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. Арифметическая прогрессия

Источник:	основная
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	разумное

Задана последовательность натуральных чисел из диапазона $[1, 2147483647]$. Количество чисел в этой последовательности не превышает 100000. Необходимо определить, можно ли выстроить эти числа в отрезок арифметической прогрессии. При необходимости порядок чисел в последовательности можно изменять. Требуется написать программу для решения вышеназванной задачи.

Формат входных данных

Входной файл содержит заданную последовательность натуральных чисел. Числа в файле разделены пробелами или символами перехода на новую строку.

Формат выходных данных

Выходной файл должен содержать либо шаг прогрессии в случае положительного ответа, либо строку `NO` в противоположном случае.

Пример

input.txt	output.txt
80 50 10 30 70 40 20 60 90	10

Задача 5. Разворот файла - 1

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

В данной задаче вам дан файл, строки которого требуется вывести в обратном порядке.

Формат входных данных

Входной файл состоит не более, чем из 1000 строк. Каждая строка состоит не более, чем из 1000 символов.

Формат выходных данных

В выходной файл нужно вывести строки входного файла в обратном порядке.

Пример

input.txt	output.txt
string one string two	string two string one

Задача 6. Разворот файла - 2

Источник:	основная
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	разумное

В данной задаче вам дан файл, строки которого требуется вывести в обратном порядке.

Формат входных данных

Размер входного файла не превышает $3 \dot{1} 0^6$ байт.

Формат выходных данных

В выходной файл нужно вывести строки входного файла в обратном порядке.

Пример

input.txt	output.txt
string one string two	string two string one

Задача 7. 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 8. Таблица

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

Есть набор видео, у каждого видео есть свой идентификатор (ID) — целое число в диапазоне от 0 до 1 000. Некоторые из этих видео могут быть разбиты на фрагменты, остальные видео заданы одним фрагментом.

Во входном файле заданы все фрагменты, для каждого из них указан идентификатор видео и длительность фрагмента в секундах. Нужно вывести статистику по каждому видео в виде тройки: ID, количество фрагментов, суммарная длительность. Все тройки нужно записать в красиво отформатированную таблицу.

Формат таблицы виден в примере выходного файла. В каждой ячейке таблицы число выровнено по правому краю ячейки. Слева и справа от ячейки стоят специальные пробелы. Ширину всех ячеек в каждом столбце таблицы нужно выбрать минимально возможной с учётом этих условий.

Формат входных данных

В первой строке задано число N — суммарное количество фрагментов ($1 \leqslant N \leqslant 10^4$).

В каждой из оставшихся N строк указано по два целых числа: ID видео, в которое входит фрагмент, и длительность фрагмента. Все длительности целые, неотрицательные, не превышают 10^5.

Формат выходных данных

Выведите информацию о каждом видео в строку таблицу. Первый столбец — это ID видео, второй — количество его фрагментов, а третий — суммарная длительность. Видео должны быть перечислены в таблице в порядке увеличения ID.

Пример

input.txt	output.txt
5 37 5 1 17 313 2378 1 79 1 35	+-----+---+------+ | 1 | 3 | 131 | +-----+---+------+ | 37 | 1 | 5 | +-----+---+------+ | 313 | 1 | 2378 | +-----+---+------+

Пояснение к примеру

Здесь задано три видео, и только одно из них (ID = 1) разбито на фрагменты. У него три фрагмента, и если просуммировать их длительность, то получается 131 секунда.

Задача 9. Комментарии

Источник:	повышенной сложности
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	**специальное**

Дан текст, нужно удалить из него комментарии согласно правилам языка C (точнее C99). Есть два вида комментариев:

- *Блочный комментарий*: начинается с `/*`, заканчивается `*/`, может занимать несколько строк.
- *Строчный комментарий*: начинается с `//` и заканчивается символом перевода строки.

При удалении комментария все символы перевода строки остаются в файле, даже если они находятся внутри блочного комментария. В конце текста может остаться открытый комментарий. В данной задаче **все пробелы и переводы строк в выходном файле должны быть выведены точно**.

Текст непустой, имеет длину до миллиона символов. Объём используемой вашей программой памяти должен быть много меньше мегабайта. В частности, сохранять в памяти программы всё содержимое входного файла **нельзя**.

В тексте могут быть следующие символы:

- маленькие и большие латинские буквы,
- цифры,
- пробелы и переводы строк,
- символы `/` и `*`,
- дополнительные символы: круглые и фигурные скобки, запятая и точка с запятой, знаки плюс и минус.

Пример

input.txt	output.txt

`/*C-style comments can contain`
`multiple lines*/ /*or just one*/`
`// C++-style comment lines`
`int main() {`
`  // The below code wont be run`
`  //return 1;`
`  return 0; //this will be run`
`}`
	`int main() {`
	
	`  return 0;`
	`}`

Комментарий

Рекомендуется читать текст по символам. Для проверки, закончились ли символы в файле, можно сравнивать возвращаемое значение `scanf` с единицей:

```
while (1) {
    if (scanf("%c", &curr) != 1)
        break;
    ... //читаем и обрабатываем очередной символ
}
```

Задача 10. Реальные логи

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

В данной задаче во входной файл подаётся "как есть" лог проигрывания видео-файлов из настоящей игры TheDarkMod. Нужно прочитать логи и вывести некоторую простую информацию о каждом видео.

В логах записано множество сообщений, поступающих из кода, который в реальном времени декодирует видео и выдаёт наружу готовые для отрисовки кадры. Каждое сообщение начинается с указания момента времени, когда оно произошло (timestamp), который записан в формате `A.BBB.CCC`, где A, B и C — количество секунд, миллисекунд и микросекунд соответственно.

Когда какой-либо видео-файл начинает проигрываться, в логи пишется сообщение "`Decoded first frame`", а когда заканчивает проигрываться, тогда пишется сообщение "`Video ended: no more frames`". Время, прошедшее от начала проигрывания до конца будем считать реальным временем показа видео (его надо выводить в ответ). Известно, что в любой момент времени проигрывается не больше одного видео-файла одновременно.

В процессе проигрывания видео-файла программа декодирует кадры из него. Для декодирования каждого кадра сначала вызывается функция распаковки кадра из библиотеки FFmpeg (сообщение "`Packet decoded`"), а потом распакованный кадр переводится в цветовое пространство RGB (сообщение "`Converted to RGBA`"). У каждого из этих двух сообщений подписано, сколько времени заняла соответствующая процедура в миллисекундах, Сумму этих двух значений будем считать временем декодирования одного конкретного кадра.

Нужно собрать простую статистику по тому, как долго декодировались кадры каждого отдельного видео. Нужно посчитать:

- сколько всего кадров было декодировано,
- сколько в сумме времени затрачено на декодирование кадров,
- минимальное, максимальное и среднее время декодирования кадра.

В выходной файл нужно вывести эти результаты в формате, аналогичном показанному в примере. Если в логе проигрывалось несколько видео-файлов, то нужно указать статистику для каждого из них, перечисляя их в том порядке, в котором они проигрывались.

Формат входных данных

Внимание: ниже в поле входного файла указана лишь ссылка: по ней можно скачать пример входного файла. А вот в поле выходного файла указан собственно вывод, который должна получить ваша программа.

В данной задаче кроме примера есть ещё лишь два теста. Они очень похожи на пример (записаны таким же образом).

Пример

input.txt

```
www.ledas.com/~stgatilov/fitprog/tdmlog_sample.in
```

output.txt

```
Video "video/ffmpeg_test/tearsofsteel_avi_mpeg4_mp3.avi":
  Total frames: 1007
  Total decode time: 257.397 ms
  Actual playback time: 41934.506 ms
  Min/avg/max decode time: 0.233/0.256/0.584 ms

Video "video/ffmpeg_test/sykes_roq_roq.roq":
  Total frames: 354
  Total decode time: 171.307 ms
  Actual playback time: 11788.245 ms
  Min/avg/max decode time: 0.419/0.484/0.719 ms
```