Отборочный-1 10-11

Распаковка числа

Рассмотрим способ упакованной записи чисел в двоичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 1111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в двоичной системе и закрывающей скобки –]. Так число 111111100 в упакованной записи будет записано следующим образом: 1[1[1[10]]]0[10]
Распакуйте строку 1[11]0[1]0[1[11]]10[10[11]1]1[1011]0 в запись двоичного числа.

Распаковка числа

Рассмотрим способ упакованной записи чисел в троичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 111111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в троичной системе и закрывающей скобки –]. Так число 11111111000 в упакованной записи будет записано следующим образом: 1[2[2]]0[10]
Распакуйте строку 1[11]2[2[2]]1[1]0[12]1[1[10]]0[1] в запись троичного числа.

Распаковка числа

Рассмотрим способ упакованной записи чисел в двоичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 1111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в двоичной системе и закрывающей скобки –]. Так число 111111100 в упакованной записи будет записано следующим образом: 1[1[1[10]]]0[10]
Распакуйте строку 1[1010]10[10]01[10]10[10[1[11]]]10[101] в запись двоичного числа.

Распаковка числа

Рассмотрим способ упакованной записи чисел в троичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 111111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в троичной системе и закрывающей скобки –]. Так число 11111111000 в упакованной записи будет записано следующим образом: 1[2[2]]0[10]
Распакуйте строку 1[1[10]]0[2]1[10]2[2[1]]10[11] в запись троичного числа.

Последовательность

Последовательность чисел 2303, 511, 3311, 1519, 2527, 735, 1743, 2751, 959, 1967, ... , 1792 была получена следующим образом. Сначала были выписаны 500 первых натуральных чисел, кратных 7, по возрастанию (7, 14, … 3500). Затем все числа были переведены в шестнадцатеричную систему, причём каждое число было записано тремя цифрами (007, 00E, …, DAC). Затем каждая запись числа была преобразована в строку и символы в этой строке были записаны в обратном порядке (700, E00, …, CAD). После чего полученные строки были отсортированы в порядке, обратном алфавитному (FF8, FF1, …, 007). Затем строки были обратно преобразованы в исходные числа (2303, 511, …, 1792). Определите номер места, на котором стоит в последовательности число 3024 (места нумеруются с 1).
Указание: Для решения можно использовать электронные таблицы.
При сдаче задачи в первой строке запишите ответ, в следующих строках приведите описание процесса получения ответа (это может быть - описание действий, программа и т.д.)

Последовательность

Последовательность чисел 4095, 1791, 2799, 495, 3807, 1503, 2511, 207, 3519, 1215, ... , 2304 была получена следующим образом. Сначала были выписаны 455 первых натуральных чисел, кратных 9, по возрастанию (9, 18, … 4095). Затем все числа были переведены в шестнадцатеричную систему, причём каждое число было записано тремя цифрами (009, 012, …, FFF). Затем каждая запись числа была преобразована в строку и символы в этой строке были записаны в обратном порядке (900, 210, …, FFF). После чего полученные строки были отсортированы в порядке, обратном алфавитному (FFF, FF6, …, 009). Затем строки были обратно преобразованы в исходные числа (4095, 1791, …, 2304). Определите число, которое стоит в последовательности на 239-м месте (нумеруются с 1).
Указание: Для решения можно использовать электронные таблицы.
При сдаче задачи в первой строке запишите ответ, в следующих строках приведите описание процесса получения ответа (это может быть - описание действий, программа и т.д.)

Последовательность

Последовательность чисел 2559, 1791, 1023, 255, 2799, 2031, 1263, 495, 2271, 1503, ... , 768 была получена следующим образом. Сначала были выписаны 1000 первых натуральных чисел, кратных 3, по возрастанию (3, 6, … 3000). Затем все числа были переведены в шестнадцатеричную систему, причём каждое число было записано тремя цифрами (003, 006, …, BB8). Затем каждая запись числа была преобразована в строку и символы в этой строке были записаны в обратном порядке (300, 600, …, 8BB). После чего полученные строки были отсортированы в порядке, обратном алфавитному (FF9, FF6, …, 003). Затем строки были обратно преобразованы в исходные числа (2559, 1791, …, 768). Определите номер места, на котором стоит в последовательности число 1971 (места нумеруются с 1).

Указание: Для решения можно использовать электронные таблицы.
При сдаче задачи в первой строке запишите ответ, в следующих строках приведите описание процесса получения ответа (это может быть - описание действий, программа и т.д.)

Последовательность

Последовательность чисел 4095, 2815, 1535, 255, 3055, 1775, 495, 3295, 2015, 735, ... , 1280 была получена следующим образом. Сначала были выписаны 819 первых натуральных чисел, кратных 5, по возрастанию (5, 10, … 4095). Затем все числа были переведены в шестнадцатеричную систему, причём каждое число было записано тремя цифрами (005, 00A, …, FFF). Затем каждая запись числа была преобразована в строку и символы в этой строке были записаны в обратном порядке (500, A00, …, FFF). После чего полученные строки были отсортированы в порядке, обратном алфавитному (FFF, FFA, …, 005). Затем строки были обратно преобразованы в исходные числа (4095, 2815, …, 1280). Определите число, которое стоит в последовательности на 745-м месте (нумеруются с 1).
Указание: Для решения можно использовать электронные таблицы.
При сдаче задачи в первой строке запишите ответ, в следующих строках приведите описание процесса получения ответа (это может быть - описание действий, программа и т.д.)

Формула

Написать программу, которая по заданному целому положительному числу до 10000 получает минимальную по длине формулу дающую это число. Формула может состоять из символов '2', '3', '+', '*'.
Если получить число невозможно, выведите "No solution" без кавычек

Examples

Input

5

Output

3+2

Input

11

Output

3*3+2

Распаковка числа - 2

Рассмотрим способ упакованной записи чисел в p-чной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 111111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в p-чной системе и закрывающей скобки –]. Так число 111111111000 в упакованной записи в двоичной системе может быть записано следующим образом: 1[10[10]1]0[11]
Составьте программу, которая распаковывает число в p-чной системе счисления.

Input format

В первой строке вводится основание системы счисления p (1 < p < 11). Во второй строке находится упакованное число.

Output format

Выведите распакованное число. Гарантируется, что длина распакованного числа не превосходит 100000 символов

Examples

Input

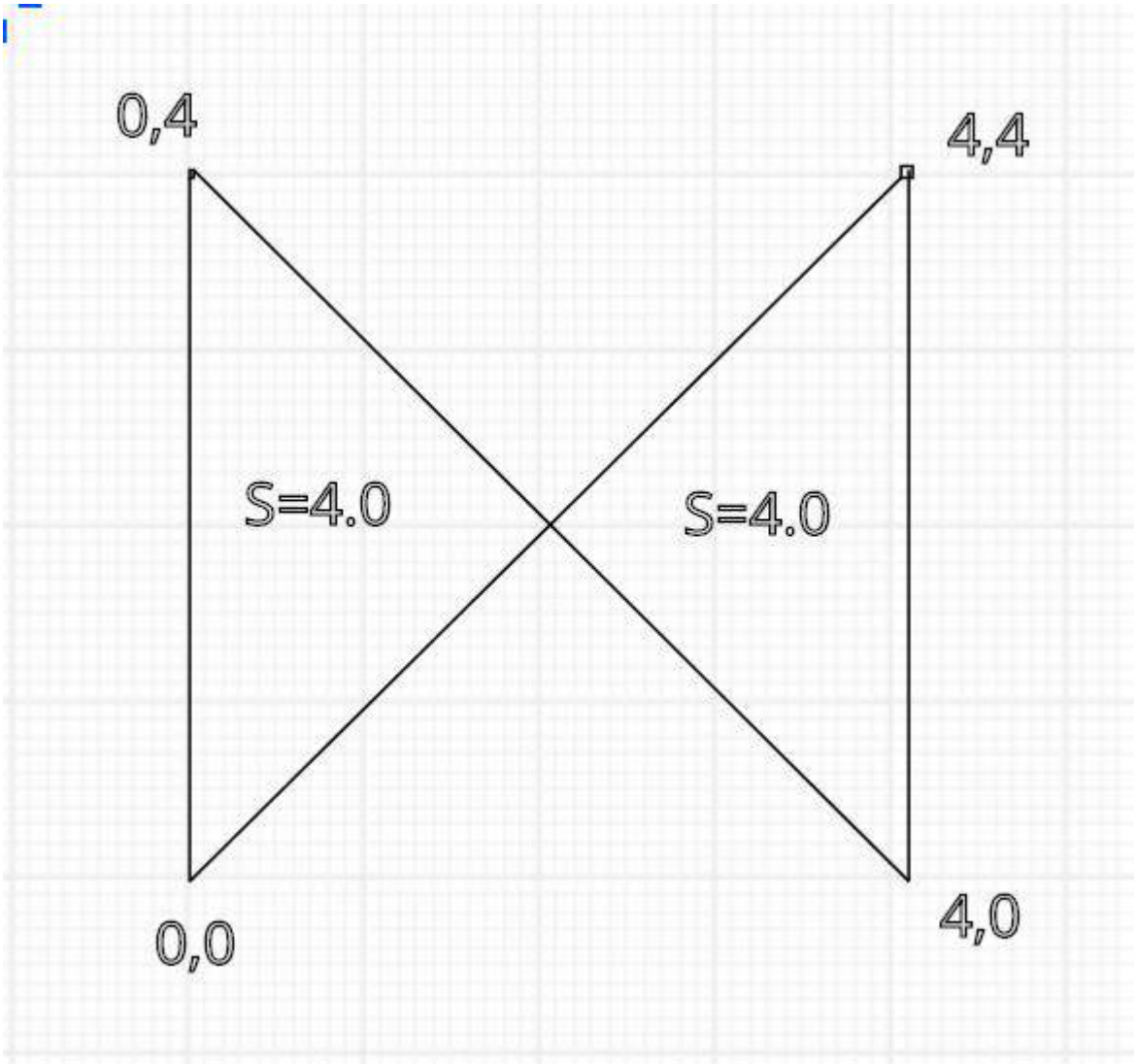
2
1[1010]000

Output

1111111111000

Площади

Задана замкнутая ломаная, напишите программу, которая находит конечные площади каждой области, которые эта ломаная образует.



Input format

В первой строке вводится N - число точек замкнутой ломаной (2 < N < 11). В следующих N строках целочисленные координаты точек ломаной на плоскости.

Output format

Выведите ненулевые площади конечных областей в порядке возрастания с точностью 1e-6

Examples

Input

```
4
0 0
0 4
4 0
4 4
```

Output

4.000000
4.000000

Problem 6: Пересечение отрезков

Дана задача. На стандартном потоке ввода подаются четыре 32-битных знаковых целых чисел L1, H1, L2, H2. Числа L1, H1 задают отрезок на числовой прямой вида [L1;H1) (то есть L1 входит в отрезок, а H1 — не входит). Гарантируется, что L1 <= H1, если L1 == H1, такой отрезок задает пустое множество. Числа L2, H2 задают границы второго отрезка на числовой прямой аналогично первому отрезку.
На стандартный поток вывода напечатайте пару 32-битных знаковых целых чисел L3, H3, задающих отрезок на числовой прямой, который является пересечением отрезков [L1;H1) и [L2;H2). Если результатом пересечения является пустое множество, оно всегда выводится в виде "0 0".
Ваша задача — написать тесты для проверки решений этой задачи. Множество тестов должно быть корректным и полным. Корректность означает, что любое правильное решение исходной задачи должно успешно проходить все тесты. Полнота означает, что любое разумно неправильное решение исходной задачи должно не проходить хотя бы один тест. Под разумно неправильным решением понимается решение, которое не выдает намеренно неправильный ответ в зависимости от случайного входного набора или для входных данных, не следующих из условия задачи. Например, "неразумное" неправильное решение может давать неправильный ответ только если значение H2 равно 1231412421. Такие "неразумные" неправильные решения рассматривать не нужно.
Тесты состоят из набора тестовых пар <входные данные;правильный ответ>. Входные данные должны быть корректными и удовлетворять ограничениям задачи. В данном случае, входные данные должны представлять собой четыре целых числа, представимых 32-битным знаковым типом, с указанными ограничениями на границы. Числа могут разделяться пробельными символами произвольным образом.
Правильный ответ должен быть корректен. В данном случае, правильный ответ должен представлять собой два числа, являющихся правильным ответом на входные данные. Числа могут разделяться пробельными символами произвольным образом.
Файлы с входными данными должны называться 001.dat, 002.dat, 003.dat и т. д. Соответствующие файлы с правильным ответом должны называться 001.ans, 002.ans, 003.ans и т. д. Размер любого файла не должен превышать 1KiB.
Тестовый набор необходимо сдать на проверку в виде архива в формате .tgz. Архив должен содержать единственный каталог с именем tests, в котором должны размещаться разработанные вами тесты. Количество тестов не должно превышаться 20.
Для создания архива формата .tgz в командной строке выполните команду:
tar cfz arch.tgz tests
Или воспользуйтесь архиватором 7zip

Папайя

Откапывая исчезнувший город древней цивилизации Папайя, археологи обнаружили довольно странный механизм, который при ближайшем рассмотрении оказался вычислительной машиной. Исходные данные машина читает с медленно задвигаемого во входное отверстие бамбукового стебля, на котором могут быть записаны числа в виде десятичных дробей и латинские буквы (пробелы могут быть использованы для разделения стоящих рядом чисел, в остальных случаях пробелы игнорируются). Чтобы показать, что ввод данных закончен, стебель в нужном месте отпиливают.

Результаты вычислений машина вырезает автоматическим резцом на другом бамбуковом стебле, причём числа с ненулевой дробной частью она почему-то всегда отображает в виде десятичных дробей с шестью знаками после запятой; если же число целое, то дробная часть для него не отображается. Центральный процессор машины оснащён двадцатью шестью регистрами, которые обозначаются заглавными латинскими буквами от A до Z. К каждому регистру прилагается ещё некое подобие оперативного запоминающего устройства, организованного в виде стека, причём ячеек в каждом из 26 стеков оказалось довольно много - во всяком случае, исследователи быстро сбились со счёта. Каждый регистр, а также каждая ячейка каждого стека могут в любой момент времени содержать в качестве значения число (целое или дробное), символ (при этом в алфавит машины входят заглавные и строчные латинские буквы, точка, запятая, вопросительный и восклицательный знаки, а также пробел), либо специальное "пустое" значение, обозначаемое []. Значение "пусто" также используется машиной для обозначения логической лжи, любые другие значения считаются истинными.

Поскольку регистр A участвует во всех вычислениях, археологи решили называть его аккумулятором; кроме того, любой из 26 регистров, в том числе аккумулятор, может быть назначен текущим, причём в начале работы текущим считается как раз аккумулятор. При выполнении любой арифметической операции машина берёт первый операнд из аккумулятора, второй операнд - из текущего регистра, результат помещает обратно в аккумулятор.

Программа для машины Папайя представляет собой строку символов, каждый из которых задаёт машинную команду; программа, состоящая из символов-команд, выполняется последовательно справа налево, кроме двух команд, которые могут нарушить эту последовательность. Команды a, b, c и все остальные строчныелатинские буквы означают "занести данную букву в аккумулятор", а команда :(двоеточие) означает "изменить регистр буквы в аккумуляторе на противоположный", при этом заглавная буква меняется на строчную, строчная на заглавную, точка меняется на восклицательный знак и обратно, запятая меняется на вопросительный знак и обратно, если же в аккумуляторе что-то другое, то ничего не происходит. Команды A, B, C и все остальные заглавные латинские буквы означают "назначить данный регистр текущим". Команды 0, 1, ..., 9 означают "занести в аккумулятор соответственно число 0.0, 1.0, ... 9.0". Команда @ означает "занести в аккумулятор пробел". Команды +, -, *, / означают соответствующие арифметические действия, ^ означает возведение в степень, команды <, >, = означают сравнение двух чисел или двух символов, & и | означают логическое "и" и логическое "или"; все эти команды

используют значение из аккумулятора в качестве левого операнда, значение из текущего регистра в качестве правого операнда, результат заносится обратно в аккумулятор. Команда # умножает аккумулятор на десять, команда _ делит аккумулятор на десять. Команда !. работает как логическое отрицание аккумулятора: если там значение "пусто", то заносится значение 1, если любое другое - заносится значение "пусто".

Команда $ выдаёт текущее значение аккумулятора на печать, при этом буквы, знаки препинания и пробелы печатаются как они есть, числа с нулевой дробной частью печатаются как целые, остальные числа печатаются с пятью знаками после запятой; команда ? вводит очередное число или букву в аккумулятор, а если входной бабмуковый стебель кончился, заносит в аккумулятор значение "пусто". Следует обратить внимание, что на входном стебле любая последовательность цифр, возможно, разделённая одной точкой, и, возможно, с минусом в начале, воспринимается как единое значение (число) точка воспринимается как обычный символ только в случае, если она идёт не после цифры, либо если в данной последовательности цифр это уже вторая точка.

Команда] заносит значение из текущего регистра в его стек (напомним, что стек у каждого регистра свой); команда [извлекает значение с вершины стека текущего регистра и заносит его в текущий регистр (если извлекать нечего, в регистр заносится значение "пусто"); команда ~ меняет местами значения в текущем регистре и на вершине его стека, а если стек пуст, заносит в регистр значение "пусто", а в стек - бывшее содержимое регистра.

Команда) копирует значение из аккумулятора в текущий регистр, команда (копирует значение из текущего регистра в аккумулятор.

Команды { и } предназначены для организации ветвлений и циклов и всегда должны в программе стоять парами, то есть в программе должен обязательно соблюдаться баланс фигурных скобок. Выполняются они так. Команда {, если в аккумуляторе "пусто", идёт по программе вперёд, пока не найдёт парную скобку, и после этого выполнение продолжится со следующей за этой закрывающей скобкой позиции; если в аккумуляторе было что-то другое, команда вообще ничего не делает, то есть выполнение продолжается прямо с команды, следующей за ней. Команда }, наоборот, если в аккумуляторе "пусто", не делает ничего, тогда как если там что-то другое, просматривает программу назад до парной фигурной скобки, после чего продолжает выполнение с команды, стоящей после такой скобки справа.

Наконец, команда " прекращает выполнение программы, при этом выполнение считается успешным. Если программа кончилась, не встретив эту команду, она завершается аварийно. Пробелы в программе игнорируются
Список команд

- a-z - занести букву; 0-9 - занести число; @ -- занести пробел
- : - сменить регистр символа; A-Z - сменить текущий регистр
- +, -, *, /, - арифм. операция; ^ - возведение в степень
- <, >, = - операция сравнения; &, | - лог.операция

- # - умножение на 10; _ - деление на 10
- ! - отрицание аккумулятора
- ? - ввод; $ - вывод
- [- из стека текущего регистра;] - в стек текущего регистра
- ~ - поменять местами текущий регистр и вершину его стека
-) - из аккумулятора в текущий регистр; (- из текущего регистра в аккумулятор
- { - если ложь, то вперёд до парной скобки; } - если истина, то назад до парной скобки
- " - стоп (успех)

В вашем распоряжении оказался эмулятор данного компьютера) и требуется написать для него программу, которая по заданному на ленте году выводит строку "YES", если он високосный и "NO" в противном случае (без кавычек). Год является високосным в двух случаях: либо он кратен 4, но при этом не кратен 100, либо кратен 400. Год не является високосным, если он не кратен 4, либо он кратен 100, но при этом не кратен 400.

Examples

Input

2014

Output

NO

Input

2004

Output

YES

Площадь суши

В некотором плоском клетчатом прямоугольном мире (N на M клеток) существует 2 континента. Континент представляет из себя связанный набор клеток. Континенты не имеют общих точек. Черные клетки (обозначаются 1) – суша, белые клетки (обозначаются 0) – вода.
На одном из них живет красный квадрокоптер, который мечтает съездить в путешествие на другой континент. Квадрокоптер умеет двигаться вверх, вниз, вправо, влево (соответствующие команды U, D, R, L), хранить в памяти любой кусочек карты и видеть в радиусе одной клетки (в том числе по диагонали). После перемещения на стандартный поток ввода будет записано то, что видит квадрокоптер вокруг себя. У квадрокоптера зарядки хватает на 8 действий движения. После истощения зарядки он приземляется на клетку, над которой находится. Если это суша – он может сесть и

зарядиться от солнечной энергии. Если это вода – квадрокоптер тонет. Принудительная посадка квадрокоптера – P. Утверждается, что между 2 континентами существует путь, занимающий не более 5 действий квадрокоптера. Ваша задача определить площадь суши. Как только программа готова вывести ответ напечатайте W и одно число - площадь суши. В начале работы программы на стандартном потоке ввода находится информация о том, что видит квадрокоптер вокруг себя. После каждого вывода требуется произвести операцию очистки буфера вывода. Примеры кода для этого действия можно найти на http://ejudge.cs.msu.ru/kvadrocopter

Input format

В начале работы программы и после каждого перемещения ввядится 9 чисел 0 или 1 - квадрат, который видит квадрокоптер

Output format

Выводится одна из команд перемещения или W и число, когда ответ найден

Examples

Input

```
0 0 0
0 1 0
0 0 0

0 0 0
1 0 1
0 0 0

0 0 0
0 1 0
0 0 0
```

Output

```
R
R
W 2
```

Отборочный-1 5-9

Распаковка числа

Рассмотрим способ упакованной записи чисел в двоичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 1111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в двоичной

системе и закрывающей скобки –]. Так число 111111100 в упакованной записи будет записано следующим образом: 1[1[1[10]]]0[10]
Распакуйте строку 1[11]0[1]0[1[11]]10[10[11]1]1[1011]0 в запись двоичного числа.

Распаковка числа

Рассмотрим способ упакованной записи чисел в троичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 111111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в троичной системе и закрывающей скобки –]. Так число 11111111000 в упакованной записи будет записано следующим образом: 1[2[2]]0[10]
Распакуйте строку 1[11]2[2[2]]1[1]0[12]1[1[10]]0[1] в запись троичного числа.

Распаковка числа

Рассмотрим способ упакованной записи чисел в двоичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 1111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в двоичной системе и закрывающей скобки –]. Так число 111111100 в упакованной записи будет записано следующим образом: 1[1[1[10]]]0[10]
Распакуйте строку 1[1010]10[10]01[10]10[10[1[11]]]10[101] в запись двоичного числа.

Распаковка числа

Рассмотрим способ упакованной записи чисел в троичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 111111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в троичной системе и закрывающей скобки –]. Так число 11111111000 в упакованной записи будет записано следующим образом: 1[2[2]]0[10]
Распакуйте строку 1[1[10]]0[2]1[10]2[2[1]]10[11] в запись троичного числа.

Последовательность

Последовательность чисел 2303, 511, 3311, 1519, 2527, 735, 1743, 2751, 959, 1967, ... , 1792 была получена следующим образом. Сначала были выписаны 500 первых натуральных чисел, кратных 7, по возрастанию (7, 14, … 3500). Затем все числа были переведены в шестнадцатеричную систему, причём каждое число было записано тремя цифрами (007, 00E, …, DAC). Затем каждая запись числа была преобразована в строку и символы в этой строке были записаны в обратном порядке (700, E00, …, CAD). После чего полученные строки были отсортированы в порядке, обратном алфавитному (FF8, FF1, …, 007). Затем строки были обратно преобразованы в

исходные числа (2303, 511, …, 1792). Определите номер места, на котором стоит в последовательности число 3024 (места нумеруются с 1).
Указание: Для решения можно использовать электронные таблицы.
При сдаче задачи в первой строке запишите ответ, в следующих строках приведите описание процесса получения ответа (это может быть - описание действий, программа и т.д.)

Последовательность

Последовательность чисел 4095, 1791, 2799, 495, 3807, 1503, 2511, 207, 3519, 1215, ... , 2304 была получена следующим образом. Сначала были выписаны 455 первых натуральных чисел, кратных 9, по возрастанию (9, 18, … 4095). Затем все числа были переведены в шестнадцатеричную систему, причём каждое число было записано тремя цифрами (009, 012, …, FFF). Затем каждая запись числа была преобразована в строку и символы в этой строке были записаны в обратном порядке (900, 210, …, FFF). После чего полученные строки были отсортированы в порядке, обратном алфавитному (FFF, FF6, …, 009). Затем строки были обратно преобразованы в исходные числа (4095, 1791, …, 2304). Определите число, которое стоит в последовательности на 239-м месте (нумеруются с 1).
Указание: Для решения можно использовать электронные таблицы.
При сдаче задачи в первой строке запишите ответ, в следующих строках приведите описание процесса получения ответа (это может быть - описание действий, программа и т.д.)

Последовательность

Последовательность чисел 2559, 1791, 1023, 255, 2799, 2031, 1263, 495, 2271, 1503, ... , 768 была получена следующим образом. Сначала были выписаны 1000 первых натуральных чисел, кратных 3, по возрастанию (3, 6, … 3000). Затем все числа были переведены в шестнадцатеричную систему, причём каждое число было записано тремя цифрами (003, 006, …, BB8). Затем каждая запись числа была преобразована в строку и символы в этой строке были записаны в обратном порядке (300, 600, …, 8BB). После чего полученные строки были отсортированы в порядке, обратном алфавитному (FF9, FF6, …, 003). Затем строки были обратно преобразованы в исходные числа (2559, 1791, …, 768). Определите номер места, на котором стоит в последовательности число 1971 (места нумеруются с 1).
Указание: Для решения можно использовать электронные таблицы.
При сдаче задачи в первой строке запишите ответ, в следующих строках приведите описание процесса получения ответа (это может быть - описание действий, программа и т.д.)

Последовательность

Последовательность чисел 4095, 2815, 1535, 255, 3055, 1775, 495, 3295, 2015, 735, ... , 1280 была получена следующим образом. Сначала были выписаны 819 первых натуральных чисел, кратных 5, по возрастанию (5, 10, … 4095). Затем все числа были переведены в шестнадцатеричную систему, причём каждое число было записано тремя цифрами (005, 00A, …, FFF). Затем каждая запись числа была преобразована в строку и символы в этой строке были записаны в обратном порядке (500, A00, …, FFF). После чего полученные строки были отсортированы в порядке, обратном алфавитному (FFF, FFA, …, 005). Затем строки были обратно преобразованы в исходные числа (4095, 2815, …, 1280). Определите число, которое стоит в последовательности на 745-м месте (нумеруются с 1).
Указание: Для решения можно использовать электронные таблицы.
При сдаче задачи в первой строке запишите ответ, в следующих строках приведите описание процесса получения ответа (это может быть - описание действий, программа и т.д.)

Формула

Написать программу, которая по заданному целому положительному числу до 10000 получает минимальную по длине формулу дающую это число. Формула может состоять из символов '2', '3', '+', '*'.
Если получить число невозможно, выведите "No solution" без кавычек

Examples

Input

5

Output

3+2

Input

11

Output

3*3+2

Распаковка числа - 2

Рассмотрим способ упакованной записи чисел в p-чной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 111111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в p-чной системе и закрывающей скобки –]. Так число 111111111000 в упакованной записи в двоичной системе может быть записано следующим образом: 1[10[10]1]0[11]
Составьте программу, которая распаковывает число в p-чной системе счисления.

Input format

В первой строке вводится основание системы счисления p (1 < p < 11). Во второй строке находится упакованное число.

Output format

Выведите распакованное число. Гарантируется, что длина распакованного числа не превосходит 100000 символов

Examples

Input

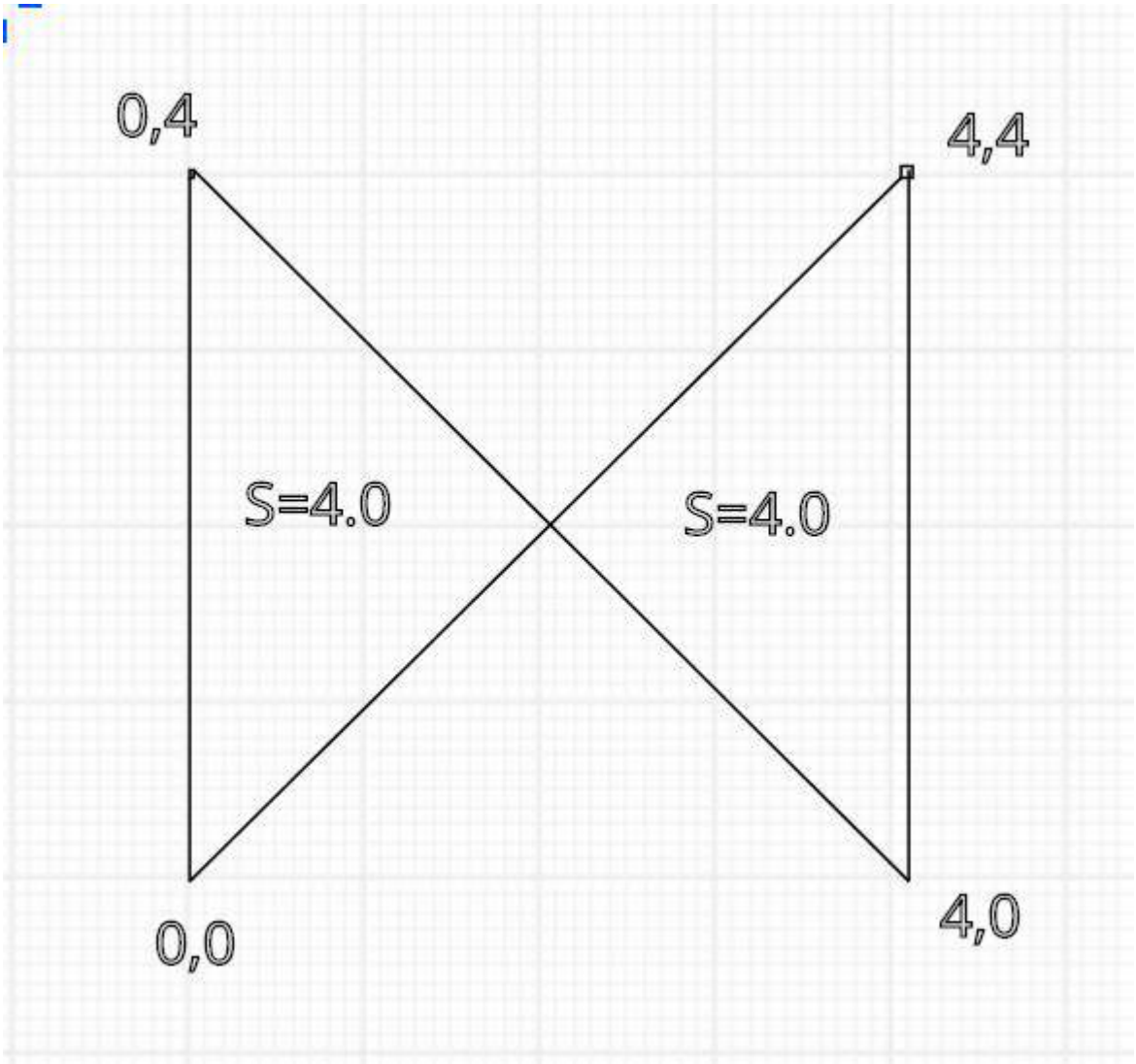
2
1[1010]000

Output

1111111111000

Площади

Задана замкнутая ломаная, напишите программу, которая находит конечные площади каждой области, которые эта ломаная образует.



Input format

В первой строке вводится N - число точек замкнутой ломаной (2 < N < 11). В следующих N строках целочисленные координаты точек ломаной на плоскости.

Output format

Выведите ненулевые площади конечных областей в порядке возрастания с точностью 1e-6

Examples

Input

```
4
0 0
0 4
4 0
4 4
```

Output

4.000000
4.000000

Problem 6: Пересечение отрезков

Дана задача. На стандартном потоке ввода подаются четыре 32-битных знаковых целых чисел L1, H1, L2, H2. Числа L1, H1 задают отрезок на числовой прямой вида [L1;H1) (то есть L1 входит в отрезок, а H1 — не входит). Гарантируется, что L1 <= H1, если L1 == H1, такой отрезок задает пустое множество. Числа L2, H2 задают границы второго отрезка на числовой прямой аналогично первому отрезку.
На стандартный поток вывода напечатайте пару 32-битных знаковых целых чисел L3, H3, задающих отрезок на числовой прямой, который является пересечением отрезков [L1;H1) и [L2;H2). Если результатом пересечения является пустое множество, оно всегда выводится в виде "0 0".
Ваша задача — написать тесты для проверки решений этой задачи. Множество тестов должно быть корректным и полным. Корректность означает, что любое правильное решение исходной задачи должно успешно проходить все тесты. Полнота означает, что любое разумно неправильное решение исходной задачи должно не проходить хотя бы один тест. Под разумно неправильным решением понимается решение, которое не выдает намеренно неправильный ответ в зависимости от случайного входного набора или для входных данных, не следующих из условия задачи. Например, "неразумное" неправильное решение может давать неправильный ответ только если значение H2 равно 1231412421. Такие "неразумные" неправильные решения рассматривать не нужно.
Тесты состоят из набора тестовых пар <входные данные;правильный ответ>. Входные данные должны быть корректными и удовлетворять ограничениям задачи. В данном случае, входные данные должны представлять собой четыре целых числа, представимых 32-битным знаковым типом, с указанными ограничениями на границы. Числа могут разделяться пробельными символами произвольным образом.
Правильный ответ должен быть корректен. В данном случае, правильный ответ должен представлять собой два числа, являющихся правильным ответом на входные данные. Числа могут разделяться пробельными символами произвольным образом.
Файлы с входными данными должны называться 001.dat, 002.dat, 003.dat и т. д. Соответствующие файлы с правильным ответом должны называться 001.ans, 002.ans, 003.ans и т. д. Размер любого файла не должен превышать 1KiB.
Тестовый набор необходимо сдать на проверку в виде архива в формате .tgz. Архив должен содержать единственный каталог с именем tests, в котором должны размещаться разработанные вами тесты. Количество тестов не должно превышаться 20.
Для создания архива формата .tgz в командной строке выполните команду:
tar cfz arch.tgz tests
Или воспользуйтесь архиватором 7zip

Папайя

Откапывая исчезнувший город древней цивилизации Папайя, археологи обнаружили довольно странный механизм, который при ближайшем рассмотрении оказался вычислительной машиной. Исходные данные машина читает с медленно задвигаемого во входное отверстие бамбукового стебля, на котором могут быть записаны числа в виде десятичных дробей и латинские буквы (пробелы могут быть использованы для разделения стоящих рядом чисел, в остальных случаях пробелы игнорируются). Чтобы показать, что ввод данных закончен, стебель в нужном месте отпиливают.

Результаты вычислений машина вырезает автоматическим резцом на другом бамбуковом стебле, причём числа с ненулевой дробной частью она почему-то всегда отображает в виде десятичных дробей с шестью знаками после запятой; если же число целое, то дробная часть для него не отображается. Центральный процессор машины оснащён двадцатью шестью регистрами, которые обозначаются заглавными латинскими буквами от A до Z. К каждому регистру прилагается ещё некое подобие оперативного запоминающего устройства, организованного в виде стека, причём ячеек в каждом из 26 стеков оказалось довольно много - во всяком случае, исследователи быстро сбились со счёта. Каждый регистр, а также каждая ячейка каждого стека могут в любой момент времени содержать в качестве значения число (целое или дробное), символ (при этом в алфавит машины входят заглавные и строчные латинские буквы, точка, запятая, вопросительный и восклицательный знаки, а также пробел), либо специальное "пустое" значение, обозначаемое []. Значение "пусто" также используется машиной для обозначения логической лжи, любые другие значения считаются истинными.

Поскольку регистр A участвует во всех вычислениях, археологи решили называть его аккумулятором; кроме того, любой из 26 регистров, в том числе аккумулятор, может быть назначен текущим, причём в начале работы текущим считается как раз аккумулятор. При выполнении любой арифметической операции машина берёт первый операнд из аккумулятора, второй операнд - из текущего регистра, результат помещает обратно в аккумулятор.

Программа для машины Папайя представляет собой строку символов, каждый из которых задаёт машинную команду; программа, состоящая из символов-команд, выполняется последовательно справа налево, кроме двух команд, которые могут нарушить эту последовательность. Команды a, b, c и все остальные строчныелатинские буквы означают "занести данную букву в аккумулятор", а команда :(двоеточие) означает "изменить регистр буквы в аккумуляторе на противоположный", при этом заглавная буква меняется на строчную, строчная на заглавную, точка меняется на восклицательный знак и обратно, запятая меняется на вопросительный знак и обратно, если же в аккумуляторе что-то другое, то ничего не происходит. Команды A, B, C и все остальные заглавные латинские буквы означают "назначить данный регистр текущим". Команды 0, 1, ..., 9 означают "занести в аккумулятор соответственно число 0.0, 1.0, ... 9.0". Команда @ означает "занести в аккумулятор пробел". Команды +, -, *, / означают соответствующие арифметические действия, ^ означает возведение в степень, команды <, >, = означают сравнение двух чисел или двух символов, & и | означают логическое "и" и логическое "или"; все эти команды

используют значение из аккумулятора в качестве левого операнда, значение из текущего регистра в качестве правого операнда, результат заносится обратно в аккумулятор. Команда # умножает аккумулятор на десять, команда _ делит аккумулятор на десять. Команда !. работает как логическое отрицание аккумулятора: если там значение "пусто", то заносится значение 1, если любое другое - заносится значение "пусто".

Команда $ выдаёт текущее значение аккумулятора на печать, при этом буквы, знаки препинания и пробелы печатаются как они есть, числа с нулевой дробной частью печатаются как целые, остальные числа печатаются с пятью знаками после запятой; команда ? вводит очередное число или букву в аккумулятор, а если входной бабмуковый стебель кончился, заносит в аккумулятор значение "пусто". Следует обратить внимание, что на входном стебле любая последовательность цифр, возможно, разделённая одной точкой, и, возможно, с минусом в начале, воспринимается как единое значение (число) точка воспринимается как обычный символ только в случае, если она идёт не после цифры, либо если в данной последовательности цифр это уже вторая точка.

Команда] заносит значение из текущего регистра в его стек (напомним, что стек у каждого регистра свой); команда [извлекает значение с вершины стека текущего регистра и заносит его в текущий регистр (если извлекать нечего, в регистр заносится значение "пусто"); команда ~ меняет местами значения в текущем регистре и на вершине его стека, а если стек пуст, заносит в регистр значение "пусто", а в стек - бывшее содержимое регистра.

Команда) копирует значение из аккумулятора в текущий регистр, команда (копирует значение из текущего регистра в аккумулятор.

Команды { и } предназначены для организации ветвлений и циклов и всегда должны в программе стоять парами, то есть в программе должен обязательно соблюдаться баланс фигурных скобок. Выполняются они так. Команда {, если в аккумуляторе "пусто", идёт по программе вперёд, пока не найдёт парную скобку, и после этого выполнение продолжится со следующей за этой закрывающей скобкой позиции; если в аккумуляторе было что-то другое, команда вообще ничего не делает, то есть выполнение продолжается прямо с команды, следующей за ней. Команда }, наоборот, если в аккумуляторе "пусто", не делает ничего, тогда как если там что-то другое, просматривает программу назад до парной фигурной скобки, после чего продолжает выполнение с команды, стоящей после такой скобки справа.

Наконец, команда " прекращает выполнение программы, при этом выполнение считается успешным. Если программа кончилась, не встретив эту команду, она завершается аварийно. Пробелы в программе игнорируются
Список команд

- a-z - занести букву; 0-9 - занести число; @ -- занести пробел
- : - сменить регистр символа; A-Z - сменить текущий регистр
- +, -, *, /, - арифм. операция; ^ - возведение в степень
- <, >, = - операция сравнения; &, | - лог.операция

- # - умножение на 10; _ - деление на 10
- ! - отрицание аккумулятора
- ? - ввод; $ - вывод
- [- из стека текущего регистра;] - в стек текущего регистра
- ~ - поменять местами текущий регистр и вершину его стека
-) - из аккумулятора в текущий регистр; (- из текущего регистра в аккумулятор
- { - если ложь, то вперёд до парной скобки; } - если истина, то назад до парной скобки
- " - стоп (успех)

В вашем распоряжении оказался эмулятор данного компьютера) и требуется написать для него программу, которая по заданному на ленте году выводит строку "YES", если он високосный и "NO" в противном случае (без кавычек). Год является високосным в двух случаях: либо он кратен 4, но при этом не кратен 100, либо кратен 400. Год не является високосным, если он не кратен 4, либо он кратен 100, но при этом не кратен 400.

Examples

Input

2014

Output

NO

Input

2004

Output

YES

Площадь суши

В некотором плоском клетчатом прямоугольном мире (N на M клеток) существует 2 континента. Континент представляет из себя связанный набор клеток. Континенты не имеют общих точек. Черные клетки (обозначаются 1) – суша, белые клетки (обозначаются 0) – вода.
На одном из них живет красный квадрокоптер, который мечтает съездить в путешествие на другой континент. Квадрокоптер умеет двигаться вверх, вниз, вправо, влево (соответствующие команды U, D, R, L), хранить в памяти любой кусочек карты и видеть в радиусе одной клетки (в том числе по диагонали). После перемещения на стандартный поток ввода будет записано то, что видит квадрокоптер вокруг себя. У квадрокоптера зарядки хватает на 8 действий движения. После истощения зарядки он приземляется на клетку, над которой находится. Если это суша – он может сесть и

зарядиться от солнечной энергии. Если это вода – квадрокоптер тонет. Принудительная посадка квадрокоптера – P. Утверждается, что между 2 континентами существует путь, занимающий не более 5 действий квадрокоптера. Ваша задача определить площадь суши. Как только программа готова вывести ответ напечатайте W и одно число - площадь суши. В начале работы программы на стандартном потоке ввода находится информация о том, что видит квадрокоптер вокруг себя. После каждого вывода требуется произвести операцию очистки буфера вывода. Примеры кода для этого действия можно найти на http://ejudge.cs.msu.ru/kvadrocopter

Input format

В начале работы программы и после каждого перемещения ввядится 9 чисел 0 или 1 - квадрат, который видит квадрокоптер

Output format

Выводится одна из команд перемещения или W и число, когда ответ найден

Examples

Input

```
0 0 0
0 1 0
0 0 0

0 0 0
1 0 1
0 0 0

0 0 0
0 1 0
0 0 0
```

Output

```
R
R
W 2
```