Задача 1. Игра в алфавит

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

Два товарища играют в слова. Игроки по очереди приводят примеры слов из некоторой предметной области. Начинает Игрок 1. На каждом ходу игрок должен привести пример, который содержит указанную букву алфавита, которая изменяется на следующем шаге. В частности, Игрок 1 должен сначала привести пример слова, содержащего букву 'a'. Игрок 2 должен затем привести пример слова, содержащего букву 'b'. Игрок 1 должен на следующем шаге привести пример слова, содержащего 'c', и так далее. Если игрок не может назвать соответствующее слово, то он проигрывает, а другой игрок, соответственно, выигрывает.

Для заданной последовательности слов, которые по очереди произносили игроки, вам нужно написать программу, которая бы определяла, кто победил. Проигравшим считается тот игрок, который первым не смог привести соответствующий пример слова.

Формат входных данных

Во входном файле записана непустая строка, содержащая последовательность слов, разделенных одним пробелом. Все слова содержат маленькие латинские буквы. Количество слов не превышает 26. Длина одного слова может изменяться от 1 до 20.

Формат выходных данных

В выходной файл необходимо вывести одну из трёх возможных комбинаций слов. Если выигрывает первый игрок, то в выходной файл нужно вывести `PLAYER 1`. Если же выигрывает второй игрок, нужно вывести `PLAYER 2`. Если никто не выигрывает, то вывести `NO WINNER`.

Примеры

input.txt	output.txt
alpha beta gamma delta epsilon	PLAYER 2
january february march april may	PLAYER 1
cab cab cab ford chevy ford dodge honda	NO WINNER

Задача 2. Смешивание цветов

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

Известно, что можно смешивать различные цвета красок, чтобы получать другие цвета. В частности, для того чтобы получить зеленый цвет, нужно смешать желтый и синий, оранжевый может получиться смешиванием красного и желтого, а фиолетовый — из красного и синего.

В магазине есть автоматическая линия для смешивания красок. В нее заряжены баллончики с красками разных цветов. Она может смешивать любые два баллончика. Ваша задача по заданным требуемым цветам (`green`, `orange` или `purple`) определить, какие баллончики смешивать.

Формат входных данных

Во входном файле записана строка, состоящая из двух слов, разделенных одним пробелом.

Первое слово, записанное маленькими буквами, определяет требуемый цвет, один из трех: `green`, `orange` или `purple`.

Второе слово состоит из последовательности больших латинских букв, каждая буква представляет какой-то цвет. Цвета красный, желтый и синий будут представлены буквами `R`, `Y` и `B`, соответственно. Другие буквы тоже могут в этом слове встречаться. Буквы `R`, `Y` и `B` встречаются во втором слове только один раз. Его длина может изменяться от 3 до 26.

Формат выходных данных

В выходной файл необходимо вывести два целых числа, разделенных пробелом. Эти числа задают номера баллончиков с краской, которые нужно смешать, чтобы получить нужный цвет. Целые нужно выводить в возрастающем порядке. Нумерация баллончиков начинается с 1.

Примеры

input.txt	output.txt
`purple KWBYNXRS`	`3 7`
`green BARCYZ`	`1 5`
`orange GSWXBRY`	`6 7`

Задача 3. Разность множеств

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

Дано два множества символов $S1$ и $S2$. Вам нужно вычислить их разность $(S1 - S2)$ – новое множество, в котором присутствуют только те элементы $S1$, которых нет в $S2$.

Формат входных данных

В единственной строке входного файла через пробел записано две непустые последовательности латинских букв в верхнем регистре. Эти последовательности задают два множества – $S1$ и $S2$, соответственно. Каждый элемент множества встречается в последовательности не более одного раза. Количество элементов в каждом множестве не более 26.

Формат выходных данных

В выходной файл необходимо вывести последовательность символов, которые будут представлять разность заданных множеств. Элементы искомого множества выдавать в возрастающем порядке. Если результирующее множество – пусто, то вывести `Empty Set`.

Примеры

input.txt	output.txt
AEIOU AEIOUY	Empty Set
PQRSTUV UVWXY	PQRST
TEXAS COMPUTER	ASX

Задача 4. Слова

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

Перечислить все слова из заданной строки, не содержащие букв, из которых состоит первое слово.

Формат входных данных

Во входном файле записана строка, состоящая из слов, разделенных одним пробелом. Каждое слово состоит из маленьких латинских букв.

Формат выходных данных

В выходной файл необходимо вывести все слова, по одному на строку, обладающие заданным свойством, в порядке их встречаемости во входном файле.

Пример

input.txt	output.txt
to be or not to be that is the question	be be is

Задача 5. Развернуть строку

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 разворачивать заданную строку.

Функция должна иметь сигнатуру:

```
void reverse(char *start, int len);
```

Здесь `start` — указатель на начало строки (т.е. на первый её символ), а `len` — количество символов в ней (исключая завершающий нулевой символ).

Формат входных данных

В первой строке содержится целое число N — количество строк в файле ($2 \leqslant N \leqslant 1\,000$). В каждой из следующих N строк записана одна строка, которую нужно развернуть. Строка состоит только из символов латинского алфавита, и её длина лежит в диапазоне от 1 до 100 включительно.

Формат выходных данных

Нужно вывести N развёрнутых строк.

Пример

input.txt	output.txt
4 C is cool language	C si looc egaugnal

Задача 6. Прочитать время

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

Требуется реализовать функцию, будет считывать время из заданной строки.

Функция должна иметь сигнатуру:

```
int readTime(char *iStr, int *oHours, int *oMinutes, int *oSeconds);
```

Здесь `iStr` — указатель на строку, в которой должно быть записано время. Параметры `oHours`, `oMinutes`, `oSeconds` — выходные параметры, т.е. вызывающий должен передать в них указатель на свои локальные переменные, куда будет записаны соответсвующие результаты: `*oHours` — количество часов, `oMinutes` — количество минут, `oSeconds` — количество секунд. Функция должна возвращать код возврата: `1`, если прочитать время удалось, и `0` в случае неудачи.

Параметры `oMinutes`, `oSeconds` опциональные: вызывающий может передать в них NULL, если его, например, не интересует количество секунд. При этом если указатель `oMinutes` нулевой, то и указатель `oSeconds` тоже должен быть нулевой.

Время записано в формате *H:M:S* или *H:M*. То есть строка состоит из двух или трёх частей, отделённых друг от друга одним двоеточием. Каждая часть — это целое число из одной или двух цифр, возможно с ведущими нулями. При этом если задано две части, то это часы и минуты, а если три — то это часы, минуты и секунды. Заметим, что часы должны быть в диапазоне до 0 до 23, а минуты и секунды в диапазоне от 0 до 59.

Используя функцию, нужно решить тестовую задачу. В файле записана тестовая строка длины от 3 до 15 символов без пробелов. Нужно применить функцию `readTime` к каждой строке и распечатать результат её вызова.

Нужно вызвать функцию три раза:

1. Указатели `oHours`, `oMinutes`, `oSeconds` ненулевые. После вызова нужно распечатать код возврата и числа `*oHours`, `*oMinutes`, `*oSeconds` через пробел.
2. Указатель `oSeconds` нулевой. После вызова нужно распечатать код возврата и числа `*oHours`, `*oMinutes` через пробел.
3. Указатели `oMinutes` и `oSeconds` нулевые. После вызова нужно распечатать код возврата и число `*oHours` через пробел.

Заметим, что вызывать функцию три раза подряд по сути бессмысленно: мы делаем это только для того, чтобы протестировать все случаи с нулевыми указателями.

Формат входных данных

В единственной строке задано время в формате, указанном выше (с секундами или без).

Время на входе также может быть указано неверно. Чтобы не было разногласий, в каком случае считать время корректным, а в каком нет, гарантируется, что во всех тестах будут только ошибки двух видов:

1. Формат полностью соответствует условию, но число часов, минут или секунд выходит за пределы допустимого диапазона.
2. Формат некорректный: взято корректно записанное время, и между каждой парой символов в строке вставлен символ вертикальной черты '`|`' (ASCII 124).

Таким образом, вы можете самостоятельно решить, считать ли случаи вроде `12:001:35` или `17:0xy` корректными или нет: в тестах таких ситуаций не будет.

Формат выходных данных

Нужно вывести три строки с 4-мя, 3-мя и 2-мя целыми числами соответственно (см. описание выше в условии).

Заметьте, что если дата задана неверно, то все числа кроме кода возврата должны быть равны -1.

Пример

input.txt	output.txt
15:01:13	1 15 1 13 1 15 1 1 15
7:9	1 7 9 0 1 7 9 1 7
7:99	0 -1 -1 -1 0 -1 -1 0 -1
1|3|:|5|:|1|0	0 -1 -1 -1 0 -1 -1 0 -1

Комментарий

Вам может пригодиться функция `sscanf` и её возвращаемое значение.

Пояснение к примеру

В первом примере указаны секунды, а во втором — нет. В третьем тесте 99 минут, что выходит за пределы диапазона, поэтому возвращается неуспех. Четвертый тест неверно отформатирован: это время `13:5:10` со вставленными вертикальными чертами.

Задача 7. Склеить строки

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 конкатенировать заданные строки. Если конкретнее, она должна дописывать вторую строку в конец первой строки.

Функция должна иметь сигнатуру:

```
char* concat(char *pref, char *suff);
```

Здесь `pref` — указатель на начало первой строки, а `suff` — указатель на начало второй строки. Нужно дописать содержимое строки `suff` в конец строки `pref`. Предполагается, что вызывающий гарантирует, что в буфере `pref` хватит места на дописываемую строку.

Обе входных строки заканчиваются нулевым символом, и склеенная строка также должна заканчиваться на него. В качестве возвращаемого значения нужно вернуть указатель на конец (т.е. на нулевой символ) склеенной строки.

В качестве тестовой задачи нужно объединить все заданные строки.

Формат входных данных

В первой строке содержится целое число N — количество строк в файле ($2 \leqslant N \leqslant 10\,000$). В каждой из следующих N строк записана одна строка. Строка состоит только из символов латинского алфавита, и её длина лежит в диапазоне от 1 до 100 включительно.

Формат выходных данных

Нужно вывести одну строку, в которой объединены по порядку все N строк из входного файла.

Пример

input.txt	output.txt
4 C is cool language	Ciscoollanguage

Задача 8. Целое: значение - запись

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

Перевести заданное целое число из десятичной системы счисления в b-ичную систему счисления.

Формат входных данных

В первой строке входного файла записано целое число b – основание системы счисления, в которую нужно перевести число $(2 \leqslant b \leqslant 16)$.

Во второй строке дана десятичная запись целого числа N $(0 \leqslant b \leqslant 10^6)$.

Формат выходных данных

В выходной файл необходимо вывести представление числа N в b-ичной системе счисления. Цифры, большие 9, выводить маленькими буквами латинского алфавита.

Пример

input.txt	output.txt
2 25	11001

Задача 9. Перевод целых чисел

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

Дано число N в p-ичной системе счисления.

Требуется выполнить перевод числа N в q-ичную систему счисления.

Формат входных данных

В первой строке через пробел записаны три числа p, q и N ($2 \leqslant p, q \leqslant 36$).

Гарантируется, что значение числа N в десятичной системе счисления не превосходит 10^9.

Для записи цифр, значения которых в десятичной системе счисления имеют значения от 10 до 36, используются строчные латинские буквы ‘`a`‘, ‘`b`‘, . . ., ‘`z`‘.

Формат выходных данных

Выведите число N в q-ичной системе счисления.

Примеры

input.txt	output.txt
2 16 101010	2a
7 20 22	g
20 7 g	22

Задача 10. Дробное: значение - запись

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

Перевести заданное дробное число из десятичной системы счисления в b-ичную систему счисления.

Формат входных данных

В первой строке входного файла записаны два целых числа b и k – основание системы счисления, в которую нужно перевести число, и максимальное количество знаков после b-ичной точки $(2 \leqslant b \leqslant 16, 1 \leqslant k \leqslant 20)$.

Во второй строке дана десятичная запись дробного числа N $(0 < N < 1)$.

Формат выходных данных

В выходной файл необходимо вывести представление числа N в b-ичной системе счисления, не более чем с k знаками после точки. Выравнивающие нули не выводить. Цифры, большие 9, выводить маленькими буквами латинского алфавита.

Дробную часть вычислять с точностью до 10^{-5}.

Примеры

input.txt	output.txt
2 8 0.25	0.01
3 4 0.25	0.0202

Задача 11. Дробное: запись - значение

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

Перевести заданное дробное число из b-ичной системы счисления в десятичную систему счисления.

Формат входных данных

В первой строке входного файла записано целое число b – основание системы счисления, из которой нужно перевести число ($2 \leqslant b \leqslant 16$).

Во второй строке дана b-ичная запись дробного числа N ($0 < N < 1$). Ее длина не превосходит 50. Цифры, большие 9, обозначаются маленькими буквами латинского алфавита.

Формат выходных данных

В выходной файл необходимо вывести десятичное представление вещественного числа N с пятью знаками после точки.

Примеры

input.txt	output.txt
2 0.01	0.25000
3 0.0202	0.24691

Задача 12. Перевод вещественных чисел

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

Перевести заданное вещественное число из b_1-ичной системы счисления в b_2-ичную систему счисления.

Формат входных данных

В первой строке входного файла записаны три целых числа b_1, b_2 и k, где:

b_1 – основание системы счисления, в которой записано входное число A,

b_2 – основание системы счисления, в которую нужно перевести число A,

k – максимальное количество знаков после b_2-ичной точки ($2 \leqslant b_1, b_2 \leqslant 16, 1 \leqslant k \leqslant 20$).

Во второй строке дана запись числа A в системе счисления с основанием b_1. Число может быть и целым и вещественным. Целая часть отделяется от дробной точкой. Длина записи числа не превосходит 100. Цифры, большие 9, обозначаются маленькими буквами латинского алфавита.

Формат выходных данных

В выходной файл необходимо вывести запись числа в системе счисления с основанием b_2. Дробную часть, если таковая имеется, выводить не более чем с k знаками после точки. Несущественные нули не выводить. Вычисления производить с точностью до 10^{-5}. Цифры, большие 9, выводить маленькими буквами латинского алфавита.

Если входное число задано некорректно, то вывести слово `NO`.

Примеры

input.txt	output.txt
2 4 10 11101.01101	131.122
2 3 10 11101.01201	NO
3 9 2 2102	72