задача выбрать место для столицы поручена Вам.

Формат входных данных

Первая строка входного файла содержит число N – количество городов ($1 \leq N \leq 100$). Следующие N строк содержат координаты городов – пары целых чисел, не превышающих 1000 по абсолютной величине.

Формат выходных данных

Выведите в выходной файл два целых числа – координаты точки, где следует построить столицу. Если решений несколько, выведите любое.

Примеры

<code>capital.in</code>	<code>capital.out</code>
8 0 0 1 0 2 0 0 1 2 1 0 2 1 2 2 2	1 1
4 0 0 1 1 0 1 1 0	0 2

Задача G Перестановки

Имя входного файла: perm.in
Имя выходного файла: perm.out
Максимальное время работы на одном тесте: 2 секунды
Максимальный объем доступной памяти: 8 мегабайт

Ваня и Петя играют в следующую игру. Ваня пишет на бумаге какую-либо перестановку чисел от 1 до N (то есть выписывает все числа от 1 до N в некотором порядке), и расставляет на столе в ряд N предметов. После этого Петя переставляет предметы в соответствии с Ваниной перестановкой. А именно, Петя выполняет следующие действия: если i -ое число в Ваниной перестановке равно a_i , то Петя ставит предмет, который стоит на i -ом месте, на место с номером a_i .

Обозначим предметы числами от 1 до N . Тогда начальное расположение предметов можно обозначить последовательностью чисел $(1, 2, \dots, N)$. К примеру, если $N = 5$, то начальное расположение предметов есть $(1, 2, 3, 4, 5)$. Пусть Ваня написал перестановку $\langle 2, 5, 4, 3, 1 \rangle$. Это значит, что после перемещения предметов они окажутся расставлены в следующем порядке: $(5, 1, 4, 3, 2)$.

Однако, переставив предметы, Петя не останавливается на достигнутом и вновь переставляет их в соответствии с Ваниной перестановкой. Снова, если i -ое число в Ваниной перестановке равно a_i , то Петя ставит предмет, который стоит на i -ом месте на место с номером a_i . Так, если в приведенном выше примере повторно применить перестановку, предметы окажутся расположены в следующем порядке: $(2, 5, 3, 4, 1)$.

Таким образом, Петя переставляет предметы в соответствии с Ваниной перестановкой, пока их расположение не окажется таким же, как исходное. В нашем примере Пете потребуется сделать еще 4 действия, порядок предметов после каждого из них будет следующим: $(1, 2, 4, 3, 5)$, $(5, 1, 3, 4, 2)$, $(2, 5, 4, 3, 1)$, $(1, 2, 3, 4, 5)$. Всего Пете потребовалось применить перестановку 6 раз.

Добрый Ваня хочет, чтобы Пете пришлось выполнить как можно больше действий. Помогите ему выбрать соответствующую перестановку.

Формат входных данных

Входной файл содержит единственное целое число N – количество предметов ($1 \leq N \leq 100$).

Формат выходных данных

Выведите в выходной файл перестановку чисел от 1 до N , при которой количество действий, которое придется сделать Пете максимально. Если таких перестановок несколько, можно вывести любую.

Пример

perm.in	perm.out
5	2 5 4 3 1

Задача Н Военная сортировка

Имя входного файла: `sort.in`
Имя выходного файла: `sort.out`
Максимальное время работы на одном тесте: 2 секунды
Максимальный объем доступной памяти: 8 мегабайт

Фирма *Macrohard* получила заказ от армии одной страны на реализацию комплекса программного обеспечения для нового суперсекретного радара. Одной из наиболее важных подпрограмм в разрабатываемом комплексе является процедура сортировки.

Однако в отличие от обычной сортировки, эта процедура должна сортировать не произвольный массив чисел, который передается ей на вход, а специальный заранее заданный массив из N чисел, в котором записана некоторая фиксированная перестановка чисел от 1 до N , и кроме того, ни одно число в нем изначально не находится на своем месте (то есть на позиции с номером i изначально не находится число i).

В связи с повышенными требованиями к надежности комплекса в процессе сортировки разрешается выполнять единственную операцию – менять местами два соседних элемента массива. Кроме того, в связи с необходимостью соответствия уставу, не разрешается изменять положение числа, которое уже находится на своем месте.

Например, если массив из 6 элементов в некоторый момент имеет вид $\langle 2, 1, 3, 6, 4, 5 \rangle$, то можно поменять местами 1 и 2, 6 и 4 или 4 и 5, а менять местами 1 и 3 или 3 и 6 нельзя, поскольку число 3 находится на своем месте (на позиции с номером 3).

Вам дан входной массив и поставлено важное задание. Найти последовательность обменов (не обязательно кратчайшую), сортирующую массив и удовлетворяющую приведенным условиям.

Подсказка

Найти такую последовательность обменов всегда возможно.

Формат входных данных

Первая строка входного файла содержит целое число N – размер входного массива ($1 \leq N \leq 100$). Вторая строка содержит N целых чисел – исходную перестановку чисел от 1 до N в массиве. Изначально ни одно число не стоит на своем месте.

Формат выходных данных

Выведите в выходной файл K строк, где K – количество обменов в Вашей сортировке. На каждой строке выведите по два числа x_i и y_i , разделенных пробелом – позиции в массиве, числа на которых следует поменять местами на i -ом обмене. Помните, что должно выполняться условие $|x_i - y_i| = 1$ и что нельзя перемещать число, которое уже стоит на своем месте.

Пример

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

В приведенном примере массив последовательно имеет следующий вид:

исходный вид массива	2 3 1 6 4 5
поменяли местами числа на 2 и 3 позициях	2 1 3 6 4 5
поменяли местами числа на 1 и 2 позициях	1 2 3 6 4 5
поменяли местами числа на 4 и 5 позициях	1 2 3 4 6 5
поменяли местами числа на 5 и 6 позициях	1 2 3 4 5 6

Задача I Космический мусорщик

Имя входного файла: trash.in
Имя выходного файла: trash.out
Максимальное время работы на одном тесте: 2 секунды
Максимальный объем доступной памяти: 8 мегабайт

В околоземном космическом пространстве накопилось много мусора, поэтому ученые сконструировали специальный аппарат – ловушку для космического мусора. Для того, чтобы хорошо собирать мусор, этот аппарат должен двигаться по достаточно сложной траектории, сжигая собранный по пути мусор. Ловушка может передвигаться в пространстве по 6 направлениям: на север (N), на юг (S), на запад (W), на восток (E), вверх (U) и вниз (D). Движением ловушки управляет процессор. Программа движения задается шестью правилами движения, которые соответствуют каждому из указанных направлений. Каждое такое правило представляет собой строку символов из множества {N, S, W, E, U, D}.

Команда ловушки есть пара из символа направления и параметра – целого положительного числа M . При исполнении такой команды ловушка в соответствии со своей программой выполняет следующее. Если параметр больше 1, то она перемещается на один метр в направлении, которое указано в команде, а затем последовательно выполняет команды, заданные правилом для данного направления, с параметром меньше на 1. Если же параметр равен 1, то просто перемещается на один метр в указанном направлении.

Пусть, например, заданы следующие правила:

Направление	Правило
N	N
S	NUSDDUSE
W	UEWWD
E	
U	U
D	WED

Тогда при выполнении команды S(3) мусорщик выполнит следующие действия:

- 1) переместится на 1 метр в направлении S
- 2) выполнит последовательно команды N(2), U(2), S(2), D(2), D(2), U(2), S(2), E(2).

Если далее проанализировать действия мусорщика при выполнении команд из пункта 2, получим, что в целом он совершит следующие перемещения:

SNNUUSNUSDDUSEDWEDDWEDUUSNUSDDUSEE

По заданной команде определите, какое общее количество перемещений на один метр совершит ловушка при выполнении заданной команды. В приведенном примере это количество равно 34.

Формат входных данных

Первые шесть строк входного файла задают правила для команд с направлением N, S, W, E, U и D соответственно. Каждая строка содержит не более 100 символов (и может быть пустой). Следующая строка содержит команду ловушки: сначала символ из множества {N, S, W, E, U, D}, затем пробел и параметр команды – целое положительное число, не превышающее 100.

Формат выходных данных

Выведите в выходной файл единственное число – количество перемещений, которое совершит ловушка. Гарантируется, что ответ не превышает 10^9 .

Пример

trash.in	trash.out
N NUSDDUSE UEWWD U WED S 3	34