# A. Сбор запросотары

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 512Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Гуляя по волшебному марсианскому лесу, Петя наткнулся на большую поляну [URLов](https://ru.wikipedia.org/wiki/URL) из HTTP-запросов к некоторым сайтам. Как-то раз в городе он проходил мимо необычного места с вывеской «Приёмный пункт запросов». Петя быстро смекнул, что он может собрать URL’ы с поляны и сдать их на переработку: так он поможет планете и немного заработает с этого. Радостный Петя принялся преобразовывать URL’ы в формат, который принимают в пункте приёма, однако спустя 4 минуты он понял, что не сможет обработать все найденные URL’ы самостоятельно. Помогите Пете: напишите программу, разбирающую найденный URL некоторого HTTP-запроса на части (см. примеры).

URL запроса HTTP состоит из следующих частей: протокол (http, https), хост (например, tyndex.com), путь запроса и список параметров (разделены символом &) с их значениями (пара ключ-значение вида key=value). Например, в URL https://tyndex.com/api/search?text=funny+cats&page=1: протокол – https, хост – tyndex.com, путь запроса – api/search, параметры – text -> funny+cats и page -> 1.

В задаче считается, что порт запроса – часть хоста (см. тест 2).

## Формат ввода

Единственная строка содержит корректный URL формата HTTP-запроса. Длина строки не превышает 105 символов.

Формат подаваемого на вход URL: %protocol%://%hostname%/%handle%?%param\_1%=%value\_1%...%param\_n%=%value\_n%, где %proto% – протокол, %hostname% – имя хоста, %handle% – путь запроса.

Гарантируется, что протокол имеет значение http или https, имя хоста может состоять только из букв латинского алфавита, цифр, точек и не более чем из одного двоеточия. Путь запроса может состоять только из букв латинского алфавита и символа слеша (/), а аргументы запроса – из букв латиницы, цифр и символов \_ и +. Гарантируется, что все аргументы различны.

Гарантируется, что входная строка не содержит пробельных символов и содержит корректный URL.

Обратите внимание, что handle и параметры запроса необязательны для URL, когда протокол и хост – обязательны. Также часть URL до параметров запроса (протокол, имя хоста и путь запроса) может заканчиваться символом /. Например, URL https://example.com/ и https://example.com/some/handle/?some=params также будут корректными.

## Формат вывода

Первые две строки вывода должны соблюдать формат:

Proto: %protocol%

Host: %hostname%

Где %proto% – протокол, %hostname% – имя хоста.

Далее, если в запросе был путь запроса и он не пустой, выведите строку Handle: %handle%, где %handle% – путь.

Имя хоста и путь запроса в выходном файле **не должны**начинаться или заканчиваться на символ / (см. примеры).

Далее, для каждого аргумента выведите его на отдельной строке в виде аргумент -> значение. Если у запроса не было аргументов, ничего выводить не нужно. Порядок аргументов не имеет значения.

### Пример 1

| **Ввод**  IMG_256 | **Вывод**  IMG_257 |
| --- | --- |
| https://tyndex.com/api/search?text=funny+cats&page=1 | Proto: https  Host: tyndex.com  Handle: api/search  page -> 1  text -> funny+cats |

### Пример 2

| **Ввод**  IMG_258 | **Вывод**  IMG_259 |
| --- | --- |
| http://tyndex.com:5000/api/search/?text=funny+cats | Proto: http  Host: tyndex.com:5000  Handle: api/search  text -> funny+cats |

### Пример 3

| **Ввод**  IMG_260 | **Вывод**  IMG_261 |
| --- | --- |
| https://www.youtube.com/watch?v=dQw4w9WgXcQ | Proto: https  Host: www.youtube.com  Handle: watch  v -> dQw4w9WgXcQ |

### Пример 4

| **Ввод**  IMG_262 | **Вывод**  IMG_263 |
| --- | --- |
| http://example.com | Proto: http  Host: example.com |

# B. Подбор пар

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

В преддверии крупного соревнования по шахматам, тренер Василий хочет обучить n учеников. Для этого он собирается провести партии для каждой пары учеников. Однако Василий понимает, что у него осталось мало времени до соревнования, поэтому он хочет оптимизировать такой процесс.

Для начала Василий оценил умение каждого ученика как si. Василий понял, что у него в группе учатся ребята разных уровней и было бы некорректно проводить партию между очень сильным и очень слабым учеником: последний может получить глубокую психологическую травму, поэтому Василий решает не проводить партию между парой учеников, если разница в их умениях превосходит m. Формально, он хочет проводить партию между учениками i,j только когда ∣∣si−sj∣∣≤m.

Также у Василия есть странное убеждение, что если партия проходит между учениками, суммарное умение которых превосходит его умение k, то он не сможет проанализировать эту партию и от неё не будет никакого толка. Формально, он **не**будет проводить партию между учениками i,j, если si+sj>k. Теперь Василий хочет посчитать, сколько партий ему придётся провести с такими ограничениями. Он позвал помочь ему с такой непростой задачей.

## Формат ввода

В первой строке вводятся три натуральных числа n,m,k(1≤n≤200000,0≤m,k≤109).

Во второй строке вводится n натуральных чисел s1,s2,...,sn(1≤si≤109).

## Формат вывода

В единственной строке выведите количество подходящих пар.

### Пример 1

| **Ввод**  IMG_256 | **Вывод**  IMG_257 |
| --- | --- |
| 4 1 5  1 3 2 3 | 3 |

### Пример 2

| **Ввод**  IMG_258 | **Вывод**  IMG_259 |
| --- | --- |
| 9 1 5  1 1 2 3 2 2 1 3 3 | 24 |

### Пример 3

| **Ввод**  IMG_260 | **Вывод**  IMG_261 |
| --- | --- |
| 10 4 15  1 2 3 4 5 6 7 8 9 10 | 24 |

## Примечания

В первом примере подойдут пары с индексами (1,3),(2,3),(3,4):

* (1,3): |s1−s3|=|1−2|=1≤1 и s1+s3=1+2=3≤5
* (2,3): |s2−s3|=|3−2|=1≤1 и s2+s3=3+2=5≤5
* (3,4): |s3−s4|=|2−3|=1≤1 и s3+s4=2+3=5≤5

# C. Замена вершины

|  |  |
| --- | --- |
| Ограничение времени | 4 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Корневое дерево называется бинарным, если у каждой его вершины не более двух потомков. Будем считать, что каждой вершине дерева поставлено в соответствие некоторое целое число, а потомки вершины указаны как левые и правые.

Поддерево некоторой вершины — набор вершин с рёбрами между ними такой, что каждая вершина набора является потомком этой вершины. Назовем левым поддеревом поддерево левого потомка, а правым поддеревом поддерево правого потомка. Тогда дерево является поисковым, если все ключи в левом поддереве **меньше**ключа в вершине, а все ключи в правом поддереве **больше либо** **равны**ключу в вершине.

Вам дано корректное бинарное поисковое дерево, содержащее в качестве ключей целые положительные числа в промежутке [l;r](l≤r). Необходимо вычислить число способов заменить ключ в **одной**из вершин дерева на число в интервале [l;r] так, чтобы дерево осталось правильным.

## Формат ввода

На вход в первой строке дано одно целое положительное число n — число вершин бинарного поискового дерева (1≤n≤3⋅105).

Во второй строке задано одно целое положительное число v — ключ в корневой вершине бинарного поискового дерева.

Далее в n−1 строках следует описание остальных вершин бинарного поискового дерева. В строке с номером i даны целые положительные числа k и p, а также символ c (1≤k≤109, 1≤p≤i−2, c∈{L,R}) — ключ в данной вершине, номер предка этой вершины (номером вершины является d−1, где d — номер строки, в которой дано описание этой строки), а также символ «L», если это левый потомок вершины, и «R», если правый.

В последней строке дано два целых положительных числа l и r (1≤l≤r≤109).

Гарантируется, что все ключи дерева лежат на промежутке [l;r].

## Формат вывода

Выведите одно целое неотрицательное число — ответ на задачу.

## Пример

| **Ввод**  IMG_256 | **Вывод**  IMG_257 |
| --- | --- |
| 5  5  2 1 L  2 2 R  7 1 R  7 4 R  1 7 | 9 |

## Примечания

В случае из примера есть следующие способы заменить ключи в вершинах:

* В вершине 1: ключи {3,4,6,7}
* В вершине 2: ключ {1}
* В вершине 3: ключи {3,4}
* В вершине 4: ключи {5,6}

Всего таких способов 9.

# D. Идеальная подматрица (бонус)

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

*Данная задача является бонусной.*

Даны два целых положительных числа n и m, а также матрица a из n строк и m столбцов, которая может содержать только числа 0 и 1. Назовем *подматрицей* четверку чисел (up,down,left,right) такую, что 1≤up≤down≤n, 1≤left≤right≤m. Назовем *идеальной подматрицей*такую подматрицу, что, если рассмотреть только элементы ai,j такие, что up≤i≤down,left≤j≤right, то среди них будет равное число нулей и единиц.

Необходимо найти количество идеальных подматриц заданной матрицы.

## Формат ввода

На вход в первой строке даны два целых положительных числа n и m (1≤n,m≤500) — число строк и столбцов в матрице соответственно.

В следующих n строках задана матрица a. Каждая из этих строк содержит m символов, которые являются либо нулем, либо единицей.

## Формат вывода

Выведите одно целое неотрицательное число — количество идеальных подматриц.

## Пример

| **Ввод**  IMG_256 | **Вывод**  IMG_257 |
| --- | --- |
| 3 4  0000  0011  0011 | 9 |

## Примечания

В примере из условия идеальными подматрицами являются (1,2,3,3), (1,2,3,4), (1,2,4,4), (2,2,2,3), (2,2,2,4), (2,3,2,3), (2,3,1,4), (3,3,2,3) и (3,3,1,4).

<https://site.ada.edu.az/~medv/acm/Docs%20e-olimp/Volume%2036/3535.htm>

3 4

1000

0000

0000

3 4

0000

1111

0000

2 3

000

000

1 3

0

0

0