Inverse. Количество инверсий

Имя входного файла: `inverse.in`
Имя выходного файла: `inverse.out`

Напишите программу, которая для заданного массива $A = \langle a_1, a_2, \ldots, a_n \rangle$ находит количество пар (i, j) таких, что $i < j$ и $a_i > a_j$.

Формат входного файла

Первая строка входного файла содержит натуральное число n $(1 \le n \le 50\,000)$ — количество элементов массива. Вторая строка содержит n попарно различных элементов массива A.

Формат выходного файла

В выходной файл выведите одно число — ответ на задачу.

Пример

inverse.in	inverse.out
4 1 2 4 5	0
4 5 4 2 1	6

Suffarr. Сортировка суффиксов

Имя входного файла: `suffarr.in`
Имя выходного файла: `suffarr.out`

Непустая строка S называется суффиксом строки T, если она получается из нее удалением произвольного (возможно, нулевого) количества начальных символов. Например, строка `summer` имеет следующие суффиксы: `summer`, `ummer`, `mmer`, `mer`, `er`, `r`. Напишите программу, которая сортирует все суффиксы строки T.

Формат входного файла

Входной файл содержит строку T, состоящую из строчных латинских букв. Длина T не превышает $5\,000$ символов.

Формат выходного файла

В выходной файл выведите последовательность, i-ый элемент которой задает индекс, с которого начинается i-ый в лексикографическом порядке суффикс.

Пример

suffarr.in	suffarr.out
sis	2 3 1
summer	5 4 3 6 1 2

Cycles4. 4-циклы (*)

Имя входного файла: `cycles4.in`
Имя выходного файла: `cycles4.out`

Требуется найти в графе количество простых циклов длины четыре. Цикл называется простым, если он не содержит повторяющихся вершин.

Формат входного файла

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно $(1 \le n \le 1\,000,\ 1 \le m \le 10\,000)$. Следующие m строк содержат описания ребер по одному на строке. Ребро номер i описывается двумя натуральными числами b_i, e_i — номерами концов ребра $(1 \le b_i, e_i \le n)$. Петель и кратных ребер в графе нет.

Формат выходного файла

Первая строка выходного файла должна содержать одно натуральное число — количество простых циклов длины четыре. Гарантируется, что ответ не превышает 10^9.

Пример

cycles4.in	cycles4.out
4 6 1 2 2 3 3 4 4 1 1 3 4 2	3

QSort. Интеллект против QSort

Имя входного файла: `qsort.in`
Имя выходного файла: `qsort.out`

Для сортировки последовательности чисел широко используется быстрая сортировка — QSort. Ниже приведена программа, которая сортирует массив a, используя данный алгоритм.

```
var a : array [1..N] of integer;

procedure QSort(left, right : integer);
var m, i, j, t : integer;
begin
  m := a[(left+right) div 2];
  i := left; j := right;
  repeat
    while a[i] < m do inc(i); {первый while}
    while a[j] > m do dec(j); {второй while}
    if i <= j then begin
      t := a[i]; a[i] := a[j]; a[j] := t;
      inc(i); dec(j);
    end;
  until i > j;
  if j > left then QSort(left, j);
  if i < right then QSort(i, right);
end;
```

Хотя QSort является самой быстрой сортировкой в среднем, существуют тесты, на которых она работает очень долго. Будем оценивать время работы алгоритма количеством сравнений, сделанных с элементами массива (т.е. суммарным количеством сравнений в первой и втором while). Требуется написать программу, генерирующую тест, на котором быстрая сортировка сделает наибольшее число таких сравнений.

Формат входного файла

Во входном файле задано число N ($1 \le N \le 70\,000$).

Формат выходного файла

В выходной файл вывести перестановку чисел от 1 до N, на которой быстрая сортировка выполнит максимальное число сравнений. Если таких перестановок несколько, вывести любую из них.

Пример

qsort.in	qsort.out
3	1 3 2