Задача A. Оцените пассажиропоток

Имя входного файла:	bus.in
Имя выходного файла:	bus.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

Маршрут автобуса проходит через N остановок (включая конечные). Отдел по исследованию пассажиропотоков записал данные о том, сколько человек выходило и сколько садилось в автобус на каждой остановке. Напишите программу, которая по этим данным определит, какое максимальное количество человек одновременно в этот рейс ехало в автобусе.

Формат входных данных

Во входном файле записано сначала число N ($2 \le N \le 100$) – количество остановок на маршруте. Далее задается количество человек, севших в автобус на конечной. Далее идет (N-2) пары чисел, задающих для промежуточных остановок количество вышедших и количество вошедших пассажиров. Наконец, идет число, задающее количество вышедших из автобуса на конечной остановке.

Количество вошедших пассажиров на каждой остановке не превышало 100. Данные корректны, в частности, суммарное количество вошедших в автобус на всех остановках пассажиров всегда равно суммарному количеству вышедших.

Формат выходных данных

В выходной файл выведите одно целое число — максимальное количество человек, которые в какой-то момент одновременно ехали в автобусе.

Примеры

bus.in	bus.out	Комментарии
5 10 3 1 5 10 0 2 15	15	На конечной в автобус село 10 человек. Далее 3 вышло и 1 зашел. В автобусе стало 8 человек. На следующей остановке вышло 5 и зашло 10. Стало 13 человек. На последней промежуточной остановке никто не вышел, а зашло 2 человека. На конечной вышло 15 человек. Итого максимальное количество – 15.
5 10 10 0 0 0 0 10 10	10	На конечной село 10 человек, которые на следующей остановке вышли. После этого на следующей остановке никто не сел и никто не вышел. Дальше опять село 10 человек, которые доехали до конечной.
5 0 0 9 3 4 2 0 8	10	С конечной автобус отправился без пассажиров. Дальше в него село 9 человек. На следующей остановке 3 вышли и зашли 4. В автобусе стало 10 человек. На следующей остановке 2 вышли и никто не зашел. 8 человек доехало до конечной. Максимальное количество пассажиров составляло 10
3 100 0 100 200	200	На начальной и на промежуточной остановку в автобус село по 100 человек. Вышло из автобуса 200 человек.

Задача B. Найдите последовательность

Имя входного файла:	words.in
Имя выходного файла:	words.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

Вася и Петя играют в следующую игру. Они взяли некоторую последовательность символов и дальше получают из нее новые последовательности, отбрасывая несколько первых символов исходной последовательности (разрешается в том числе не отбрасывать ни одного символа, но не разрешается отбрасывать сразу все символы). Каждый называет по одной такой последовательности. Выигрывает тот, чья последовательность будет идти раньше в алфавитном порядке.

Напомним, что если мы сравниваем две последовательности, и у них первые K символов совпадают, а (K+1)-е символы отличаются, то раньше будет идти по алфавиту та, в которой (K+1)-й символ идет раньше по алфавиту. Если же одна последовательность является началом другой, то раньше по алфавиту идет более короткая из них.

Напишите программу, которая по данной последовательности определит, что нужно назвать Васе, чтобы не проиграть Пете.

Формат входных данных

Во первой строке входного файла записано число N — длина исходной последовательности ($1 \le N \le 250$). Во второй строке идет сама последовательность. Последовательность состоит только из заглавных латинских букв.

Формат выходных данных

В выходной файл выведите выигрышную последовательность.

Примеры

words.in	words.out	Комментарии
4 MAMA	A	Рассматриваются строки MAMA, AMA, MA, A. Выигрышная строка A
4 ALLO	ALLO	Выигрышной является исходная строка
5 BBABB	ABB	

Задача C. Проверьте правильность ситуации

Имя входного файла:	xzeros.in
Имя выходного файла:	xzeros.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

Напишите программу, которая по изображению поля для игры в «Крестики-нолики» определит, могла ли такая ситуация возникнуть в результате игры с соблюдением всех правил.

Напомним, что игра в «Крестики-нолики» ведется на поле 3×3. Два игрока ходят по очереди. Первый ставит крестик, а второй – нолик. Ставить крестик и нолик разрешается в любую еще не занятую клетку поля. Когда один из игроков поставит три своих знака в одной горизонтали, вертикали или диагонали, или когда все клетки поля окажутся заняты, игра заканчивается.

Формат входных данных

Вводится три строки по три числа в каждой, описывающих игровое поле. Число 0 обозначает пустую клетку, 1 – крестик, 2 – нолик. Числа в строке разделяются пробелами.

Формат выходных данных

Требуется вывести слово YES, если указанная ситуация могла возникнуть в ходе игры, и NO в противном случае.

Примеры

xzeros.in	xzeros.out
1 1 1 1 1 1 1 1 1	NO
2 1 1 1 1 2 2 2 1	YES
1 1 1 2 0 2 0 0 0	YES
0 0 0 0 1 0 0 0 0	YES
1 1 1 2 2 2 0 0 0	NO

Задача D. Наберите текст

Имя входного файла:	phone.in
Имя выходного файла:	phone.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

В новой модели сотового телефона разработчики предусмотрели только 5 кнопок: «стрелка вверх», «стрелка вниз», «стрелка вправо», «стрелка влево» и «Ok». Чтобы набрать номер телефона, на экран выводится виртуальная клавиатура в виде такой таблицы:

1	2	3
4	5	6
7	8	9
выход	0	звонок

По ячейкам этой таблицы пользователь телефона может перемещать курсор вверх, вниз, вправо или влево. Для выбора цифры необходимо нажать кнопку «Ok». После набора всех цифр пользователь должен подвинуть курсор на ячейку «звонок» и нажать «Ok». Таблица не «зациклена» (это означает, что нажатие кнопки «вправо», например, из ячейки с цифрой 6 не приведет к перемещению курсора).

При вызове экранной клавиатуры курсор находится на кнопке с надписью «выход».

По заданному семизначному номеру телефона определите, сколько нажатий кнопок должен сделать пользователь, чтобы позвонить по этому номеру.

Формат входных данных

Во входном файле задан номер телефона, содержащий ровно 7 цифр.

Формат выходных данных

В выходной файл выведите одно число — минимальное количество нажатий суммарно на все кнопки, которое необходимо, чтобы совершить звонок по этому номеру.

Примеры

phone.in	phone.out	Пояснение
0800000	12	Это кнопки: вправо, Ok, вверх, Ok, вниз, Ok, Ok, Ok, Ok, Ok, вправо, Ok
1234567	24	

Задача E. Введите одностороннее движение

Имя входного файла:	oneway.in
Имя выходного файла:	oneway.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

В Тридевятом Царстве было N городов, некоторые из которых были соединены дорогами. На 1 апреля Министр транспорта решил все дороги сделать односторонними, т.е. разрешить проезд по каждой дороге только в одном направлении, при этом сделав так, чтобы, однажды уехав из любого города, вернуться в него уже было невозможно вплоть до 2 апреля (когда двухстороннее движение будет восстановлено).

Помогите Министру ввести такое одностороннее движение.

Формат входных данных

Во входном файле записано сначала число N — количество городов ($1 \le N \le 100$). Затем записано число M — количество дорог ($1 \le M \le 10000$). Далее идет M пар чисел, задающих дороги (каждая дорога описывается номерами городов, которые она соединяет). Не бывает дорог из некоторого города в тот же город. Между двумя городами может быть несколько дорог.

Формат выходных данных

В выходной файл нужно выдать M пар чисел, соответствующих дорогам (дороги должны быть выданы в том же порядке, в котором они заданы во входном файле). Для каждой дороги сначала должен быть записан номер города, из которого по ней можно будет уехать после введения одностороннего движения, а затем — номер города, куда эта дорога ведет. Если решений несколько, выведите любое из них.

Если невозможно ввести одностороннее движение так, чтобы уехав из любого города в него уже нельзя было вернуться, выходной файл должен содержать одно число 0.

Примеры

oneway.in	oneway.out
4 6 1 2 1 2 2 3 2 4 4 3 1 4	1 2 1 2 2 3 2 4 4 3 1 4
2 1 1 2	2 1

Задача F. Округлите равенство

Имя входного файла:	round.in
Имя выходного файла:	round.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

Дано верное равенство вида $a_1+a_2+...+a_N=b_1+b_2+...b_M$, где $a_1,a_2,...,a_N,,b_1,b_2,...,b_M$ – некоторые действительные (не обязательно целые) числа. Требуется «округлить» это равенство, т.е. получить новое верное равенство $c_1+c_2+...+c_N=d_1+d_2+...+d_M$, где $c_1,c_2,...,c_N,d_1,d_2,...,d_M$ — целые числа, и при этом c_1 получено округлением числа a_1 до целого вверх или вниз (так, например, число 1.7 разрешается округлить как до 1, так и до 2), c_2 получено округлением a_2, …, c_N – округлением a_N, d_1 – округлением b_1, …, d_M – округлением b_M. Если какое-то из чисел в исходном равенстве было целым, оно должно остаться без изменений.

Формат входных данных

Во входном файле задано сначала число N, затем N чисел $a_1, a_2, ..., a_N$, затем число M, затем числа $b_1, b_2, ..., b_M$. Каждое число задается на отдельной строке. M и N – натуральные числа, не превышающие 1000. Остальные числа — вещественные, каждое из них по модулю не превышает 1000 и содержит не более 6 цифр после десятичной точки. При этом $a_1+a_2+...+a_N=b_1+b_2+...b_M$.

Формат выходных данных

Если «округлить» равенство можно, то в выходной файл выведите сначала числа $c_1,c_2,...,c_N$, а затем числа $d_1,d_2,...,d_M$. Все числа должны быть целыми и выведены без десятичной точки. Числа должны разделяться пробелами или переводами строки. Если решений несколько, выведите любое из них.

Если округлить исходное равенство до верного целочисленного равенства невозможно, выведите одно число 0.

Примеры

round.in	round.out	Комментарии
3 0.15 -3.000 2.7 1 -0.15	0 -3 2 -1	Обратите внимание, что число –3 может округляться только в –3, в то время как 0.15 можно округлить как до 0, так и до 1, 2.7 – до 2 или до 3, –0.15 – до –1 или до 0. Приведенное решение не является единственным: так же верным является, например, такое округление: 1+(–3)+2=0
2 1.7 2.5 3 1 2.000 1.20	1 3 1 2 1	Приведенное решение 1+3=1+2+1 не является единственным. Верными ответами также являются 2+2=1+2+1 и 2+3=1+2+2.
1 0.5 1 0.5	1 1	Здесь верными являются как ответ 1=1, так и 0=0.

Задача G. Выложите побольше карт

Имя входного файла:	cards.in
Имя выходного файла:	cards.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

Вася и Петя играют в следующую игру. Они берут колоду из 36 карточек. На каждой карточке написано число от 1 до 9 и каждая карточка покрашена в один из 4 цветов так, что есть ровно по 9 карточек каждого цвета, и они пронумерованы числами от 1 до 9. Карты перемешиваются, и игрокам раздается по несколько (не более чем по 18) карточек.

Дальше игроки по очереди делают ходы. За один ход игрок может выложить на стол одну или последовательно несколько карточек по следующим правилам. Карточку с цифрой 5 любого цвета можно выкладывать на стол без дополнительных условий. Карточку с другой цифрой можно выкладывать только если на стол уже выложена карточка того же цвета, на которой написано число на 1 большее или на 1 меньшее, чем на данной, или же карточка с той же цифрой, но другого цвета (не важно, была ли эта карточка выложена вами или вашим противником, и была ли она выложена на предыдущем ходе или раньше). Если ни одну карточку игрок выложить не может, он пропускает ход.

Напишите программу, которая по информации о том, какие карточки уже лежат на столе и какие есть у игрока, определит наибольшее число карточек, которое может быть выложено этим игроком на данном ходе.

Формат входных данных

Во входном файле записано сначала число K — количество уже выложенных на стол карточек. Далее идет K пар чисел, описывающих эти карточки. Затем записано число N — количество карточек на руках у игрока, который сейчас должен делать ход. Далее записано N пар чисел, описывающих эти карточки.

Каждая карточка описывается двумя числами — номером цвета (от 1 до 4) и цифрой, которая написана на карточке (от 1 до 9).

Ограничения: $0 \le K \le 35$, $1 \le N \le 36$, $N+K \le 36$, все карточки различны.

Формат выходных данных

В выходной файл выведите одно число — наибольшее количество карточек, которые могут быть выложены на данном ходе.

Примеры

cards.in	cards.out	Пояснения
2 1 5 1 4 3 1 3 1 6 2 8	2	Это карты 1 3 (потому что на столе есть 1 4) и 1 6 (потому что на столе есть 1 5)
0 4 2 8 1 5 3 6 1 6	3	Первым ходом можно выложить 1 5, после этого мы имеем право выложить и 1 6, после которой выкладываем 3 6
3 1 4 1 5 1 6 2 2 8 2 9	0	Нельзя выложить ни одной карточки

Задача H. Определите порядок действий

Имя входного файла:	order.in
Имя выходного файла:	order.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

Ученику второго класса рассказали правила, как нужно выполнять арифметические действия, чтобы вычислить значение арифметического выражения, состоящего из чисел, скобок и знаков арифметических операций + (сложение) и * (умножение). После этого ему дали упражнения — несколько задач, в которых требуется расставить порядок выполнения действий. Помогите ему.

Правила вычисления выражения, рассказанные ученику, звучат так. Если в выражении вообще нет скобок, то сначала выполняются все операции умножения слева направо, а затем — операции сложения также слева направо. Если же в выражении есть скобки, то находится самая левая закрывающая скобка и соответствующая ей открывающая. Выражение между ними не содержит скобок и может быть вычислено по вышеописанным правилам. Дальше это выражение (вместе со скобками) мысленно удаляется из выражения и заменяется числом – результатом. Если в выражении остались скобки, то процедура повторяется.

Напишите программу, которая для корректного выражения будет определять порядок выполнения арифметических действий. Поскольку сами числа в этой задаче нам будут не существенны, мы заменим их на знаки #.

Формат входных данных

Во входном файле записана одна строка, состоящая из символов #, +, *, (,). Длина строки не превышает 250 символов. Строка соответствует правильному арифметическому выражению.

Формат выходных данных

В выходной файл нужно вывести ту же строку, заменив знаки операций + и * в ней натуральными числами, задающими порядок выполнения действий в соответствии с описанными правилами.

Примеры

order.in	order.out
#+#*#	#2#1#
#+#+(#+#)	#2#3(#1#)
#+(#+#*#)*#+#	#4(#2#1#)3#5#
#+#+#+#+#+#+#+#+#+#	#1#2#3#4#5#6#7#8#9#10#
#+#+(#+(#+#))+(#+#)	#4#5(#2(#1#))6(#3#)

Задача I. Соберите числа

Имя входного файла:	bricks.in
Имя выходного файла:	bricks.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

У Васи в распоряжении оказался набор кубиков. Вася решил на каждой грани каждого кубика написать по цифре и дальше использовать кубики для того, чтобы складывать из них числа. Вася хочет написать цифры так, чтобы уметь складывать любое число от 1 до некоторого числа K. Посчитайте такое максимальное K, до которого Вася сможет выкладывать все числа, если в распоряжении у Васи оказалось N кубиков. Заметьте, что если на какой-то грани какого-то кубика написана цифра 6, то эту же грань можно использовать и как цифру 9, просто перевернув соответствующий кубик.

При выкладывании числа Вася не обязан использовать все кубики. Ведущие нули в числах не нужны.

Рассмотрим примеры.

Пусть N=1. Тогда, написав на гранях кубика цифры от 1 до 6, Вася сможет выкладывать числа от 1 до 6. Тем самым, K=6.

Пусть N=2. Тогда, написав на гранях одного кубика цифры от 1 до 6, а на гранях другого цифры 0, 1, 2, 3, 7, 8, Вася сможет выложить любое число от 1 до 43.

Формат входных данных

Во входном файле записано одно число N ($1 \le N \le 13$).

Формат выходных данных

В выходной файл выведите максимальное значение K такое, что имея N кубиков Вася может так написать на их гранях цифры, чтобы , было возможно выложить любое число от 1 до K.

Примеры

bricks.in	bricks.out
1	6
2	43