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. Стилистическая ценность

Источник:	базовая
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

В связи с внезапно образовавшимся дефицитом IT-кадров охранник Вася был назначен на должность project manager.

В Интернете Вася нашёл рассказ про то, насколько важно делать отступы при написании кода. После чего он придумал новый параметр оценки кода подчинённых — *стилистическую ценность*.

Стилистическая ценность кода — это сумма по всем строкам числа пробелов перед первым символом в строке с кодом, большим 32 (если такого символа в строке нет, то количество пробелов в сумму не входит).

На вход Вам подаётся код. Ваша задача — написать программу, которая вычисляет стилистическую ценность этого кода.

Формат входных данных

На вход даётся текстовый файл, состоящий из символов с кодами от 32 до 126 включительно, а также из символов перевода строки. При этом возможны как строки, заканчивающиеся некоторым количеством пробелов (или состоящие только из пробелов), так и пустые строки в любом месте текста, в том числе и в его конце. Длина файла не превосходит 100K.

Формат выходных данных

В выходной файл необходимо вывести одно целое число — стилистическую ценность кода из входного файла.

Пример

input.txt	output.txt

`#include <cstdio>`
 	
`int main()`
`{`
` int a,b;`
`  scanf ("%d%d",&a,&b);`
`   printf ("%d\n",a+b);`
`    return 0;`
`}`
	10

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. Сколько буков

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

Требуется реализовать функцию, которая будет определять, сколько в строке больших букв, маленьких букв и цифр.

Функция должна иметь сигнатуру:

```
int calcLetters(char *iStr, int *oLowerCnt, int *oUpperCnt, int *oDigitsCnt);
```

Здесь `iStr` — указатель на начало строки, завершающейся нулевым символом. Параметры `oLowerCnt`, `oUpperCnt` и `oDigitsCnt` выходные: вызывающий передаёт в них указатель на какие-нибудь локальные переменные, чтобы получить в них соответствующий результат. Функция возвращает длину строки `iStr`, в переменную `*oLowerCnt` нужно записать количество маленьких букв, в `*oUpperCnt` записать количество больших букв, а в `*oDigitsCnt` записать количество цифр.

В качестве тестовой задачи нужно прочитать все строки файла и распечатать статистику для каждой из них.

Формат входных данных

Строки файла могут содержать любые печатаемые символы ASCII, включая пробелы (коды от 32 до 126 включительно). Поэтому рекомендуется использовать `gets` для чтения строк.

Длина любой строки не превышает 100, строки могут быть пустыми. Учтите, что последняя строка файла также завершается символом перевода строки.

Формат выходных данных

Для каждой строки входного файла выведите статистику ровно в том же формате, как в примере выходных данных.

Пример

input.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
}
```

output.txt

```
Line 1 has 30 chars: 24 are letters (23 lower, 1 upper), 0 are digits.
Line 2 has 32 chars: 22 are letters (22 lower, 0 upper), 0 are digits.
Line 3 has 26 chars: 18 are letters (17 lower, 1 upper), 0 are digits.
Line 4 has 12 chars: 7 are letters (7 lower, 0 upper), 0 are digits.
Line 5 has 31 chars: 21 are letters (20 lower, 1 upper), 0 are digits.
Line 6 has 13 chars: 6 are letters (6 lower, 0 upper), 1 are digits.
Line 7 has 30 chars: 19 are letters (19 lower, 0 upper), 1 are digits.
Line 8 has 1 chars: 0 are letters (0 lower, 0 upper), 0 are digits.
```

Задача 6. 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 7. 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 8. 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 $3 \cdot 10^6$ 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 9. 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 10. Числительные

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

Вася пишет программу, которая оценивает строк кода в проекте. Всё самое сложное уже написано, осталось лишь написать вывод ответа пользователю. Для красоты Вася хочет красиво распечатать результат словами, например: “две тысячи сто тридцать пять строк”. Вам нужно написать алгоритм построения этой строки.

Формат входных данных

В первой строке задано число T — количество тестовых случаев в файле ($1 \leqslant T \leqslant 10^4$).

В каждой из остальных T строк указано одно целое число N — количество строк в проекте.

Сегодня проекты стали довольно большими, поэтому диапазон: $1 \leqslant N < 10^9$

(ссылка).

Формат выходных данных

В каждую строку выходного файла нужно записать словами, сколько строк в соответствующем проекте.

Чтобы избежать вопросов кодировок, слова нужно выводить транслитом. Мягкий знак заменяется на кавычку (ASCII 39).

Используется транслит BGN/PCGN. Онлайн-транслитератор есть на странице.

(пример приведён на следующей странице)

Пример

input.txt

```
17
1
4
9
21
33
40
54
73
173
345
797
987
1000
1986
100000
5001002
32011171
```

output.txt

```
odna stroka
chetyre stroki
devyat' strok
dvadtsat' odna stroka
tridtsat' tri stroki
sorok strok
pyat'desyat chetyre stroki
sem'desyat tri stroki
sto sem'desyat tri stroki
trista sorok pyat' strok
sem'sot devyanosto sem' strok
devyat'sot vosem'desyat sem' strok
odna tysyacha strok
odna tysyacha devyat'sot vosem'desyat shest' strok
sto tysyach strok
pyat' millionov odna tysyacha dve stroki
tridtsat' dva milliona odinnadtsat' tysyach sto sem'desyat odna stroka
```

Замечания

На вкладке «Новости» текущего тура выложен пример входных-выходных данных для этой задачи.

Задача 11. 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
Пример входного файла во вкладке "Новости" данного тура.

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

Замечания

На вкладке «Новости» текущего тура выложен пример входных-выходных данных для этой задачи.