# Алгоритм Маркова

Нормальный алгоритм Маркова - вещь довольно известная в теории алгоритмов. Почитать можно много где, не рассказывать же, как делается поиск в Интернете... Изложим несколько упрощенную схему. Имеем некоторый набор символов (алфавит), и строку из этих символов. Кроме того, имеем правила замены, в каждом из которых указано, какую подстроку с исходной строке нужно заменить, и на что ее менять. Подстроки могут быть и пустыми. Порядок правил фиксирован. Функционирует все следующим образом. Правила просматриваются в указанном порядке на предмет применимости. Первое же правило, которое может быть применено, однократно применяется (то есть выполняется описанная в нем замена), после чего цикл обработки повторяется (список просматривается заново). "Однократно" означает, что если в строке к моменту выполнения правила существует несколько подстрок, которые могут быть заменены, то заменяется только одна подстрока из возможных, а именно - самая левая. Процесс заканчивается, если после очередного просмотра строка не изменилась. В процессе выполнения замен длина строки может изменяться, увеличиваться, уменьшаться – всё пожалуйста.

**Пример**:
Есть строка *ababaab*
И набор правил
*ab->c*
*cc->ab*
В результате применения описанной процедуры строка будет преобразована в cac. При этом будут получаться такие промежуточные результаты: *ababaab, cabaab, ccaab, ccac, abac, cac*.

Ну вот. А теперь несколько задачек.

**Задача 1**: Начальная непустая строка была получена из обычного арифметического выражения путем удаления всех символов, кроме открывающейся и закрывающейся круглой скобки. Требуется написать последовательность команд, которая приводит эту строку к строке "RIGHT" либо "WRONG" в зависимости от того, верно или неверно были расставлены скобки в исходной строке в соответствии с обычными правилами записи арифметических выражений. Например, строку ()((())()) предлагаемая последовательность замен должна перевести в RIGHT, а строку (() та же самая последовательность замен должна перевести в строку WRONG.

**Задача 2**: Начальная строка представляет из себя произвольную строку типа [число1]+[число2]=? где [число1] и [число2] - десятичные записи некоторых натуральных чисел. Требуется написать последовательность замен, которая переведет эту запись в завись вида [число1]+[число2]=[сумма], где [сумма] - десятичная запись результата сложения числа 1 и числа 2. Ну, то есть строка 2+2=? должна быть переведена в 2+2=4, а строка 25+76=? та же самая последовательность замен должна перевести в 25+76=101. Длина исходной строки любая (то есть в качестве слагаемых могут быть представлены большие натуральные числа (в данной задаче до 100 знаков)).

**Задача 3**: Задана строка из больших букв английского алфавита (символы от A до Z), заканчивающаяся знаком вопроса "?". Необходимо вывести их в обратном порядке, уже без знака вопроса. То есть строка ABBCD? должна стать строкой DCBBA

**Задача 4**: Задано двоичное число, состоящее из 0 и 1. Необходимо вывести его в виде набора букв z. Где количество таких букв равно заданному двоичному числу. То есть для 110 алгоритм должен вывести zzzzzz

**Задача 5**: Задана строка из больших букв английского алфавита (символы от A до Z)), заканчивающаяся знаком вопроса "?". Надо написать набор правил, который отсортирует её по возрастанию и выведет без знака вопроса. В отсортированном виде DFAAS? будет выглядеть как AADFS.

**Задача 6**: Задано два десятичных числа через символ подчеркивания "_" и затем строка "=?", например 30_42=?. Требуется вывести вместо них значение наибольшего общего делителя для этой пары. Для 30_42=? это будет 6.

Замечание: Ограничение на увеличение строки составляет 100000 символов.

## Входные данные

В данной задаче нет входных данных.

## Выходные данные

Решения для задач необходимо выводить последовательно. Если вы не хотите решать задачу, то выведите 0, иначе выведите длину вашего решения N. Затем выведите ровно N строк вида abc->def, где abc подстрока для замены, а def подстрока которой мы заменяем.

## Начисление очков

За каждую корректно решенную задачу вы получите 1 балл и дополнительно 1/N бонусных баллов, где N длина предложенного вами решения.

## Пример

**Выходные данные:**
0
0
0
5
a->b
b->c
d->v
g->l
l->a

**Начисление очков:**
В данном случае если цепочка приведет к правильному результату
вы получите 1 + 0.2 = 1.2 балла за 4 задачу.