**Задача № 1**

1. A. Определить количество счастливых билетов. Счастливым билетом называется билет с 6-значным номером, у которого сумма первых 3 десятичных цифр равна сумме 3 последних десятичных цифр.
2. B. Даны натуральные числа a, b, c, d. Выяснить, можно ли в прямоугольник со сторонами a, b целиком поместить прямоугольник со сторонами c, d. Стороны прямоугольников должны быть параллельны. Вывести YES или NO.
3. C. Посчитать количество единиц в двоичной записи натурального числа n.
4. D. Даны натуральные числа a и b. Вычислить остаток от деления a на b, не используя операции %.
5. E. Даны натуральные числа a и b. Вычислить частное от деления a на b, не используя операции /.
6. F. Дано значение n!. Вычислить n.
7. G. Вывести десятичную запись натурального числа в обратном порядке (например, для 123 необходимо вывести 321).

**Задача № 2**

1. H. Найти, на сколько нулей оканчивается n! = 1 \* 2 \* 3 \* … \* n.
2. I. Вывести разложение натурального числа n на простые множители. Простые множители должны быть упорядочены по возрастанию и разделены пробелами. Пример:

|  |  |
| --- | --- |
| in | out |
| 75 | 3 5 5 |

1. J. Даны натуральные числа n и m. 0 < m <= 36. Вывести число n в системе счисления по основанию m. Используйте буквы A-Z для обозначения значения в разряде большее 9. Пример:

|  |  |
| --- | --- |
| in | out |
| 127 16 | 7F |

1. K. Даны две рациональные дроби: a/b и c/d. Сложить их и результат представьте в виде несократимой дроби m/n. Вывести числа m и n.
2. L. Дано натуральное число N. Представить N в виде A + B, так, что НОД(A, B) максимален, A <= B. Вывести A и B.
3. M. Вывести квадраты натуральных чисел от 1 до n, используя только O(n) операций сложения и вычитания (умножением пользоваться нельзя).
4. N. Вычислить площадь выпуклого n-угольника, заданного координатами своих вершин. Вначале вводится количество вершин, затем координаты каждой вершины. Пример:

|  |  |
| --- | --- |
| in | out |
| 3 0 1 1 0 2 2 | 1.5 |

**Задача № 3**

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

1. O. 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). Пример:

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

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

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

1. Q. Даны два строго возрастающих массива целых чисел A[0..n) и B[0..m). Найти количество общих элементов в этих массивах за O(n + m). Пример:

|  |  |
| --- | --- |
| in | out |
| 4  -5 3 9 12  3  -1 3 12 | 2 |

1. R. Дан массив целых чисел A[0..n). Для каждого участка из m стоящих рядом членов вычислить сумму элементов. Общее число действий O(n). На вход вначале поступает массив, а затем число m. Пример:

|  |  |
| --- | --- |
| in | out |
| 4  3 9 -5 2  3 | 7 6 |

1. S. Дан массив целых чисел A[0..n). В массиве находятся все числа от 0 до n кроме одного. Найти пропущенное число за O(n), памяти O(1). Пример:

|  |  |
| --- | --- |
| in | out |
| 4  3 0 2 4 | 1 |

1. T. 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). Подсказка: свести задачу к №17.

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

1. U. “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.

|  |  |
| --- | --- |
| in | out |
| 10 3 | 4 |

**Задача № 4**

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 минимального элемента массива, равного или превосходящего b: A[k] >= B[i]. Если такого элемента нет, выведите n. Время работы поиска каждого элемента B[i]: O(log(k)).

|  |  |
| --- | --- |
| in | out |
| 2 1  1 2  2 | 1 |

1. Дан массив целых чисел А[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 n).

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

1. Даны два массива неповторяющихся целых чисел, упорядоченные по неубыванию. A[0..n-1] и B[0..m-1]. n >> m. Найдите их пересечение. Требуемое время работы: O(m \* log n)

|  |  |
| --- | --- |
| in | out |
| 3  5  1 3 5  1 2 3 4 5 | 1 3 5 |

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 элемента массива A[k], ближайшего по значению к B[i]. Время работы поиска для каждого элемента B[i]: O(log(k)).

|  |  |
| --- | --- |
| in | out |
| 3  10 20 30  3  9 18 35 | 0 1 2 |

**Задача № 5**

Во всех задачах из следующего списка следует написать код, обрабатывающий команды push\* и pop\*.

В первой строке количество команд.  
Каждая команда задаётся как 2 целых числа: a b.

a=1 - push back

a=2 - pop back

a=3 - push front

a=4 - pop front

Для стека используются команды 1 и 2. Для очереди используются команды 2 и 3. Для дека используются все 4 команды.

Если дана команда pop\*, то число b - ожидаемое значение.

Требуется напечатать YES - если все ожидаемые значения совпали, в противном случае - NO.

1. Реализовать очередь с динамическим буфером.
2. Реализовать дек с динамическим буфером.
3. Реализовать стек с помощью односвязного списка.
4. Реализовать очередь с помощью односвязного списка.
5. Реализовать дек с помощью двусвязного списка.
6. Реализовать очередь с помощью стека. Использовать стек, реализованный с помощью динамического буфера.

**Задача № 6 (Стек)**

**1. Скобочная последовательность.**

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

Требуется определить, возможно ли получить корректную последовательность, добавляя 0 или больше скобок слева и/или справа.

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

Входные данные: строка, содержащая символы (){}[] и, возможно, перевод строки.

**2. Стековые анаграммы.**

Слово "вертикаль" можно составить из букв слова "кильватер". Такая пара называется анаграммой. Для некоторых анаграмм 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.

Входные данные: пара слов, являющихся анаграммой  
Выходные данные: **YES**, если последовательность стековых операций существует и **NO** в противном случае.

|  |  |
| --- | --- |
| in | out |
| STOL SLOT | YES |
| ABC CAB | NO |

**3. Польская нотация.**

Дано выражение в инфиксной записи. Например : (123+34)-12+(48-2).

Требуется напечатать его в польской нотации: + - + 123 34 12 - 48 2

Выражение может состоять из неотрицательных целых чисел, скобок и операций - + и скобок. Гарантируется, что введенное выражение корректно. Выражение не содержит унарный минус.

**Входные данные:** строка, состоящая их символов “0123456789-+()”

Гарантируется, что входное выражение корректно.

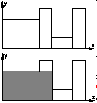
Максимальная длина строки 10^6 символов.

**Выходные данные:** строка из символов “ 0123456789-+”

Операторы и операнды в выходной строке разделяются пробелом.

|  |  |
| --- | --- |
| in | out |
| 123+(12-34) | + 123 - 12 34 |
| 123+12-34 | - + 123 12 34 |

**4. Прямоугольники.**

Дана последовательность *N* прямоугольников различной ширины и высоты (wi,hi). Прямоугольники расположены, начиная с точки (0, 0), на оси ОХ вплотную друг за другом (вправо). Требуется найти *M* - площадь максимального прямоугольника (параллельного осям координат), который можно вырезать из этой фигуры.  
  
Время работы - O(N).  
Входные данные: В первой строке задано число *N* (1 ≤ *N* ≤ 8000). Далее идет *N* строк. В каждой строке содержится два числа: ширина и высота i-го прямоугольника  
Выходные данные: Вывести число *M*.

Максимальное время: 100мс, память: 5Мб.

|  |  |
| --- | --- |
| in | out |
| 4  30 30  10 40  20 10  10 40 | 1200 |
| 1  1 3000 | 3000 |
| 5  10 10  10 20  10 30  10 40  10 10 | 600 |

**Задача № 7 (Куча)**

**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. Вовочка подходит к корзине несколько раз. За один подход он выбирает несколько самых тяжелых фруктов, котор ые может поднять одновременно, откусывает от каждого половину и кладет огрызки обратно в корзину. Так он подходит к корзине снова и снова. Если фрукт весит нечетное число грамм, он откусывает большую половину. Фрукт массы 1гр он съедает полностью.  
У Вовочки просыпается совесть, если в очередной подход он пытается взять из корзины не меньше половины фруктов от той массы фруктов, которая была перед подходом, в этом случае он останавливается.

Определить массу оставшихся в корзине фруктов.

Входные данные: Вначале вводится n - количество фруктов и n строк с массами фруктов. Затем K - "грузоподъемность".

Выходные данные: Натуральное число - масса оставшихся фруктов.

Максимальное время: 100мс, память: 5Мб.

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

**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 зачисимости 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.

Входные данные: 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).

Выходные данные: натуральное число - минимальное время.

|  |  |
| --- | --- |
| in | out |
| 5  5 2 3 4 6 | 45 |
| 5  3 7 6 1 9 | 56 |

**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.  
Входные данные: 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.  
Выходные данные: 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 |

**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 k)

Входные данные: вначале вводится n - количество элементов массива. Затем вводится n строк со значением каждого элемента. Затем вводится k - размер окна.

Выходные данные: 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 |

**Задача № 8 \* (необязательная)**

**Принтеры**

В подвале МГТУ им Баумана стоит старый, но производительный матричный принтер.

Притер постоянно загружен, поэтому для него написали программу, отправляющую задания на печать в зависимости от приоритета задачи. Особенность алгоритма в том, что когда в очереди более одного задания, выбирается то, которое делит очередь пополам. Так что в левой половине остаются задачи с более высоким приоритетом и те что поступили раньше, а в правой - с более низким приоритетом, и те что поставлены позже. (Если размер очереди чётное число - выбирается такой элемент, что левая "половина" на 1 меньше правой).

В связи с обновлением парка оборудования программу требуется перенести на машину с другой архитектурой.

К сожалению, исходный код программы утрачен. Вам нужно реализовать описанный алгоритм.

Входные данные:

Натуральное число N - Количество задач.

В следующих N строках по три числа: время поступления задачи, длительность выполнения задачи, приоритет.

Числа в первой колонке - неубывающая последовательность.

Формат вывода:

последовательность номеров заданий в которой они отправятся на печать.