

Преждевременная оптимизация — корень всех (или большинства) проблем в программировании.

Дональд Кнут

Алгоритмы и структуры данных–1

2024–2025 учебный год

SET 2. Домашняя работа

Асимптотический анализ. Рекуррентные соотношения.
Разделяй–и–властвуй

октябрь							
30	1	2	3	4	5	6	
7	8	9	10	11	12	13	
14	15	16	17	18	19	20	
21	22	23	24	25	26	27	
28	29	30	31				
пн	вт	ср	чт	пт	сб	вс	

Немного инструкций

Задания в рамках домашней работы SET 2 подразделяются на два блока:

1. *Блок А* «Аналитические задания» — задачи, связанные с разработкой и анализом асимптотических границ временной сложности рекурсивных алгоритмов.

Решения заданий Блока А оформляются в письменном виде в любом удобном формате (*L^AT_EX*, текстовый документ, скан, изображение и др.) и загружаются для оценки в соответствующую форму раздела SET 2. Домашняя работа на странице курса в LMS.

2. *Блок Р* «Задания на разработку» — задачи, связанные с реализацией алгоритмов по стратегии «разделяй и властвуй»

Решения заданий Блока Р загружаются в систему *CODEFORCES* и проходят автоматизированное тестирование. Для загрузки нужно перейти на <https://dsahse.contest.codeforces.com> и выбрать соответствующее соревнование. Доступ к соревнованию предоставлен по тем же учетным данным, что и к системе Яндекс.Контест.

Домашняя работа SET 2 содержит 4 обязательные задачи в Блоке А и 5 обязательных задач в блоке Р. Баллы, которые можно получить за решение задач, распределены следующим образом:

Блок А				Блок Р					
A1	A2	A3	A4	P1	P2	P3	P4	P5	P6b
9	7	7	9	7	8	9	7	10	10

Задачи, помеченные “b” не являются обязательными — баллы за их решение относятся к *бонусным*. Подтверждение решений бонусных задач сопровождается обязательной *устной защитой*.

Важные даты

1. Домашняя работа SET 2 открыта с **14:30 30 сентября 2024 г.**
2. Прием решений на оценку завершается в **00:00 14 октября 2024 г.**
3. Загруженное решение бонусной задачи P6b должно быть защищено до **25 октября 2024 г.**

Содержание

Задание A1.	Временная сложность рекурсии	1
Задание A2.	Применение мастер-теоремы	2
Задание A3.	Быстрее Штрассена!	3
Задание A4.	Значительные инверсии	4
Задание P1.	Ичиго и банкай Бъякуи	5
Задание P2.	Ичиго и первый древний навык	7
Задание P3.	Ичиго и второй древний навык	9
Задание P4.	Ичиго и поезда в замок Короля Душ	11
Задание P5.	Ичиго и двумерный переход в Уэко Мундо	13
Задание P6b.	Ичиго и трехмерный переход в Уэко Мундо	15

Успехов!

Задача А1. Временная сложность рекурсии

Ниже приведены два рекурсивных алгоритма обработки целочисленного массива A размера n .

```
1  algorithm1(A, n)
2      if n <= 20
3          return A[n]
4      x = algorithm1(A, n - 5)
5
6      for i = 1 to [n / 2]
7          for j = 1 to [n / 2]
8              A[i] = A[i] - A[j]
9      x = x + algorithm1(A, n - 8)
10
11     return x
```

```
1  algorithm2(A, n):
2      if n <= 50
3          return A[n]
4      x = algorithm2(A, [n / 4])
5
6      for i = 1 to [n / 3]
7          A[i] = A[n - i] - A[i]
8
9      x = x + algorithm2(A, [n / 4])
10
11     return x
```

1. 2 балла

Для каждого из представленных алгоритмов составить рекуррентное соотношение, которое выражает их временную сложность $T(n)$. Обратите внимание, что рекуррентное соотношение должно давать полное представление о сложности алгоритма, т.е., охватывать как рекурсивную, так и нерекурсивную ветку вычислений. Предполагается, что все арифметические операции выполняются за постоянное время.

2. 7 баллов

Вычислите асимптотическую точную границу $\Theta(f(n))$ временной сложности для каждого из представленных алгоритмов, если это возможно. В случае невозможности формирования асимптотической точной границы, представить отдельно верхнюю и нижнюю границы. Обоснуйте свой ответ с помощью метода подстановки, дерева рекурсии, или индукции.

Задача А2. Применение мастер-теоремы

Дан ряд рекуррентных соотношений, которые описывают временную сложность некоторых рекурсивных алгоритмов (при $n = 1$ во всех случаях принимаем $T(1) = 1$):

- $T(n) = 7 \cdot T\left(\frac{n}{3}\right) + n^2$.
- $T(n) = 4 \cdot T\left(\frac{n}{2}\right) + \log_2 n$.
- $T(n) = 0.5 \cdot T\left(\frac{n}{2}\right) + \frac{1}{n}$.
- $T(n) = 3 \cdot T\left(\frac{n}{3}\right) + \frac{n}{2}$.
- $T(n) = T(n-1) + T(n-2) + n \cdot \log_2 n$.

1. 4 балла

Для приведенных рекуррентных соотношений вычислите асимптотическую верхнюю границу временной сложности $O(g(n))$ с помощью основной теоремы о рекуррентных соотношениях (мастер-теоремы), если это возможно. Если применение мастер-теоремы невозможно, поясните причины.

2. 3 балла

Для рекуррентного(-ых) соотношения(-ий), не разрешимых с помощью мастер-теоремы, определите возможную асимптотическую верхнюю границу, используя метод итерации или метод подстановки.

Задача А3. Быстрее Штрассена!

Вы планируете разработать алгоритм **MULT**, предназначенный для умножения двух квадратных матриц A и B размерности $N \times N$ и асимптотически более эффективный, чем алгоритм Штрассена. Разрабатываемый алгоритм будет также использовать стратегию «разделяй-и-властвуй».

Исходные матрицы A и B разделяются на неизвестное количество фрагментов размера $N/4 \times N/4$ для дальнейшей рекурсивной обработки. Асимптотическая точная граница общих временных затрат на выполнение шагов *CONQUER* и *COMBINE* алгоритма **MULT** — $\Theta(N^2)$. Таким образом, времененная сложность алгоритма **MULT** будет описываться рекуррентным соотношением $T(N) = a \cdot T(N/4) + \Theta(N^2)$, где коэффициент a отвечает за количество решаемых подзадач — количество блоков-подматриц размерности $N/4 \times N/4$. Например, для алгоритма Штрассена в соответствии с рекуррентным соотношением $T(N) = 7 \cdot T(N/2) + \Theta(N^2)$ известно, что для каждой задачи решается 7 подзадач *вдвое меньшего* размера.

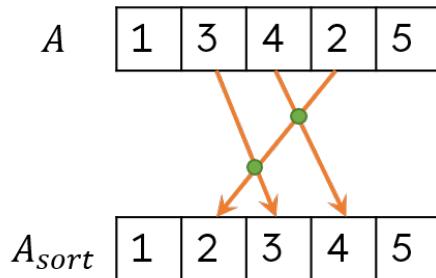
7 баллов

В каком диапазоне должен находиться параметр a разрабатываемого вами алгоритма **MULT** для того, чтобы в результате он был асимптотически *более эффективным* по временной сложности в сравнении с алгоритмом Штрассена? Обоснуйте свой ответ.

Задача А4. Значительные инверсии

Рассмотрим механизм подсчета так называемой «степени упорядоченности» некоторого целочисленного массива $A = [a_1, a_2, a_3, \dots, a_n]$, заполненного уникальными значениями.

Элементы a_i и a_j массива A назовем *инвертированными*, если $i < j$, но $a_i > a_j$. Например, в массиве $A = [1, 3, 4, 2, 5]$ достаточно выполнить две *инверсии*, а именно $3 \leftrightarrow 2$ и $4 \leftrightarrow 2$ (см. количество пересечений стрелок на рисунке ниже), чтобы получить отсортированный массив $A' = [1, 2, 3, 4, 5]$.



1. 6 балла

Разработайте DaC-алгоритм **CINV**, временная сложность которого должна соответствовать $O(n \cdot \log n)$, для подсчета степени упорядоченности массива путем вычисления количества необходимых перестановок. Описание алгоритма представьте в любом удобном формате. Опишите суть шагов *DIVIDE*, *CONQUER* и *COMBINE*, а также представьте рекуррентное соотношение для $T(n)$ и обоснуйте соответствие требуемой асимптотической верхней границе временной сложности. Проанализируйте, возвращает ли разработанный вами алгоритм **CINV** минимальное количество необходимых инверсий.

2. 3 балла

Элементы a_i и a_j массива A назовем *значительно инвертированными*, если $i < j$, но $a_i > 2 \cdot a_j$. Какие изменения и доработки необходимо внести в алгоритм **CINV**, разработанный на предыдущем шаге, чтобы в качестве степени упорядоченности велся подсчет количества пар значительно инвертированных элементов? Например, в массиве $A = [1, 3, 4, 2, 5]$ нет значительно переставленных элементов, а в массиве $A = [5, 3, 2, 4, 1]$ всего 4 пары значительно переставленных элементов: $5 \leftrightarrow 1$, $5 \leftrightarrow 2$, $4 \leftrightarrow 1$ и $3 \leftrightarrow 1$. Асимптотическая верхняя граница временной сложности измененного алгоритма должна остаться *неизменной*.

Задача Р1. Ичиго и банкай Бъякуи

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Наш герой, Ичиго Кurosаки, стал синигами почти случайно и вскоре схлестнулся в неравной битве с капитаном шестого отряда Готея 13 — Бъякуей Кучики, чтобы защитить его же сестру Рукию. Главное оружие синигами — это меч, который может принимать различные формы в зависимости от уровня духовной силы его владельца. Высшая форма меча именуются «банкай», в котором меч Бъякуи распадается на несчетное количество осколков разной длины.



Каждый осколок определяется интервалом $[a, b]$ — упорядоченным множеством целых чисел от a до b . Например, интервалу $[16, 23]$ соответствует множество $\{16, 17, 18, 19, 20, 21, 22, 23\}$.

Длиной интервала назовем количество элементов в соответствующем множестве.

Перекрытием двух интервалов $[a, b]$ и $[c, d]$ назовем другой интервал, содержащий результат пересечения множеств, соответствующих исходным интервалам. Например, перекрытию интервалов $[1, 5]$ и $[3, 6]$ соответствует интервал $[3, 5]$, длина которого равна 3.

Ичиго быстро понял, что для решающего удара ему нужно найти самое длинное перекрытие пары осколков.

Помогите Ичиго реализовать алгоритм по стратегии «разделяй-и-властвуй», который получает на вход конечный список из n заданных интервалов $[a_1, b_1], [a_2, b_2], \dots, [a_n, b_n]$ и выдает в ответ длину самого длинного перекрытия между парой интервалов среди заданных, а также интервал, соответствующий самому длинному перекрытию.

Представление интервала $[a, b]$ должно быть реализовано в виде самостоятельной структуры данных с методами для вычисления его длины и поиска перекрытия с другим интервалом:

```
struct Interval {  
    int left;  
    int right;  
  
    size_t length();  
    Interval overlap(const Interval& other);  
}
```

Реализация других методов и полей этой структуры остается на ваше усмотрение. Использование библиотечных методов для сортировки, поиска и других действий **не допускается**.

Формат входных данных

Первая строка содержит число n ($0 \leq n \leq 10^5$) — количество интервалов.

Следующие n строк содержат по два числа a, b — границы интервала.

Формат выходных данных

В первой строке выводится длина максимального перекрытия среди заданных интервалов.

Если длина максимального перекрытия больше 0, то в следующей строке через пробел выводятся два числа x и y — границы интервала, соответствующего максимальному перекрытию. Если таких интервалов несколько, то выводится тот, у которого *наименьшие* границы.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	—	тесты из условия	—	полная
1	0.5	небольшие тесты	0	первая ошибка
2	0.5	$n \leq 100$	0-1	первая ошибка
3	1	$n \leq 10^3$	0-2	первая ошибка
4	1	$n \leq 10^4$	0-3	первая ошибка
5	1	$n \leq 2 \cdot 10^4$	0-4	первая ошибка
6	1	$n \leq 5 \cdot 10^4$	0-5	первая ошибка
7	2	—	0-6	первая ошибка

Примеры

стандартный ввод	стандартный вывод
3 1 2 1 5 2 4	3 2 4
2 -3 5 -5 -4	0

Задача Р2. Ичиго и первый древний навык

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Становление синигами не может происходить без овладения древними математическими навыками, первым из которых является умножение сколь угодно больших целых чисел. Ичиго вновь обратился к вам за помощью в разработке «разделяй-и-властвуй» алгоритма решения этой задачи.



Формат входных данных

На вход подается две строки, в каждой из которых располагается одно неотрицательное число. Каждое из чисел не превышает $10^{5 \cdot 10^4}$.

Формат выходных данных

Программе необходимо вывести результат перемножения введенных чисел.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
1	—	тесты из условия	—	полная
2	0.5	$N \leq 3 \cdot 10^9$	1	первая ошибка
3	0.5	$N \leq 10^{100}$	1-2	первая ошибка
4	0.5	$N \leq 10^{500}$	1-3	первая ошибка
5	0.5	$N \leq 10^{1000}$	1-4	первая ошибка
6	1	$N \leq 10^{10000}$	1-5	первая ошибка
7	2	$N \leq 10^{20000}$	1-6	первая ошибка
8	3	$N \leq 10^{50000}$	1-7	первая ошибка

Примеры

стандартный ввод	стандартный вывод
10 4	40
1234 4321	5332114

Замечание

Обратите внимание, что вам необходимо самостоятельно определить границу перехода от рекурсивного умножения к обычному алгоритму умножения целых чисел «в столбик».

Задача Р3. Ичиго и второй древний навык

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

После овладения первым древним навыком перемножения сколь угодно больших целых чисел Ичиго должен получить второй древний математический навык — перемножение квадратных матриц сколь угодно большой размерности. Как и всегда, нашему герою требуется ваша помощь в разработке «разделяй-и-властвуй» алгоритма для решения этой задачи.



Формат входных данных

В первой строке входных данных даётся число $n(1 \leq n \leq 1024)$ — размер матрицы. Гарантируется, что $n = 2^k$.

В следующих n строках даётся по n чисел $a_{i,j}(-10^9 \leq a_{i,j} \leq 10^9)$ — элементы матрицы A .

В следующих n строках даётся по n чисел $b_{i,j}(-10^9 \leq b_{i,j} \leq 10^9)$ — элементы матрицы B .

Формат выходных данных

Выполните по n строк, в каждой из которых по n чисел $c_{i,j}$ — элементы матрицы произведения.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
1	—	тесты из условия	—	полная
2	0.5	$n \leq 4$	1	первая ошибка
3	0.5	$n \leq 8$	1-2	первая ошибка
4	1	$n \leq 32$	1-3	первая ошибка
5	1	$n \leq 64$	1-4, 6	первая ошибка
6	1	$n \leq 128$	1-2	первая ошибка
7	1	$n \leq 256$	1-6	первая ошибка
8	2	$n \leq 512$	1-7	первая ошибка
9	2	$n \leq 1024$	1-8	первая ошибка

Пример

стандартный ввод	стандартный вывод
2	2 2
1 0	2 2
0 1	
2 2	
2 2	

Замечание

Обратите внимание, что вам необходимо самостоятельно определить границу перехода от рекурсивного умножения к обычному алгоритму умножения квадратных матриц.

Задача Р4. Ичиго и поезда в замок Короля Душ

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Главнокомандующий Готея 13, капитан первого отряда синигами Генрюусай Шигекуни Ямamoto и другие его члены должны часто посещать замок Короля Душ. В качестве самого надежного способом передвижения из Сейрейтея в замок Короля были выбраны воздушные поезда, но в связи с участвившимся числом аварий на железнодорожной ветке Сейрейтея–замок Короля, Генрюусай Шигекуни Ямamoto и члены первого отряда решили изменить график движения поездов.



Тщательный анализ состояния воздушного железнодорожного полотна показал, что оптимальным является следующий график движения поездов с учетом остановок на станциях: сначала поезд идет на протяжении T_1 минут со скоростью V_1 метров в минуту, затем T_2 минут со скоростью V_2 метров в минуту, ..., наконец, T_N минут со скоростью V_N метров в минуту. В течение некоторых интервалов поезд может выполнять остановку (скорость равна 0).

По действующей инструкции обеспечения безопасности движения поездов расстояние между локомотивами двух следующих друг за другом поездов должно быть не менее L метров. Генрюусай Шигекуни Ямamoto, узнав о невиданных успехах Ичиго в овладении двумя древними математическими навыками, обратился к нему. Помогите Ичиго определить минимально допустимый интервал в минутах между отправлениями поездов, позволяющий им двигаться по этому графику без опасного сближения.

Формат входных данных

В первых двух строках входного файла содержится два натуральных числа, задающие минимально допустимое расстояние L и количество участков пути N ($100 \leq L \leq 10\,000$, $1 \leq N \leq 1000$). Далее следует N пар целых чисел T_i и V_i , задающих график движения поездов ($1 \leq T_i \leq 1000$, $0 \leq V_i \leq 1000$).

Формат выходных данных

В выходной файл необходимо вывести искомый интервал между отправлениями поездов в минутах, не менее чем с тремя верными знаками после десятичной точки.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	—	тесты из условия	—	полная
1	1	$N \leq 300$	0	первая ошибка
2	2	$N \leq 1\,000$	0-1	первая ошибка
3	2	$N \leq 3\,000$	0-2	первая ошибка
4	2	$N \leq 10\,000$	0-3	первая ошибка

Пример

стандартный ввод	стандартный вывод
1000 4 10 0 30 80 15 0 20 100	27.500

Задача Р5. Ичиго и двумерный переход в Уэко Мундо

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Синигами существуют для того, чтобы поддерживать баланс между Миром Людей и Обществом Душ, однако между этими мирами существует еще одно измерение — Уэко Мундо, населенное «пустыми», которые являются полной противоположностью людей. Пустые могут проникать в Мир Людей и Общество Душ, а их целью является получение абсолютной власти над обоими мирами.



Чтобы Ичиго Курасаки справиться с пустыми, ему необходимо совершить безопасный переход в Уэко Мундо через разломы в небе Общества Душ, которые расположены в строго определенных местах. Помогите ему найти этот разлом, вычислив с помощью «разделяй-и-властвуй» алгоритма минимальное расстояние между парой точек среди некоторого конечного множества точек, которые заданы своими целочисленными координатами в двумерном пространстве.

Специальных требований к представлению и хранению информации о точке в виде самостоятельной структуры данных не предъявляется.

Вам разрешается использовать встроенную сортировку std::sort.

Формат входных данных

На вход подается n ($2 \leq n \leq 10^6$) строк.

Каждая из которых содержит два целых числа x ($0 \leq |x| \leq 10^9$) и y ($0 \leq |y| \leq 10^9$), разделенных пробелами, — координаты точки.

Формат выходных данных

Пусть минимальное расстояние между парой точек среди заданных равно d . В качестве ответа на задачу выведите целую часть d .

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	—	тесты из условия	—	полная
1	1	небольшие тесты	0	первая ошибка
2	1	$n \leq 100$	0-1	первая ошибка
3	1	$n \leq 10^3$	0-2	первая ошибка
4	2	$n \leq 10^4$	0-3	первая ошибка
5	5	—	0-4	первая ошибка

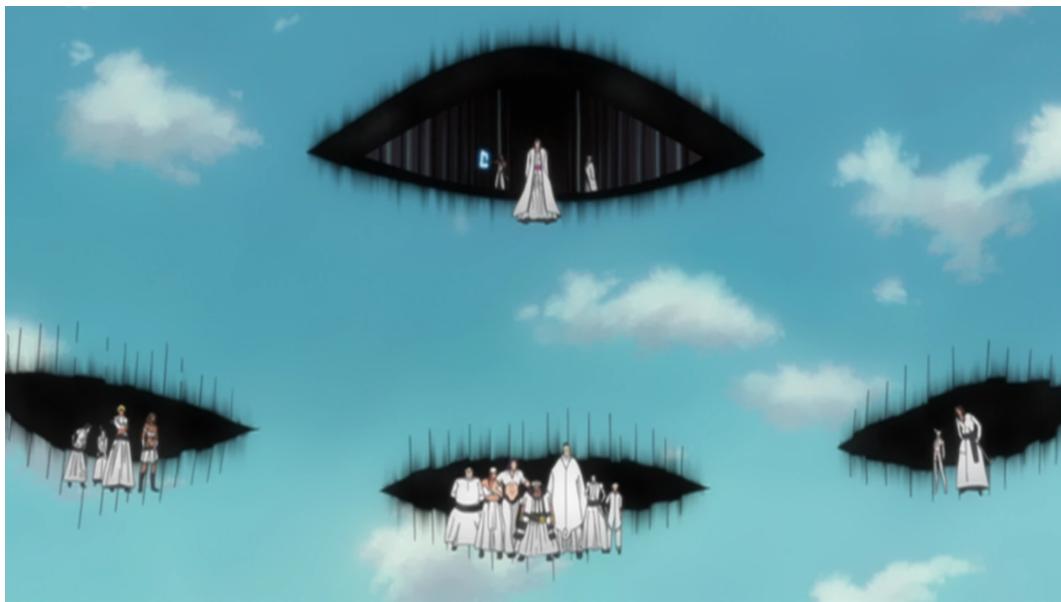
Примеры

стандартный ввод	стандартный вывод
0 1 0 0	1
0 1 0 0 0 4	1
0 0 5 5 0 10 10 0	7

Задача Рб. Ичиго и трехмерный переход в Уэко Мундо

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Пустые могут беспрепятственно усложнять переход в Уэко Мундо, меняя размерность пространства, в котором находится разлом. Помогите Ичиго усовершенствовать ранее разработанный «разделяй-и-властвуй» алгоритм и найти минимальное расстояние между парой точек в небе, заданных координатами в *трехмерном* пространстве.



Специальных требований к представлению и хранению информации о точке в виде самостоятельной структуры данных не предъявляется.

Вам разрешается использовать встроенную сортировку std::sort.

Формат входных данных

На вход подается n ($2 \leq n \leq 5 \cdot 10^4$) строк.

Каждая из которых содержит три целых числа x ($0 \leq |x| \leq 10^6$) и y ($0 \leq |y| \leq 10^6$), разделенных пробелами, – координаты точки.

Гарантируется, что все точки различны.

Формат выходных данных

В первой строке выведите единственное вещественное число d – минимальное расстояние – с точностью не менее 5 знаков после запятой. Во второй строке выведите пару целых чисел – номера точек, расстояние между которыми совпадает с ответом. Если таких пар несколько, выведите любую пару.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	–	тесты из условия	–	полная
1	1	$n \leq 100$	0	первая ошибка
2	2	$n \leq 10^4$	0-1	первая ошибка
3	2	$n \leq 3 \cdot 10^4$	0-2	первая ошибка
4	5	–	0-3	первая ошибка

Примеры

стандартный ввод	стандартный вывод
5 1 1 0 1 0 1 0 1 1 0 0 0 2 2 2	1.4142135624 4 3
2 2 1 2 4 2 4	3.0000000000 2 1