Домашняя работа №1 (2015)

Problem H11: Скобочные выражения

Напишите программу, которая читает со стандартного ввода последовательность символов. Она должна проверить, является ли ввод (до его окончания) правильным скобочным выражением. Считаются круглые скобки ( ), квадратные скобки [ ], фигурные скобки { } и угловые < >. Если является, то программа должна напечатать максимальную глубину вложенности скобок. Если не является, то программа должна напечатать -1.

Вывод должен заканчиваться переводом строки. Вся выделенная динамическая память должна быть освобождена по окончанию работы программы.

Ввод может включать и другие символы, они должны быть проигнорированы. Программа не может использовать внешнюю память. Программа может предполагать, что максимальная глубина вложенности скобок не превышает 200000.

Examples

Input	Output
abc	0
(abc]	-1
(abc)	1
((abc))	2
(abc)(abc)	1
({abc})	2

Problem H12: Частые строки

Напишите программу, которая читает строки со стандартного ввода (до конца ввода) и печатает по 1 разу все строки, встретившиеся более двух раз. Каждая строка оканчивается символом переноса строки. Строки нужно выводить в порядке их первого появления во вводе.

Предполагайте, что весь ввод умещается в оперативной памяти. Нельзя использовать внешнюю память. Вся выделенная динамическая память должна быть освобождена перед завершением программы.

Программа должна читать ввод один раз. Длина строки ограничена только размером оперативной памяти.

Examples

Input	Output
abc	
abc abc	
abc abc abc	abc
abc	abc

def abc def abc	
abc def abc abc abc def pqrrrrrrrrrrrrrrrrrrrrrrrrrr def	abc def

Problem H13: Дерево поиска

Ваша задача -- реализовать тип данных "Двоичное дерево поиска". Определите структуру для узла дерева. Он должен хранить символьную строку (не-NULL указатель на валидную память). Все операции получают такие аргументы-деревья и возвращают такие результаты-деревья, которые соответствуют типу данных "без заглавных звеньев".

Определите функцию создания пустого дерева. Функция не должна иметь аргументов и должна возвращать указатель на структуру.

Определите функцию добавления строки в дерево. Функция должна иметь 2 аргумента (корень дерева, куда надо добавить, и строка, которую надо добавить) и возвращать указатель на корень дерева с добавленной строкой. Функция может менять содержимое узлов дерева, в которое добавляется строка (тем не менее, полученное дерево должно содержать все необходимые строки и не содержать лишних строк). Если во время выполнения функции невозможно выделить новую динамическую память, функция должна возвратить`NULL` и не менять исходное дерево.

Определите функцию удаления заданной строки из дерева. Функция должна иметь те же аргументы и тип возвращаемого значения, что у функции добавления. Функция должна возвратить указатель на измененное дерево. При удалении глубина дерева не должна увеличиться.

Определите функцию проверки вхождения заданной строки в дерево. Функция должна возвращать истину, если входит, и ложь, иначе. При этом дерево не должно изменяться.

Определите функцию освобождения памяти под дерево. Она должна освободить память, занимаемую строками в дереве и узлами дерева.

Определите функцию `main`, которая считывает строки со стандартного ввода (строка заканчивается символом перевода строки или концом ввода) и делает последовательно следующее: если встретившаяся строка отсутствует в дереве, она должна быть добавлена, иначе она должна быть удалена из дерева. После окончания ввода функция `main` распечатывает строки дерева в лексикографическом порядке (после каждой строки - символ переноса строки), освобождает память и завершает выполнение программы. Предполагайте, что все строки -- не длиннее 80 символов.

Problem H14: Разведчик Абель

Разведчик Абель (настоящее имя Вильям Генрихович Фишер) перехватил отрывок вражеского сообщения с важной информацией. Он хочет узнать, что это за информация и передать своему начальству. Он знает, что перехваченные данные содержат как полезную часть, так и бесполезную, не имеющую смысла. Еще он знает, что полезная часть могла быть разбита на части и эти части могли быть помещены последовательно между бесполезными данными, причем перед каждым таким куском есть последовательность байтов `SECRET` (каждый символ - это 8-битовый символ), а после нее - последовательность байтов `STOP`. Каждая часть состоит из целого числа байтов (чего нельзя сказать про размер бесполезной информации).

Чтобы запутать разведчика, враги зашифровали свой текст, затем разбили шифрованный текст на несколько частей, снабдили их указанными ранее начальными и конечными последовательностями байтов и соединили некоторой бесполезной информацией. Из секретного донесения разведчик Абель узнал, каким образом враги шифровали текст: первый символ текста остается без изменений, второй они получили побитовым сложением по модулю 2 первого и второго символа, третий - побитовым сложением по модулю 2 первых трех символов, четвертый - четырех символов и т.д.

Наконец, он узнал, что в шифрованном вражеском сообщении не встречаются начальные и конечные последовательности байтов, помечающие часть при разбиении.

Напишите функцию `char *decode(const char *raw, int size);`, которая получает на вход перехваченный отрывок (`raw` -- указатель на валидную память из `size` байтов, в которых последовательно записаны перехваченные данные).

Функция должна возвращать строку с расшифрованным сообщением (выделив минимально необходимый кусок памяти), если оно было в `raw`, или 0, если его там не было или в системе нет достаточной памяти для размещения строки-результата.

Функция не должна использовать внешнюю память. Она должна очистить всю ненужную динамическую память по завершении.

Ваш файл не должен содержать функции `main` и подключать заголовочные файлы `stdio.h` и `stdlib.h`.