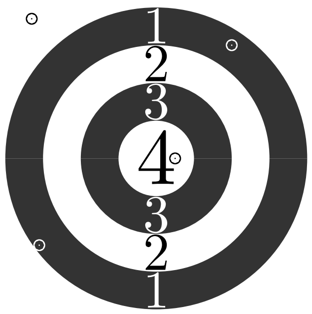
Darts. Дартс

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

Недавно в ЛКШ проводились соревнования по дартсу. В качестве мишени использовался круг радиуса $10 \times R$, разбитый на R колец толщины 10.



ЛКШата бросали в мишень по N дротиков. Вам поручили написать программу, которая по координатам мест, куда попали дротики участника, сообщала бы, какие броски ушли в «молоко» (не попали в мишень), какие попали во внешнее (первое) кольцо, какие угодили в следующее и так далее вплоть до центрально круга ($R+1$-ое кольцо).

Формат входного файла

В первой строке входного файла два целых числа R и N ($1 \leqslant R \leqslant 100; 1 \leqslant N \leqslant 10^6$). Далее следуют N строк, в каждой из которых два числа, по модулю не превышающие 1000 — координаты попадания очередного дротика. Центр мишени при этом считается началом координат.

Формат выходного файла

В выходном файле должно содержаться $R+1$ строчка. В первой строке номера бросков, угодивших в «молоко». Во второй строке номера бросков, попавших во внешнее кольцо. Во третьей строке номера бросков, попавших во второе кольцо.

...

В $R+1$ строке номера бросков, попавших в центральный круг. Если бросок попал на границу двух колец, то считается, что он попал в кольцо, находящееся ближе к центру.

Пример

darts.in	darts.out
4 4 20 30 5 0 -33 37 -31 -23	3 1 4 2
1 3 10 0 0 10 10 10	3 1 2

Примечание: если в какое-то из колец не попало ни одного броска, то соответствующая строка должна быть пустой (см. первый пример).

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 ($1 \leqslant n \leqslant 50\,000$) — 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

Kth. K-ый минимум

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

Напишите программу, которая находит k-ое в возрастающем порядке число в массиве $A = \langle a_1, a_2, \ldots, a_n \rangle$.

Массив A задается с помощью полинома $P(x) = 132x^3 + 77x^2 + 1345x + 1577$: $a_i = P(i) \bmod 1743$.

Формат входного файла

Входной файл содержит два натуральных числа n и k ($1 \leqslant k \leqslant n \leqslant 50\,000$).

Формат выходного файла

В выходной файл выведите одно число — ответ на задачу.

Пример

kth.in	kth.out
1 1	1388
10 1	402

Deques. Деки на 6-и мегабайтах

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

Напишите программу, которая умеет оперировать большим количеством деков. Дек — это "очередь с двумя концами".

Формат входного файла

Первая строка входного файла содержит общее количество команд n ($0 \leqslant n \leqslant 150\,000$). Каждая из следующих n строк содержит описание команды:

- "`pushfront A B`" — вставить число B в начало дека A;
- "`pushback A B`" — вставить число B в конец дека A;
- "`popfront A`" — удалить первый элемент дека A;
- "`popback A`" — удалить последний элемент дека A.

Для каждой команды параметры A и B — целые числа от 1 до 150 000 включительно.

Формат выходного файла

Для каждой команды `popfront` или `popback` выведите удаляемое число. Гарантируется, что перед выполнением команды удаления соответствующий дек не пуст.

Пример

deques.in	deques.out
9 pushfront 1 71819 pushback 2 71820 pushback 1 1 popfront 1 popfront 1 pushfront 2 10 pushback 2 11 popback 2 popback 2	71819 1 11 71820

Pref. Преферанс (*)

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

В новой колоде 32 карты для преферанса расположены в следующем порядке (сверху вниз): червы, бубны, трефы, пики. В каждой масти сначала лежит семерка, под ней — восьмерка, затем девятка, десятка, валет, дама, король, туз. Тасовка карт осуществляется так: 16 карт, составляющих верхнюю половину колоды, распределяются между картами нижней половины колоды. Каждая карта верхней половины вставляется в нижнюю колоду так, что в получившейся колоде карты верхней половины идут в том же порядке, в котором они были изначально. Любое число карт верхней половины можно располагать как над верхней, так и под нижней картой второй половины колоды, а также между любыми двумя соседними картами нижней половины колоды. Такие действия повторяются не более пяти раз.

Требуется написать программу, которая указывает, как надо осуществить тасовку, чтобы в итоге получить заранее заданное расположение карт.

Формат входного файла

Единственная строка входного файла содержит информацию о порядке карт, в котором они должны оказаться после тасовки. Карты перечислены сверху вниз.

Каждая карта обозначается латинской буквой, указывающей масть (пики — `S`, трефы — `C`, бубны — `D`, червы — `H`), и номиналом (туз — `A`, король — `K`, дама — `Q`, валет — `J`, десятка — `0`, остальные — в соответствии со своим значением: `9`, `8`, `7`).

Формат выходного файла

В первую строку выходного файла необходимо вывести целое число N ($0 \leqslant N \leqslant 5$) — количество шагов тасовки. Следующие N строк должны содержать информацию о каждом шаге тасовки. Каждая строка при этом должна содержать 16 чисел, указывающих номера позиций, на которых оказываются снятые карты. Номера позиций выводятся в порядке возрастания и разделены пробелами. Нумерация позиций производится сверху вниз от 1 до 32.

Пример

В примере входная строка для удобства разбита на две. Во входных файлах при тестировании задачи будет ровно одна строка, завершенная переводом строки.

pref.in
C7 H7 C8 H8 C9 H9 C0 H0 S7 D7 S8 D8 S9 D9 S0 D0 SJ DJ SQ DQ SK DK SA DA CJ HJ CQ HQ CK HK CA HA
pref.out
2 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 1 2 3 4 5 6 7 8 25 26 27 28 29 30 31 32