Задание 1

Сдать задание нужно до 9 октября. (9:00).
Контест: https://contest.yandex.ru/contest/4974/enter/
От каждой задачи нужно решить только один свой вариант. Варианты прописаны в ведомости:
https://drive.google.com/open?id=1pr1PVEYPKmFCC7Nn-jZSIjIuE-eDbJMTbFVWkugR0T8

Задача № 1 (2 балла)

1_1. Найти, на сколько нулей оканчивается n! = 1 * 2 * 3 * … * n.
n ≤ 1000.

in	out
25	6

1_2. Вывести разложение натурального числа n на простые множители. Простые множители должны быть упорядочены по возрастанию и разделены пробелами.
$2 \le n \le 10^6$.

in	out
75	3 5 5

1_3. Даны четыре неотрицательных числа a, b, c и d. Сложить две рациональные дроби a/b и c/d, а их результат представить в виде несократимой дроби m/n. Вывести числа m и n.
a, b, c, d ≤ 1000.

in	out
3 10 5 18	26 45

1_4. Дано натуральное число N. Представить N в виде A + B, так, что НОД(A, B) максимален, A ≤ B. Вывести A и B. Если возможно несколько ответов - вывести ответ с минимальным A.
$n \le 10^7$.

in	out
35	7 28

1_5. Вывести квадраты натуральных чисел от 1 до n, используя только O(n) операций сложения и вычитания (умножением пользоваться нельзя).
n ≤ 1000.

in	out
5	1 4 9 16 25

1_6. Дан массив целых чисел A[0..n). Не используя других массивов переставить элементы массива A в обратном порядке за O(n).
n ≤ 10000.

in	out
4 3 9 -5 2	2 -5 9 3

1_*. Найти все простые числа в диапазоне [2..n] за O(n).

in	out
15	2 3 5 7 11 13

Задача № 2 (3 балла)

В каждой задаче, где начальными данными является массив вначале вводится количество элементов, а затем и сами элементы массива.

2_1. Даны два массива целых чисел одинаковой длины A[0..n-1] и B[0..n-1]. Необходимо найти первую пару индексов i0 и j0, i0 ≤ j0, такую что A[i0] + B[j0] = max {A[i] + B[j], где 0 <= i < n, 0 <= j < n, i <= j}. Время работы - O(n).

n ≤ 100000.

in	out
4 4 -8 6 0 -10 3 1 1	0 1

2_2. Вычислить площадь выпуклого n-угольника, заданного координатами своих вершин. Сначала вводится количество вершин, затем последовательно целочисленные координаты всех вершин в порядке обхода по часовой стрелке.

n < 1000, координаты < 10000.

Указание. Для вычисления площади n-угольника можно посчитать сумму ориентированных площадей трапеций под каждой стороной многоугольника.

in	out
3 0 1 1 0 2 2	1.5

2_3. Даны два строго возрастающих массива целых чисел A[0..n) и B[0..m) и число k. Найти количество таких пар индексов (i, j), что A[i] + B[j] = k. Время работы O(n + m).

n, m ≤ 100000.

Указание. Обходите массив B от конца к началу.

in	out
4 -5 0 3 18 5 -10 -2 4 7 12 7	3

2_4. “Считалочка”. В круг выстроено N человек, пронумерованных числами от 1 до N. Будем исключать каждого k-ого до тех пор, пока не уцелеет только один человек. (Например, если N=10, k=3, то сначала умрет 3-й, потом 6-й, затем 9-й, затем 2-й, затем 7-й, потом 1-й, потом 8-й, за ним - 5-й, и потом 10-й. Таким образом, уцелеет 4-й.) Необходимо определить номер уцелевшего.

N, k ≤ 10000.

in	out
10 3	4

2_*. (3 балла) Дан массив целых чисел A[0..n). Массив произвольным образом заполнен натуральными числами из диапазона [0..n - 1). Одно или несколько значений в массиве может повторяться. Необходимо найти любой повтор за O(n), памяти O(1). Исходный массив хранить можно, модифицировать нельзя.

n ≤ 10000.

in	out
8 1 2 4 5 6 1 0 3	1

Задача № 3 (4 балла)

3_1. Дан отсортированный массив целых чисел A[0..n-1] и массив целых чисел B[0..m-1]. Для каждого элемента массива B[i] найдите минимальный индекс k минимального элемента массива A, равного или превосходящего B[i]: A[k] >= B[i]. Если такого элемента нет, выведите n. Время работы поиска k для каждого элемента B[i]: O(log(k)).
n, m ≤ 10000.
Формат входных данных.
В первой строчке записаны числа n и m. Во второй и третьей массивы A и B соответственно.

in	out
2 1 1 2 2	1
4 3 2 4 5 7 4 6 1	1 3 0

3_2. Дан массив целых чисел A[0..n-1]. Известно, что на интервале [0, m] значения массива строго возрастают, а на интервале [m, n-1] строго убывают. Найти m за O(log m).
2 ≤ n ≤ 10000.

in	out
10 1 2 3 4 5 6 7 6 5 4	6

3_3. Даны два массива неповторяющихся целых чисел, упорядоченные по возрастанию. A[0..n-1] и B[0..m-1]. n >> m. Найдите их пересечение. Требуемое время работы: O(m * log k), где k - позиция элементта B[m-1] в массиве A.. В процессе поиска очередного элемента B[i] в массиве A пользуйтесь результатом поиска элемента B[i-1].
n, k ≤ 10000.

in	out
5 3 1 2 3 4 5 1 3 5	1 3 5

3_4. Дан отсортированный массив различных целых чисел A[0..n-1] и массив целых чисел B[0..m-1]. Для каждого элемента массива B[i] найдите минимальный индекс элемента массива A[k], ближайшего по значению к B[i]. Время работы поиска для каждого элемента B[i]: O(log(k)).
n ≤ 110000, m ≤ 1000.

in	out
3 10 20 30 3 9 15 35	0 0 2

3 10 20 30 4 8 9 10 32	0 0 0 2

Задача № 4 (3 балла)

Во всех задачах из следующего списка следует написать структуру данных, обрабатывающую команды push* и pop*.

Формат входных данных.

В первой строке количество команд n. n ≤ 1000000.

Каждая команда задаётся как 2 целых числа: a b.

a = 1 - push front

a = 2 - pop front

a = 3 - push back

a = 4 - pop back

Для очереди используются команды 2 и 3. Для дека используются все четыре команды.

Если дана команда pop*, то число b - ожидаемое значение.Если команда pop вызвана для пустой структуры данных, то ожидается “-1”.

Формат выходных данных.

Требуется напечатать YES - если все ожидаемые значения совпали. Иначе, если хотя бы одно ожидание не оправдалось, то напечатать NO.

4_1. Реализовать очередь с динамическим зацикленным буфером.

in	out
3 3 44 3 50 2 44	YES
2 2 -1 3 10	YES
2 3 44 2 66	NO

4_2. Реализовать дек с динамическим зацикленным буфером.

in	out
3 1 44 3 50 2 44	YES
2 2 -1 1 10	YES

2 3 44 4 66	NO

4_3. Реализовать очередь с помощью двух стеков. Использовать стек, реализованный с помощью динамического буфера.

in	out
3 3 44 3 50 2 44	YES
2 2 -1 3 10	YES
2 3 44 2 66	NO

4_*. Реализовать очередь при помощи нескольких стеков. Каждая операция pop front и push back должна выполняться за O(1).

Задача № 5 (4 балла)

Решение всех задач данного раздела предполагает использование стека. Способ реализации стека может быть любым (список/динамический массив).

5_1. Скобочная последовательность.

Дан фрагмент последовательности скобок, состоящей из символов (){}[].

Требуется определить, возможно ли продолжить фрагмент в обе стороны, получив корректную последовательность.

Длина исходной последовательности ≤ 800000.

Формат входных данных. Строка, содержащая символы (){}[] и, возможно, перевод строки.

Формат выходных данных. Если возможно - вывести минимальную корректную последовательность, иначе - напечатать "**IMPOSSIBLE**".

in	out
{}[[[[{}[]	{}[[[[{}[]]]]]
{][[[[{}[]	IMPOSSIBLE
](){}[](({}	{[](){}[](({}))

5_2. Стековые анаграммы.

Пара слов называется стековой анаграмой, если одно слово можно получить из другого, проведя последовательность стековых операций с его буквами (взять очередную букву исходного слова и поместить ее в стек; взять букву из стека и добавить ее в конец выходного слова).

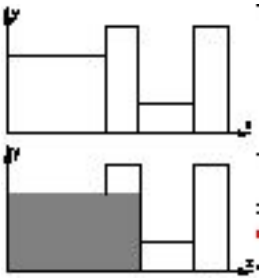
Для заданной пары слов требуется определить, можно ли выполнить последовательность стековых операций, переводящую первое слово во второе. Все буквы в слове различные.

Длина анаграммы ≤ 10000.

Формат входных данных. Пара слов, являющихся анаграммой.
Формат выходных данных. **YES**, если последовательность стековых операций существует и **NO** в противном случае.

in	out
STOL SLOT	YES
ABC CAB	NO

5_3. Прямоугольники.
Дана последовательность *N* прямоугольников различной ширины и высоты (w_i,h_i). Прямоугольники расположены, начиная с точки (0, 0), на оси OX вплотную друг за другом (вправо). Требуется найти *M* - площадь максимального прямоугольника (параллельного осям координат), который можно вырезать из этой фигуры.



Время работы - O(n).
Формат входных данных. В первой строке задано число *N* (1 ≤ *N* ≤ 10000). Далее идет *N* строк. В каждой строке содержится два числа width и height: ширина и высота i-го прямоугольника. (0 < width ≤ 10000, 0 ≤ height ≤ 10000)
Формат выходных данных. вывести число *M*. (0 ≤ M ≤ 10^9).

in	out
4 30 30 10 40 20 10 10 40	1200
1 1 3000	3000
3 1 1 1 3 3 2	8

5_4. Вычисление выражения.
Дано выражение в инфиксной записи. Вычислить его, используя перевод выражения в постфиксную запись. Выражение не содержит отрицательных чисел.
Количество операций ≤ 100.
Формат входных данных. Строка, состоящая их символов “0123456789-+*/()”
Гарантируется, что входное выражение корректно, нет деления на 0, вычислимо в целых числах. Деление целочисленное.
Формат выходных данных.

Значение выражения.

in	out
1 + 2	3
200-(123+34*2)+(48-2)	55

Задача № 6 (4 балла)

Решение всех задач данного раздела предполагает использование кучи.

6_1. Жадина.

Вовочка ест фрукты из бабушкиной корзины. В корзине лежат фрукты разной массы. Вовочка может поднять не более K грамм. Каждый фрукт весит не более K грамм. За раз он выбирает несколько самых тяжелых фруктов, которые может поднять одновременно, откусывает от каждого половину и кладет огрызки обратно в корзину. Если фрукт весит нечетное число грамм, он откусывает большую половину. Фрукт массы 1гр он съедает полностью.

Определить за сколько подходов Вовочка съест все фрукты в корзине.

Формат входных данных. Вначале вводится n - количество фруктов и n строк с массами фруктов. n ≤ 50000.

Затем K - "грузоподъемность". K ≤ 1000.

Формат выходных данных. Неотрицательное число - количество подходов к корзине.

in	out
3 1 2 2 2	4
3 4 3 5 6	5
7 1 1 1 1 1 1 1 3	3

6_2. Быстрое сложение.

Для сложения чисел используется старый компьютер. Время, затрачиваемое на нахождение суммы двух чисел равно их сумме.

Таким образом для нахождения суммы чисел 1,2,3 может потребоваться разное время, в зависимости от порядка вычислений.

((1+2)+3) -> 1+2 + 3+3 = 9

((1+3)+2) -> 1+3 + 4+2 = 10

((2+3)+1) -> 2+3 + 5+1 = 11

Требуется написать программу, которая определяет минимальное время, достаточное для вычисления суммы заданного набора чисел.

Формат входных данных. Вначале вводится n - количество чисел. Затем вводится n строк - значения чисел (значение каждого числа не превосходит 10^9, сумма всех чисел не превосходит 2*10^9).

Формат выходных данных. Натуральное число - минимальное время.

in	out
5	45

5 2 3 4 6	
5 3 7 6 1 9	56

6_3. Тупики.

На вокзале есть некоторое количество тупиков, куда прибывают электрички. Этот вокзал является их конечной станцией. Дано расписание движения электричек, в котором для каждой электрички указано время ее прибытия, а также время отправления в следующий рейс. Электрички в расписании упорядочены по времени прибытия. Когда электричка прибывает, ее ставят в свободный тупик с минимальным номером. При этом если электричка из какого-то тупика отправилась в момент времени X, то электричку, которая прибывает в момент времени X, в этот тупик ставить нельзя, а электричку, прибывающую в момент X+1 — можно.

В данный момент на вокзале достаточное количество тупиков для работы по расписанию.

Напишите программу, которая по данному расписанию определяет, какое минимальное количество тупиков требуется для работы вокзала.

Формат входных данных. Вначале вводится n - количество электричек в расписании. Затем вводится n строк для каждой электрички, в строке - время прибытия и время отправления. Время - натуральное число от 0 до 10^9. Строки в расписании упорядочены по времени прибытия.

Формат выходных данных. Натуральное число - минимальное количеством тупиков.

Максимальное время: 50мс, память: 5Мб.

in	out
1 10 20	1
2 10 20 20 25	2
3 10 20 20 25 21 30	2

6_4. Скользящий максимум.

Дан массив натуральных чисел A[0..n), n не превосходит 10^8. Так же задан размер некотрого окна (последовательно расположенных элементов массива) в этом массиве k, k<=n. Требуется для каждого положения окна (от 0 и до n-k) вывести значение максимума в окне. Скорость работы O(n log n), память O(n).

Формат входных данных. Вначале вводится n - количество элементов массива. Затем вводится n строк со значением каждого элемента. Затем вводится k - размер окна.

Формат выходных данных. Разделенные пробелом значения максимумов для каждого положения окна.

in	out
3 1 2 3 2	2 3
9 0 7 3 8 4 5 10 4 6 4	8 8 8 10 10 10