

Задача А. Место встречи изменить нельзя

Имя входного файла: `rendezvous.in`
Имя выходного файла: `rendezvous.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Даны N точек. Найдите 2 из них, такие, что расстояние между ними минимально.

Формат входных данных

Первая строка входного файла содержит целое число N ($2 \leq N \leq 100\,000$) — количество точек. Каждая из следующих N строк содержит пару целых чисел X и Y , разделённых пробелом, — координаты ($-1\,000\,000\,000 \leq X, Y \leq 1\,000\,000\,000$). Все точки различны.

Формат выходных данных

Единственная строка выходного файла должна содержать координаты двух выбранных точек.

Пример

| <code>rendezvous.in</code> | <code>rendezvous.out</code> |
|----------------------------|-----------------------------|
| 4 | 0 0 |
| 0 0 | 0 1 |
| 0 1 | |
| 1 1 | |
| 1 0 | |

Задача В. Соединение и разъединение

Имя входного файла: `connect.in`
Имя выходного файла: `connect.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Вы когда-нибудь слышали про обход в глубину? Например, используя этот алгоритм, вы можете проверить является ли граф связным за время $O(E)$. Вы можете даже посчитать количество компонент связности за то же время.

А вы когда нибудь слышали про систему непересекающихся множеств? Используя эту структуру, вы можете быстро обрабатывать запросы “Добавить ребро в граф” и “Посчитать количество компонент связности в графе”.

А вы когда-нибудь слышали о *динамической* задаче связности? В этой задаче вам необходимо обрабатывать три типа запросов:

1. Добавить ребро в граф.
2. Удалить ребро из графа.
3. Посчитать количество компонент связности в графе.

Можно считать, что граф является неориентированным. Изначально граф является пустым.

Формат входных данных

В первой строке находятся два целых числа N и K — количество вершин и количество запросов, соответственно ($1 \leq N \leq 300\,000$, $0 \leq K \leq 300\,000$). Следующие K строк содержат запросы, по одному в строке. Каждый запрос имеет один из трех типов:

1. $+ u v$: Добавить ребро между вершинами u и v . Гарантируется, что такого ребра нет.
2. $- u v$: Удалить ребро между u и v . Гарантируется, что такое ребро есть.
3. $?$: Посчитать количество компонент связности в графе.

Вершины пронумерованы целыми числами от 1 до N . Во всех запросах $u \neq v$.

Формат выходных данных

Для каждого запроса типа ‘?’, Выведите количество компонент связности в момент запроса.

Примеры

| <code>connect.in</code> | <code>connect.out</code> |
|-------------------------|--------------------------|
| 5 11 | 5 |
| ? | 1 |
| + 1 2 | 1 |
| + 2 3 | 2 |
| + 3 4 | |
| + 4 5 | |
| + 5 1 | |
| ? | |
| - 2 3 | |
| ? | |
| - 4 5 | |
| ? | |

Задача С. Тандемные повторы

Имя входного файла: `tandems.in`
Имя выходного файла: `tandems.out`
Ограничение по времени: 3 секунды
Ограничение по памяти: 256 мегабайт

Дана строка s длины n .

Тандемным повтором в ней называются два вхождения какой-либо подстроки подряд. Иными словами, тандемный повтор описывается парой индексов $i < j$ такими, что подстрока $s[i \dots j]$ — это две одинаковые строки, записанные подряд.

От вас требуется посчитать количество пар индексом $i < j$ таких, что подстрока $s[i \dots j]$ является тандемным повтором.

Формат входных данных

Во входном файле находятся не более 30 тестов. Каждый тест состоит из единственной непустой строки, состоящей из символов **A,C,G,T**. Длина строки не превосходит 10^5 . Входной файл заканчивается строкой **0**.

Формат выходных данных

Для каждого теста выведите единственное число — количество тандемных повторов. Числа разделяйте переводами строк.

Примеры

| <code>tandems.in</code> | <code>tandems.out</code> |
|-------------------------|--------------------------|
| AGGA | 1 |
| AGAG | 1 |
| ATTCGATTCGATTCG | 9 |
| AAAA | 4 |
| 0 | |