Задача 1. Количество символов

Источник:	базовая
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Для заданного тестового файла посчитать, сколько раз каждый символ встречается в этом файле.

Формат входных данных

Во входном файле записан некоторый текст. Размер файла не превосходит 1 Мб.

Формат выходных данных

В выходной файл необходимо вывести информацию по каждому символу, который встре-чается во входном файле, в следующем формате:

`<код символа> : <изображение символа> - <количество>`.

Информацию для каждого символа выводить на отдельной строке в порядке возрастания кодов. Начинать с кода, большего 12

Пример

input.txt	output.txt
to be or not to be that is the question	32 : - 9 97 : a - 1 98 : b - 2 101 : e - 4 104 : h - 2 105 : i - 2 110 : n - 2 111 : o - 5 113 : q - 1 114 : r - 1 115 : s - 2 116 : t - 7 117 : u - 1

Задача 2. Количество слов

Источник:	базовая
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Для заданного тестового файла посчитать распределение слов по их длинам, т.е. сколько раз слово определенной длины встречается в этом файле. Словом считается любая подпоследова-тельность рядом стоящих символов в тексте, ограниченная пробелом, концом строки или концом файла, не содержащая пробелов и символов конца строки

Формат входных данных

Во входном файле записан некоторый текст. Размер файла не превосходит 1 Мб.

Формат выходных данных

В выходной файл необходимо вывести информацию о длинах слов в следующем формате:
`<длина слова> - <количество слов этой длины>`
Информацию выводить в порядке возрастания длин, каждую длину на отдельной строке.

Пример

input.txt	output.txt
to be or not to be that is the question	2 - 6 3 - 2 4 - 1 8 - 1

Задача 3. Количество строк

Источник:	базовая
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

В заданном тестовом файле посчитать количество строк.

Формат входных данных

Во входном файле записан некоторый текст. Размер файла не превосходит 1 Мб.

Формат выходных данных

В выходной файл необходимо вывести одно целое число – количество строк во входном файле.

Пример

input.txt	output.txt
The state finals of the Texas Computer Education Association Computer Programming Contest is to-day. Teams from all over the state of Texas are participating in the event. In last year's contest, each team brought their own computer.	5

Задача 4. Гистограмма

Источник:	основная*
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Во входном файле содержится некоторый текст. Вам необходимо построить гистограмму встречаемости различных символов в тексте.

Формат входных данных

Входной файл содержит просто текст. Текст состоит только из ASCII-символов с кодами от 0 до 126.

Размер текста не превосходит 100 000 байт.

Формат выходных данных

Для каждого печатаемого символа (ASCII код от 32 до 126 включительно), встретившегося в тексте хотя бы раз, выведите сам символ и через пробел выведите столько символов '#', сколько раз данный символ встретился в тексте. Символы выводить в порядке увеличения их кода.

Пример

input.txt	output.txt

`This is a text.`
`Multiline text.`
	`  ####`
	`. ##`
	`M #`
	`T #`
	`a #`
	`e ###`
	`h #`
	`i ####`
	`l ##`
	`n #`
	`s ##`
	`t #####`
	`u #`
	`x ##`

Комментарий

Первый символ в примере вывода — пробел.

Для чтения данных можно использовать посимвольных ввод с помощью `getchar` или построчный с помощью `gets`.

Задача 5. A + B

Источник:	базовая
Имя входного файла:	`input.bin`
Имя выходного файла:	`output.bin`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Во входном файле дано восемь байт, которые задают два 32-битных знаковых целых числа: в первых четырёх байтах записано число A, а в последних четырёх — число B. Нужно вычислить полусумму этих чисел, округляя полученный результат **вниз**. Полученную полусумму вывести в выходной файл как 32-битное знаковое целое число. Все три числа заданы с порядком байтов little-endian.

Внимание: В качестве A и B могут быть даны любые числа из диапазона знаковых 32-битных целых чисел (т.е. от `INT_MIN` до `INT_MAX`). Настоятельно рекомендуется протестировать решение на числах, близких к крайним значениям, а также на разных комбинациях чётности чисел. Возможно, будет удобнее использовать 64-битные целые для промежуточных результатов, чтобы избежать переполнения.

Весь ввод и вывод в данной задаче бинарный. Ниже показано hex-представление бинарных данных: каждая группа из двух цифр обозначает один байт в файле. Вашей программе на вход будет подан файл с 8 байтами данных, и программа должна создать файл с 4 байтами ответа. Для создания, редактирования и просмотра бинарных файлов используйте какой-нибудь Hex-редактор, например HxD. Пример входных и выходных данных в бинарном виде можно скачать по ссылке.

Примеры

input.bin	output.bin
`AB 05 00 00 12 30 00 00`	`DE 1A 00 00`
`FF FF FF FF FE FF FF FF`	`FE FF FF FF`
`0A 0D 0A 0D 0D 0A 0D 0A`	`8B 8B 8B 0B`
`00 00 00 80 00 00 00 80`	`00 00 00 80`

Пояснение к примеру

В первом примере даны числа $A = 1451$ и $B = 12306$. Сумма равна 13757, после деления на два получаем 6878.

Во втором примере даны числа $A = -1$ и $B = -2$. Сумма равна -3, при делении на два получается -2 (округление вниз).

В третьем примере даны большие положительные числа. Если не работает, убедитесь, что открываете файлы в бинарном режиме.

В четвертом примере числа одинаковы и равны `INT_MIN` $= -2^{31}$. Очевидно, полусумма также равна `INT_MIN`.

Задача 6. Сумма чисел

Источник:	базовая
Имя входного файла:	`input.bin`
Имя выходного файла:	`output.bin`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

В первых четырёх байтах входного файла задано число N — количество чисел в последовательности ($1 \leqslant N \leqslant 10\,000$). Далее заданы сами целые числа последовательности: N штук по 4 байта в каждом. Все числа по абсолютной величине не превышают 10^5.

Требуется найти сумму N чисел последовательности и вывести её как 4-байтовое целое число.

Внимание: Весь ввод и вывод в данной задаче бинарный. В каждом конкретном тесте все числа записаны с одинаковым порядком байтов: это может быть либо **big-endian**, либо **little-endian**. Однако в разных тестах порядок байтов может быть разным. Выводить число в ответ требуется с тем же порядком байтов, с которым заданы входные данные.

Пример

input.bin
00 00 00 05 00 00 02 A7 00 00 00 A0 00 00 03 CD 00 00 00 78 00 00 01 B8
output.bin
00 00 09 44

Пояснение к примеру

Учтите, что в примере указано лишь hex-представление бинарных данных! Вашей программе на вход будет подан файл с 24 байтами данных, и программа должна создать файл с 4 байтами ответа.

Задача 7. Слияние последовательностей

Источник:	основная*
Имя входного файла:	`input.bin`
Имя выходного файла:	`output.bin`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

В первых четырёх байтах входного файла задано целое число N — количество чисел в первой последовательности, а в следующих четырёх байтах задано целое число M — количество чисел во второй последовательности. Далее идут N четырёхбатовых целых чисел первой последовательности, и затем M чисел второй последовательности. Все числа знаковые, каждая последовательность упорядочена по неубыванию. Длины последовательностей лежат в диапазоне: $1 \leqslant N, M \leqslant 10^6$.

Требуется реализовать функцию слияния двух отсортированных последовательностей с сигнатурой:

```
//merges sorted arrays a[0..ak-1] and b[0..bk-1] into
//one sorted array res[0..rk-1], returning rk from function
int merge(const int *a, int ak, const int *b, int bk, int *res);
```

и применить её к заданным в файле последовательностям.

Требуется вывести в выходной файл ровно $N + M$ четырёхбайтовых целых чисел: полученная в результате слияния упорядоченная последовательность.

Пример

input.bin
05 00 00 00 04 00 00 00 FC FF FF FF FD FF FF FF 01 00 00 00 01 00 00 00 0A 00 00 00 F9 FF FF FF 00 00 00 00 07 00 00 00 08 00 00 00
output.bin
F9 FF FF FF FC FF FF FF FD FF FF FF 00 00 00 00 01 00 00 00 01 00 00 00 07 00 00 00 08 00 00 00 0A 00 00 00

Задача 8. Разбиение массива

Источник:	основная
Имя входного файла:	`input.bin`
Имя выходного файла:	`output.bin`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

В первых четырёх байтах входного файла задано целое число N — количество чисел в массиве. В следующих четырёх байтах записано целое число p — пивот-элемент. Далее идут N четырёхбатовых целых чисел — содержимое массива. Все числа знаковые, длина последовательности лежит в диапазоне: $1 \leqslant N \leqslant 10^6$.

Требуется реализовать функцию разбиения массива относительно заданного пивот-элемента с сигнатурой:

```
//partitions array a[0..n-1] into two subarrays, returning value k
// the subarray a[0..k-1] must have all elements <= pivot
// the subarray a[k..n-1] must have all elements >= pivot
int partition(int *a, int n, int pivot);
```

и применить её к заданной в файле последовательности. Внутри функции разрешается использовать дополнительную память.

Важно: Заметим, что элементы, которые в точности равны `pivot`, можно помещать как в левую, так и в правую часть массива. В данной задаче требуется распределить эти элементы примерно поровну. Если в левую часть попадает u элементов, равных пивоту, а в правую часть — v элементов, то должно выполняться: $|u - v| \leqslant 1$.

В первые 4 байта выходного файла нужно вывести целое число `k` — сколько элементов попадает в левую часть массива. Далее нужно вывести N четырёхбайтовых целых чисел: содержимое массива `a` после выполнения функции `partition`.

Пример

input.bin
09 00 00 00 04 00 00 00 06 00 00 00 F8 FF FF FF 09 00 00 00 F8 FF FF FF FA FF FF FF 05 00 00 00 02 00 00 00 09 00 00 00 FF FF FF FF
output.bin
05 00 00 00 F8 FF FF FF F8 FF FF FF FA FF FF FF 02 00 00 00 FF FF FF FF 05 00 00 00 06 00 00 00 09 00 00 00 09 00 00 00

Задача 9. Сортировка слиянием

Источник:	повышенной сложности
Имя входного файла:	`input.bin`
Имя выходного файла:	`output.bin`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

В первых четырёх байтах входного файла задано целое число N — количество чисел в массиве A. Далее идут N четырёхбайтовых целых чисел — содержимое массива A. Размер массива лежит в диапазоне: $0 \leqslant N \leqslant 500\,000$.

Требуется отсортировать массив A по неубыванию, используя **алгоритм сортировки слиянием**.

В выходной файл нужно вывести ровно N четырёхбайтовых целых чисел: содержимое массива A после сортировки.

Пример

input.bin
0A 00 00 00 1F 00 00 00 F2 FF FF FF 06 00 00 00 04 00 00 00 26 00 00 00 FD FF FF FF 1E 00 00 00 F6 FF FF FF 0A 00 00 00 F4 FF FF FF
output.bin
F2 FF FF FF F4 FF FF FF F6 FF FF FF FD FF FF FF 04 00 00 00 06 00 00 00 0A 00 00 00 1E 00 00 00 1F 00 00 00 26 00 00 00

Задача 10. Быстрая сортировка+

Источник:	повышенной сложности
Имя входного файла:	`input.bin`
Имя выходного файла:	`output.bin`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

В первых четырёх байтах входного файла задано целое число N — количество чисел в массиве A. Далее идут N четырёхбайтовых целых чисел — содержимое массива A. Размер массива лежит в диапазоне: $0 \leqslant N \leqslant 500\,000$.

Требуется отсортировать массив A по неубыванию, используя **алгоритм быстрой сортировки**.

В выходной файл нужно вывести ровно N четырёхбайтовых целых чисел: содержимое массива A после сортировки.

Пример

input.bin
0A 00 00 00 1F 00 00 00 F2 FF FF FF 06 00 00 00 04 00 00 00 26 00 00 00 FD FF FF FF 1E 00 00 00 F6 FF FF FF 0A 00 00 00 F4 FF FF FF
output.bin
F2 FF FF FF F4 FF FF FF F6 FF FF FF FD FF FF FF 04 00 00 00 06 00 00 00 0A 00 00 00 1E 00 00 00 1F 00 00 00 26 00 00 00