Алгоритмы и структуры данных

Задание к лабораторной работе №3. Быстрая сортировка, сортировки за линейное время

Лабораторная работа посвящена методу Разделяй и властвуй, быстрой сортировке, а также сортировкам за линейное время. Аналогично предыдущим лабораторным работам, есть два способа ее выполнения и защиты:

1 **Базовый уровень.** Решается 3 задачи по вариантам. Варианты в табличке внизу, номер вашего варианта соответствует вашему номеру в списке группы. Посмотреть свой номер можно, например, в журнале успеваемости по дисциплине. В этом случае максимум за защиту можно получить 4 балла. В сумме с самой работой (0,5 балла) и отчетом (1 балл) получается 5,5 балла, что достаточно для зачета.

2 **Продвинутый уровень.** Решаются все задачи или минимум 5 задач, причем 3 из них - исходя из вашего варианта, а остальные две - по выбору. В этом случае вы сможете получить максимальные 7,5 баллов. *Задача №10 - не обязательная.*

Вариант	Номера задач	Вариант	Номера задач
1	1,4,7	16	1,2,5
2	1,4,8	17	1,2,6
3	1,4,9	18	1,2,7
4	1,5,6	19	1,2,8
5	1,5,7	20	1,2,9
6	1,5,8	21	1,3,4
7	1,5,9	22	1,3,5
8	1,6,7	23	1,3,6
9	1,6,8	24	1,3,7
10	1,6,9	25	1,3,8
11	1,7,8	26	1,3,9
12	1,7,9	27	1,4,5
13	1,8,9	28	1,4,6
14	1,2,3		
15	1,2,4		

Если какая-то из задач вашего варианта кажется вам слишком сложной, вы можете решить другую задачу, которая вам больше нравится, или задачу №10.

1 задача. Улучшение Quick sort

1. Используя *псевдокод* процедуры `Randomized - QuickSort`, а так же `Partition` из презентации к Лекции 3 (страницы 8 и 12), напишите программу быстрой сортировки на Python и проверьте ее, создав несколько рандомных массивов, подходящих под параметры:

   - **Формат входного файла (input.txt).** В первой строке входного файла содержится число n ($1 \leq n \leq 10^4$) — число элементов в массиве. Во второй строке находятся n различных целых чисел, ***по модулю*** не превосходящих 10^9.
   - **Формат выходного файла (output.txt).** Одна строка выходного файла с отсортированным массивом. Между любыми двумя числами должен стоять ровно один пробел.
   - Ограничение по времени. 2 сек.
   - Ограничение по памяти. 256 мб.
   - Для проверки можно выбрать наихудший случай, когда сортируется массив рамера $10^3, 10^4, 10^5$ чисел порядка 10^9, отсортированных в обратном порядке; наилучший, когда массив уже отсортирван, и средний - случайный. Сравните на данных сетах `Randomized-QuickSort` и простой `QuickSort`. (А также есть `Median-QuickSort`, см. задание 10.2; и `Tail-Recursive-QuickSort`, см. Кормен. 2013, стр. 217)

2. **Основное задание.** Цель задачи - переделать данную реализацию рандомизированного алгоритма быстрой сортировки, чтобы она работала быстро даже с последовательностями, содержащими много одинаковых элементов. Чтобы заставить алгоритм быстрой сортировки эффективно обрабатывать последовательности с несколькими уникальными элементами, нужно заменить двухстороннее разделение на трехстороннее (смотри в Лекции 3 слайд 17). То есть ваша новая процедура разделения должна разбить массив на три части:

   - $A[k] < x$ для всех $\ell + 1 \leq k \leq m_1 - 1$
   - $A[k] = x$ для всех $m_1 \leq k \leq m_2$
   - $A[k] > x$ для всех $m_2 + 1 \leq k \leq r$

   - Формат входного и выходного файла аналогичен п.1.
   - Аналогично п.1 этого задания сравните `Randomized-QuickSort` +с `Partition` и ее с `Partition3` на сетах случайных данных, в которых содержатся всего несколько уникальных элементов при $n = 10^3, 10^4, 10^5$. Что быстрее, `Randomized-QuickSort` +с `Partition3` или `Merge-Sort`?
   - Пример:

input.txt	output.txt
5 2 3 9 2 2	2 2 2 3 9

2 задача. Анти-quick sort

Для сортировки последовательности чисел широко используется быстрая сортировка - `QuickSort`. Далее приведена программа на языке ~~Pascal~~ Python, которая сортирует массив a, используя этот алгоритм.

```
def qsort (left, right):
        key = a [(left + right) // 2]
        i = left
        j = right
        while i <= j:
                while a[i] < key: # first while
                        i += 1
                while a[j] > key : # second while
                        j -= 1
                if i <= j :
                        a[i], a[j] = a[j], a[i]
                        i += 1
                        j -= 1
        if left < j:
                qsort(left, j)
        if i < right:
                qsort(i, right)

qsort(0, n - 1)
```

Хотя `QuickSort` является очень быстрой сортировкой в среднем, существуют тесты, на которых она работает очень долго. Оценивать время работы алгоритма будем числом сравнений с элементами массива (то есть, суммарным числом сравнений в первом и втором while). Требуется написать программу, генерирующую тест, на котором быстрая сортировка сделает наибольшее число таких сравнений. Задача на acmp.

- **Формат входного файла (input.txt).** В первой строке находится единственное число n ($1 \leq n \leq 10^6$).
- **Формат выходного файла (output.txt).** Вывести перестановку чисел от 1 до n, на которой быстрая сортировка выполнит максимальное число сравнений. Если таких перестановок несколько, вывести любую из них.
- Ограничение по времени. 2 сек.
- Ограничение по памяти. 256 мб.
- Пример:

input.txt	output.txt
3	1 3 2

- **Примечание.** На этой странице можно ввести ответ, выводимый Вашей программой, и страница посчитает число сравнений, выполняемых указанным выше алгоритмом Quicksort. Вычисления будут производиться в Вашем браузере. Очень большие массивы могут обрабатываться долго.

3 задача. Сортировка пугалом

«Сортировка пугалом» — это давно забытая народная потешка. Участнику под верхнюю одежду продевают деревянную палку, так что у него оказываются растопырены руки, как у огородного пугала. Перед ним ставятся n матрёшек в ряд. Из-за палки единственное, что он может сделать — это взять в руки две матрешки на расстоянии k друг от друга (то есть i-ую и $i + k$-ую), развернуться и поставить их обратно в ряд, таким образом поменяв их местами.

Задача участника — расположить матрёшки по неубыванию размера. Может ли он это сделать?

- **Формат входного файла (input.txt).** В первой строчке содержатся числа n и k ($1 \leq n, k \leq 10^5$) – число матрёшек и размах рук. Во второй строчке содержится n целых чисел, которые по модулю не превосходят 10^9 – размеры матрёшек.
- **Формат выходного файла (output.txt).** Выведите «ДА», если возможно отсортировать матрёшки по неубыванию размера, и «НЕТ» в противном случае.
- Ограничение по времени. 2 сек.
- Ограничение по памяти. 256 мб.
- Примеры:

input.txt	output.txt
3 2 2 1 3	НЕТ
5 3 1 5 3 4 1	ДА

4 задача. Точки и отрезки

Допустим, вы организовываете онлайн-лотерею. Для участия нужно сделать ставку на одно целое число. При этом у вас есть несколько интервалов последовательных целых чисел. В этом случае выигрыш участника пропорционален количеству интервалов, содержащих номер участника, минус количество интервалов, которые его не содержат. (В нашем случае для начала - подсчет только количества интервалов, содержащих номер участника). Вам нужен эффективный алгоритм для расчета выигрышей для всех участников. Наивный способ сделать это - просто просканировать для всех участников список всех интевалов. Однако ваша лотерея очень популярна: у вас тысячи участников и тысячи интервалов. По этой причине вы не можете позволить себе медленный наивный алгоритм.

- *Цель.* Вам дается набор точек и набор отрезков. Цель состоит в том, чтобы вычислить для каждой точки количество отрезков, содержащих эту точку.

- **Формат входного файла (input.txt).** Первая строка содержит два неотрицательных целых числа s и p. s - количество отрезков, p - количество точек. Следующие s строк содержат 2 целых числа a_i, b_i, которые определяют i-ый отрезок $[a_i, b_i]$. Последняя строка определяет p целых чисел - точек $x_1, x_2, ..., x_p$. Ограничения: $1 \leq s, p \leq 50000$; $-10^8 \leq a_i \leq b_i \leq 10^8$ для всех $0 \leq i < s$; $-10^8 \leq x_i \leq 10^8$ для всех $0 \leq j < p$.

- **Формат выходного файла (output.txt).** Выведите p неотрицательных целых чисел $k_0, k_1..., k_{p-1}$, где k_i - это число отрезков, которые содержат x_i. То есть,

$$k_i = |j : a_j \leq x_i \leq b_j|.$$

- Пример 1.

input.txt	output.txt
2 3 0 5 7 10 1 6 11	1 0 0

Здесь, у нас есть 2 отрезка и 2 точки. Первая точка принадлежит интервалу [0, 5], остальные точки не принадлежат ни одному из данных интервалов.

- Пример 2.

input.txt	output.txt
1 3 -10 10 -100 100 0	0 0 1

- Пример 3.

input.txt	output.txt
3 2 0 5 -3 2 7 10 1 6	2 0

5 задача. Индекс Хирша

Для заданного массива целых чисел citations, где каждое из этих чисел - число цитирований i-ой статьи ученого-исследователя, посчитайте индекс Хирша этого ученого.

По определению Индекса Хирша на Википедии: Учёный имеет индекс h, если h из его/её N_p статей цитируются как минимум h раз каждая, в то время как оставшиеся $(N_p - h)$ статей цитируются не более чем h раз каждая. Иными словами,

учёный с индексом h опубликовал как минимум h статей, на каждую из которых сослались как минимум h раз.

Если существует несколько возможных значений h, в качестве h-индекса принимается максимальное из них.

- **Формат ввода или входного файла (input.txt).** Одна строка citations, содержащая n целых чисел, по количеству статей ученого (длина citations), разделенных пробелом или запятой.
- **Формат выхода или выходного файла (output.txt).** Одно число - индекс Хирша (h-индекс).
- Ограничения: $1 \leq n \leq 5000$, $0 \leq citations[i] \leq 1000$.
- Пример.

input.txt	output.txt
3,0,6,1,5	3

  Пояснение. citations = [3,0,6,1,5] означает, что ученый опубликовал 5 статей в целом, и каждая из них оказалась процитирована 3, 0, 6, 1, 5 раз соответственно. Поскольку у ученого есть 3 статьи с минимум тремя цитированиями, а у оставшихся двух - не более 3 цитирований, его индекс Хирша равен 3.
- Пример.

input.txt	output.txt
1,3,1	1
- Ограничений по времени (и памяти) не предусмотрено, проверьте максимальный случай при заданных ограничениях на данные, и оцените асимптотическое время.
- Подумайте, если бы массив citations был бы изначально отсортирован по возрастанию, можно было бы еще ускорить алгоритм?

6 задача. Сортировка целых чисел

В этой задаче нужно будет отсортировать много неотрицательных целых чисел.

Вам даны два массива, A и B, содержащие соответственно n и m элементов. Числа, которые нужно будет отсортировать, имеют вид $A_i \cdot B_j$, где $1 \leq i \leq n$ и $1 \leq j \leq m$. Иными словами, каждый элемент первого массива нужно умножить на каждый элемент второго массива.

Пусть из этих чисел получится отсортированная последовательность C длиной $n \cdot m$. Выведите сумму каждого десятого элемента этой последовательности (то есть, $C_1 + C_{11} + C_{21} + ...$).

- **Формат входного файла (input.txt).** В первой строке содержатся числа n и m ($1 \leq n, m \leq 6000$) – размеры массивов. Во второй строке содержится

n чисел – элементы массива A. Аналогично, в третьей строке содержится m чисел — элементы массива B. Элементы массива неотрицательны и не превосходят 40000.

- **Формат выходного файла (output.txt).** Выведите одно число — сумму каждого десятого элемента последовательности, полученной сортировкой попарных произведенй элементов массивов A и B.
- Ограничение по времени. 2 сек.
- **Ограничение по времени распространяется на сортировку, без учета времени на перемножение. Подумайте, какая сортировка будет эффективнее, сравните на практике.**
- *Однако бытует мнение на OpenEdu, неделя 3, задача 2, что эту задачу можно решить на Python и уложиться в 2 секунды, включая в общее время перемножение двух массивов.*
- Ограничение по памяти. 512 мб.
- Пример:

input.txt	output.txt
4 4 7 1 4 9 2 7 8 11	51

- Пояснение к примеру. Неотсортированная последовательность C выглядит следующим образом:

  [14, 2, 8, 18, 49, 7, 28, 63, 56, 8, 32, 72, 77, 11, 44, 99].

  Отсортировав ее, получим:

  [**2,** 7, 8, 8, 11, 14, 18, 28, 32, 44, **49**, 56, 63, 72, 77, 99].

  Жирным выделены первый и одиннадцатый элементы последовательности, при этом двадцать первого элемента в ней нет. Их сумма — 51 — и будет ответом.

7 задача. Цифровая сортировка

Дано n строк, выведите их порядок после k фаз цифровой сортировки.

- **Формат входного файла (input.txt).** В первой строке входного файла содержатся числа n - число строк, m - их длина и k - число фаз цифровой сортировки ($1 \leq n \leq 10^6$, $1 \leq k \leq m \leq 10^6$, $n \cdot m \leq 5 \cdot 10^7$). Далее находится описание строк, но **в нетривиальном формате**. Так, i-ая строка ($1 \leq i \leq n$) записана в i-ых символах второй, ..., $(m+1)$-ой строк входного файла. Иными словами, строки написаны по вертикали. **Это сделано специально, чтобы сортировка занимала меньше времени.**

Строки состоят из строчных латинских букв: от символа "a"до символа "z"включительно. В таблице символов ASCII все эти буквы располагаются подряд и в алфавитном порядке, код буквы "a"равен 97, код буквы "z"равен 122.

- **Формат выходного файла (output.txt).** Выведите номера строк в том порядке, в котором они будут после k фаз цифровой сортировки.
- Ограничение по времени. 3 сек.
- Ограничение по памяти. 256 мб.
- Примеры:

input.txt	output.txt
3 3 1 bab bba baa	2 3 1
3 3 2 bab bba baa	3 2 1
3 3 3 bab bba baa	2 3 1

- **Примечание.** Во всех примерах входных данных даны следующие строки:
  - «bbb», имеющая индекс 1;
  - «aba», имеющая индекс 2;
  - «baa», имеющая индекс 3.

  Разберем первый пример. Первая фаза цифровой сортировки отсортирует строки по последнему символу, таким образом, первой строкой окажется «aba» (индекс 2), затем «baa» (индекс 3), затем «bbb» (индекс 1). Таким образом, ответ равен «2 3 1».

8 задача. K ближайших точек к началу координат

В этой задаче, ваша цель - найти K ближайших точек к началу координат среди данных n точек.

- Цель. Заданы n точек на поверхности, найти K точек, которые находятся ближе к началу координат (0, 0), т.е. имеют наименьшее расстояние до начала координат. Напомним, что расстояние между двумя точками (x_1, y_1) и (x_2, y_2) равно $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.

- **Формат ввода или входного файла (input.txt).** Первая строка содержит n - общее количество точек на плоскости и через пробел K - количество ближайший точек к началу координат, которые надо найти. Каждая следующая из n строк содержит 2 целых числа x_i, y_i, определяющие точку (x_i, y_i). Ограничения: $1 \leq n \leq 10^5$; $-10^9 \leq x_i, y_i \leq 10^9$ - целые числа.
- **Формат выхода или выходного файла (output.txt).** Выведите K ближайших точек к началу координат в строчку в квадратных скобках через запятую. Ответ вывести в порядке возрастания расстояния до начала координат. Если оно равно, порядок произвольный.
- Ограничение по времени. 10 сек.
- Ограничение по памяти. 256 мб.
- Пример 1.

input.txt	output.txt
2 **1** 1 3 -2 2	[-2,2]

- Пример 2.

input.txt	output.txt
3 **2** 3 3 5 -1 -2 4	[3,3],[-2,4]

9 задача. Ближайшие точки

В этой задаче, ваша цель - найти пару ближайших точек среди данных n точек (между собой). Это базовая задача вычислительной геометрии, которая находит применение в компьютерном зрении, систем управления трафиком.

- Цель. Заданы n точек на поверхности, найти наименьшее расстояние между двумя (разными) точками. Напомним, что расстояние между двумя точками (x_1, y_1) и (x_2, y_2) равно $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.
- **Формат ввода или входного файла (input.txt).** Первая строка содержит n - количество точек. Каждая следующая из n строк содержит 2 целых числа x_i, y_i, определяющие точку (x_i, y_i). Ограничения: $1 \leq n \leq 10^5$; $-10^9 \leq x_i, y_i \leq 10^9$ - целые числа.
- **Формат выхода или выходного файла (output.txt).** Выведите минимальное расстояние. Абсолютная погрешность между вашим ответом и оптимальным решением должна быть не более 10^{-3}. Чтобы это обеспечить, выведите ответ с 4 знаками после запятой.

- Ограничение по времени. 10 сек.
- Ограничение по памяти. 256 мб.
- Пример 1.

input.txt	output.txt
2 0 0 3 4	5.0

Здесь всего 2 точки, расстояние между ними равно 5.

- Пример 2.

input.txt	output.txt
4 7 7 1 100 4 8 7 7	0.0

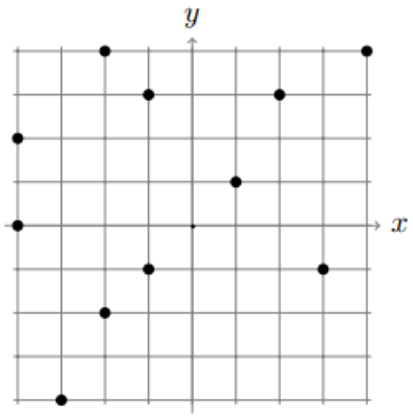
Здесь есть две точки, координаты которых совпадают, соотвественно, расстояние между ними равно 0.

- Пример 3.

input.txt	output.txt
11 4 4 -2 -2 -3 -4 -1 3 2 3 -4 0 1 1 -1 -1 3 -1 -4 2 -2 4	1.414213

Наименьшее расстояние - $\sqrt{2}$. В этом наборе есть 2 пары точек с таким расстоянием: (-1, -1) и (-2, -2); (-2,4) и (-1, 3).

! Цель - разработать $O(n \log n)$ алгоритм методом "Разделяй и властвуй". Более подробное описание задания и метода решения можно посмотреть в файле **for-lab4-9.pdf** в папке с ***заданиями к Лабораторным работам***.



10 задача ★. Дополнительно 3 несложных задания

1. Покажите, как выполнить сортировку n чисел, принадлежащих интервалу от 0 до $n^3 - 1$ за время $O(n)$.

2. Median-QuickSort. Еще один способ выбора центрального элемента заключается в том, чтобы вместо случайного элемента брать три элемента: первый, последний и один из середины массива, который надо разделить, - и в качестве центрального элемента использовать медиану всех трех элементов, то есть элемент со средним значением. Такой подход также прекрасно работает на практике, и вероятность низкой скорости работы крайне мала. Реализуйте быструю сортировку, используя описанный метод. Также можно выбирать любые три элемента массива, например, случайно.

   Сравните такой подход быстрой сортировки с обычной Quicksort и Randomized-QuickSort на разных наборах входных данных.

3. Карманная сортировка.

   (a) Используя *псевдокод* процедуры `BucketSort` из презентации к Лекции 3 (страница 28), напишите программу карманной сортировки на Python и проверьте ее, создав несколько рандомных массивов рациональных чисел x_i для i от 0 до n, $0 \leq x_i < 1$, n - длина массива.

   (b) Подумайте, как можно расширить случай для неотрицательных рациональных чисел с целой частью, например $0 \leq x_i < 10^3$.

   (c) И третье, попробуйте расширить случай для массива чисел с разными знаками.