Контрольная работа №2

В рамках данной контрольной работы необходимо решить предложенные задачи на языке программирования высокого уровня из предложенного перечня. Варианты задач находятся в отдельном файле. В качестве результата выполнения практической работы необходимо приложить **архив с файлами решенных задач, а также отчет о выполнении работы**. Файлы решенных задач представляют собой файлы с исходным кодом с установленным расширением. При решении задач необходимо писать свои собственные структуры данных и методы сортировки, использование встроенных не допускается (кроме массивов).

Отчет о выполнении практической работы должен содержать:

1. Титульный лист
2. Содержание
3. 4 параграфа, в которых раскрыто решение каждой задачи. По каждой задаче необходимо представить следующую информацию:

3.1. *Условие задачи*. Берется из файла.

3.2. *Ход решения задачи*. Студент описывает логику решения данной задачи. Какие алгоритмы и для чего использованы, как построена программа. Данная часть является описательной. Здесь следует говорить именно о построении алгоритма, опуская процессы ввода и вывода данных (если это не является основной сутью алгоритма).

3.3. *Листинг программы с комментариями*. Копируется весь программный код.

3.4. *Тестирование программы*. Если для задачи предусмотрены автотесты, приложить скриншот их прохождения. Если нет: Составляется таблица, содержащая следующие поля: номер теста, входные данные, результат выполнения программы, корректность, время выполнения (мс), затраченная память (Мб). Составляется не менее 10-ти тестовых наборов данных согласно условию задачи. Тестовые наборы входных данных студент составляет самостоятельно. В обязательном порядке программа тестируется на граничных наборах входных данных (например, если N варьируется от 0 до 10^9, то обязательно рассмотреть решение задачи при N=0 и при N близком к 10^9). Если написанная программа не позволяет решить задачу при граничных входных данных, все равно включить в тест и в качестве результата написать "Не решено". В столбце "входные данные" данные впечатываются вручную, в столбце "результат..." представляется скриншот выполнения программы (если не влезает на одну страницу, делать несколько скриншотов).

Каждая контрольная работа защищается на занятии преподавателю.

Задачи по теме 4. Деревья

Задача 1. Провода

Ограничение по времени: 2 секунды

Ограничение по памяти: 64 мегабайта

На складе есть провода различной целочисленной длины. Их можно разрезать на части. Необходимо получить K кусочков одинаковой целочисленной и как можно большей длины. Найти максимальную длину M, при которой можно получить по меньшей мере K кусочков этой длины. Все оставшиеся на складе куски проводов длиной меньшей M в подсчете не участвуют.

Формат входных данных:

В первой строке – количество проводов на складе N и необходимое количество кусочков K. В следующих N строках – длины проводов.

$1 \le N \le 100000$

$1 \le K \le 100000$

Формат выходных данных:

M – максимальная длина, на которую можно разрезать все провода так, чтобы получилось не менее K кусочков.

Примеры:

Стандартный ввод	Стандартный вывод
5 7 15 12 5 13 6	6

Задача 2. Брокеры

Ограничение по времени: 1 секунда

Ограничение по памяти: 64 мегабайта

В стране Бурляндии фирма «Котлетный рай» имеет много отделений, работающих сравнительно автономно. После неких экономических преобразований такая форма функционирования оказалась невыгодной и фирма решила сливать капиталы отделений, образуя укрупненные департаменты, отвечающие за несколько отделений сразу. Цель фирмы – слить все отделения в один громадный департамент, владеющий всеми капиталами. Проблема заключается в том, что по законам Бурляндии операция слияния капиталов должна проводиться государственной брокерской службой, которая не может производить более одной операции слияния в одной фирме одновременно. Вторая проблема состоит в том, что брокерская служба берет за свои услуги один процент всех средств, получившихся в результате слияния двух подразделений. Важно спланировать порядок операций слияния таким образом, чтобы фирма потратила на слияние как можно меньшую сумму.

Формат входных данных:

На вход программы подается число отделение $2 \le N \le 1000000$, за которым следует N капиталов отделений $1 \le C_i \le 1000000$.

Формат выходных данных:

T – минимальная сумма из возможных, которую должна заплатить брокерам фирма «Котлетный рай», с двумя знаками после запятой.

Примеры:

Стандартный ввод	Стандартный вывод
5 1 2 3 4 5	0.33
10 2 10 100 30 7 4 15 2 15 80	6.52

Задача 3. Пересечение множеств

Ограничение по времени: 2 секунды

Ограничение по памяти: 64 мегабайта

Задано $2 \leq N \leq 1000$ множеств из $3 \leq M \leq$ элементов, значения которых могут находиться в диапазоне от -2000000000 до 2000000000. Значения каждого множества задаются в произвольном порядке. Гарантируется, что для одного множества все задаваемые значения различные.

Требуется найти наибольший размер множества, получаемого при пересечении какой-либо из пар из заданных множеств.

Формат входных данных:

N M

A_1[1] A_1[2] … A_1[M]

A_2[1] A_2[2] … A_2[M]

…

A_N[1] A_N[2] … A_N[M]

Формат выходных данных:

Одно целое число.

Примеры:

Стандартный ввод	Стандартный вывод
3 4 9 7 1 8 5 7 6 3	2

5 9 8 6	
4 5 -2 6 8 4 -1 5 3 10 -5 -1 7 8 -5 -1 -2 -1 8 4 9 0	3

Задача 4. Составные строки

Ограничение по времени: 0.5 секунд

Ограничение по памяти: 16 мегабайт

В первой строке входного файла содержится число $4 \le N \le 1000$, последующие N строк состоят только из заглавных букв латинского алфавита и образуют множество, то есть, равных элементов среди них нет. Длина строки не превышает 1000 букв. Некоторые из этих строк можно составить, прописав друг за другом каких-то два элемента множества, возможно, совпадающие. То есть, если исходное множество содержит элементы A, B, AB, AA, C, ABC, то элемент AB можно составить из элементов A и B, а строку AA – из элементов A и A.

Ваша задача – вывести все элементы множества, которые можно составить из пары элементов множества по одному на строку в лексикографически возрастающем порядке.

Примеры:

Стандартный ввод	Стандартный вывод
5 A AB B AA ABC	AA AB
10 ABC DEFG AB ABCAB DEFGA FG ABFG ABCAFG FGFG ABABC	ABCAB ABFG FGFG

Задача 5. Ксерокопии

Ограничение по времени: 0,2 секунды

Ограничение по памяти: 16 мегабайт

Секретарша Людочка сегодня опоздала на работу и ей срочно нужно успеть к обеду сделать N копий одного документа. В ее распоряжении имеются два ксерокса, один из которых копирует лист за x секунд, а другой – за y секунд. (Разрешается использовать как

один ксерокс, так и оба одновременно. Можно копировать не только с оригинала, но и с копии.) Помогите ей выяснить, какое минимальное время для этого потребуется.

Формат входных данных:

Во входном файле INPUT.TXT записаны три натуральных числа N, x и y, разделенные пробелом ($1 \le N \le 2 \cdot 108$, $1 \le x, y \le 10$).

Формат выходных данных:

В выходной файл OUTPUT.TXT выведите одно число – минимальное время в секундах, необходимое для получения N копий.

Пример

INPUT.TXT	OUTPUT.TXT
4 1 1	3
5 1 2	4

Источник: здесь.

Задача 6. Роман в томах

Ограничение по времени: 1 секунда

Ограничение по памяти: 16 мегабайт

В романе N глав. В i-той главе a_i страниц. Требуется издать роман в K томах так, чтобы объем самого «толстого» тома был минимален. В каждом томе главы располагаются по порядку своих номеров.

Требуется написать программу, которая найдет количество страниц в самом «толстом» томе.

Формат входных данных:

Входной текстовый файл INPUT.TXT содержит в первой строке число N ($1 \le N \le 100$). Во второй строке через пробел записаны N чисел – количество страниц в каждой главе. Количество страниц в романе не превышает 32767. В третьей строке записано число K ($1 \le K \le N$).

Формат выходных данных:

Выходной файл OUTPUT.TXT должен содержать количество страниц в самом «толстом» томе.

Пример

INPUT.TXT	OUTPUT.TXT
3 1 2 1 2	3
4 1 2 1 1 3	2

Источник: здесь.

Задача 7. Запросы сумм

Ограничение по времени: 2 секунды

Ограничение по памяти: 64 мегабайта

В первой строке файла содержатся два числа: количество элементов в массиве V: $10 \le N \le 500000$ и количество запросов $1 \le M \le 500000$. Каждый элемент массива лежит в интервале $[0 \ldots 2^{32})$.

Каждый запрос – отдельная строка, состоящая из кода запроса, который может быть равен 1 или 2 и аргументов запроса.

Запрос с кодом один содержит два аргумента, начало L и конец отрезка R массива. В ответ на этот запрос программа должна вывести значения суммы элементов массива от V[L] до V[R] включительно.

Запрос с кодом два содержит тоже два аргумента, первый из которых есть номер элемента массива V, а второй – его новое значение.

Количество выведенных строк должно совпадать с количеством запросов первого типа.

Пример

Стандартный ввод	Стандартный вывод
10 8 1 7 15 8 9 15 15 19 5 19 1 1 8 1 6 8 1 0 6 2 6 6 2 1 6 2 0 9 1 4 7 1 3 6	93 39 70 49 38

Задача 8. Поиск множеств

Ограничение по времени: 1 секунда

Ограничение по памяти: 64 мегабайта

В первой строке файла содержится три числа: N – количество эталонных множеств, M – размер каждого из множеств и K – количество пробных множеств.

Каждое из множеств содержит целые числа от 0 до 10^9, числа могут повторяться.

Требуется для каждого из пробных множеств вывести в отдельной строке цифру ‘1’, если это множество в точности совпадает с каким-либо из эталонных множеств и цифру ‘0’, если оно ни с одним не совпадает, то есть выведено должно быть в точности K строк.

$5 \le N \le 50000$

$3 \le M \le 1000$

$5 \le K \le 50000$

Примеры:

Стандартный ввод	Стандартный вывод

10 3 5 6 5 1 7 9 3 2 3 2 7 2 9 9 6 2 6 6 6 9 4 1 8 4 4 8 3 2 1 2 6 9 7 2 1 6 5 3 7 7 4 4 6 3 9 7	1 1 0 0 1
10 7 5 8 4 0 3 6 9 2 3 5 0 4 3 1 1 7 1 0 3 1 2 4 7 1 5 1 5 5 1 3 4 0 0 3 4 0 3 3 3 6 3 9 3 3 4 1 3 1 8 1 1 1 6 8 6 8 2 5 6 8 1 3 9 3 7 5 7 1 4 0 3 1 1 3 3 8 4 1 2 1 1 6 6 8 8 1 1 5 7 5 1 5 3 4 1 3 1 1 8 0 0 1 2 8 2 6	1 1 1 1 0

Задача 9. Телефонная книга

Ограничение по времени: 2 секунды

Ограничение по памяти: 64 мегабайта

Необходимо разработать программу, которая является промежуточным звеном в реализации телефонной книги. На вход подается $N \leq 1000$ команд вида

ADD User Number

DELETE User

EDITPHONE User Number

PRINT User

Согласно этим командам нужно соответственно добавить пользователя в телефонную книгу, удалить пользователя, изменить его номер и вывести на экран его

данные. В случае невозможности выполнить действие, необходимо вывести **ERROR.** Добавлять пользователя, уже существующего в телефонной книге, нельзя.

Необходимо вывести протокол работы телефонной книги

Формат входных данных:

См. пример.

Формат выходных данных:

В случае невозможности выполнения действия требуется вывести **ERROR**

В случае команды **PRINT** User

User Number

Примеры:

Стандартный ввод	Стандартный вывод
9 ADD IVAN 1178927 PRINT PETER ADD EGOR 123412 PRINT IVAN EDITPHONE IVAN 112358 PRINT IVAN PRINT EGOR DELETE EGOR EDITHPHONE EGOR 123456	ERROR IVAN 1178927 IVAN 112358 EGOR 123412 ERROR

Задача 10. Анаграммы

Ограничение по времени: 0.5 секунд

Ограничение по памяти: 256 мегабайт

Как известно, анаграммами называются слова, которые могут получиться друг из друга путем перестановки букв, например LOOP, POOL, POLO. Будем называть все слова такого рода *комплектом*.

На вход программы подается число слов $1 \le N \le 100000$. В каждой из очередных N строк присутствует одно слово, состоящее из заглавных букв латинского алфавита. Все слова имеют одинаковую длину $3 \le L \le 10000$.

Требуется определить число комплектов во входном множестве.

Формат входных данных:

N

W1

W2

…

WN

Формат выходных данных:

NumberOfComplects

Примеры:

Стандартный ввод	Стандартный вывод
8 BCB ABA BCB BAA BBC CCB CBC CBC	3

Задача 11. Кеширующий сервер

Ограничение по времени: 3 секунды

Ограничение по памяти: 64 мегабайта

Дормидонт работает в компании, которая занимается обработкой больших данных. Обрабатываемые данные находятся где-то в распределенной системе. Количество различных данных в системе ограничено и каждое данное имеет свой номер. Эти данные регулярно требуются различным клиентам и, поскольку время обращения к ним достаточно велико, для ускорения обработки информации Дормидонту поручено написать часть middlware — сервер-посредник, к которому и обращаются теперь клиенты за данными. Так как система — распределенная, а сервер — нет, все требуемые данные на сервер не помещаются, но он имеет возможность запоминать результаты своих запросов к распределенной системе. Для этого на сервере выделена ограниченная память на N запросов. Важно, что клиент не имеет возможности обращаться к распределенной системе и результат запроса к распределенной системе всегда должны оказаться на сервере.

К большой радости Дормидонта оказалось, что самые крупные и значимые клиенты всегда обращаются за одними и теми же данными в одном и том же порядке, так что у него есть последовательность запросов. Дормидонт придумал такой алгоритм, что как можно большее количество запросов исполняется из кеша сервера, без обращения к распределенной системе. Придумаете ли вы что-то подобное?

Формат входных данных:

На вход программы подается размер памяти под кеширование запросов $1 \le N \le 100000$, количество запросов $1 \le M \le 100000$ и ровно M запросов с номерами $0 \le R_i \le 10^{18}$. Количество различных номеров запросов ограничено и не превосходит 100000.

Формат выходных данных:

Требуется вывести одно число: сколько раз пришлось обратиться к распределенной системе за данными, отсутствующими в кеше. В начале работы кеш пуст.

Пример

стандартный ввод	стандартный вывод
5 15 3 1 4 1 5	9

9 2 6 5 3 5 8 7 9 3	

Замечание

В приведенном примере первые три запроса произойдут к данным под номерами 3, 1 и 4, так как их нет в кеше. Следующий запрос, 1, есть в кеше и обращения к распределенной системе не произойдет. Запросы 5 и 9 занесут их в кеш. Следующий запрос — 2 — в кеше отсутствует, но мы выкинем из кеша запрос 1 и запрос 2 займет его место. Далее, запрос 6 вытеснит из кеша значение 2(у нас есть информация о дальнейших запросах и из нее мы видим, что запрос под номером 2 больше не повториться и нет причин хранить его далее), после чего следующие три запроса удовлетворятся из кеша. Затем произойдет еще два вытеснения — 8 и 7. Итого 9 обращений к распределенной системе. Нетрудно установить, что меньше сделать нельзя.

Задачи по теме 5. Обобщенный быстрый поиск и хеш-функции

Задача 12. Подстроки

Ограничение по времени: 2 секунды

Ограничение по памяти: 64 мегабайта

Входной файл состоит из одной строки I, содержащей малые буквы английского алфавита.

Назовем подстроковой длиной L с началом S множество непрерывно следующих символов строки.

Например, строка

abcab

содержит подстроки:

длины 1: a, b, c, a, b

длины 2: ab, bc, ca, ab

длины 3: abc, bca, cab

длины 4: abca, bcab

длины 5: abcab

В строках длины 1 есть два повторяющихся элемента — a и b. Назовем весом подстрок длины L произведение максимального количества повторяющихся подстрок этой длины на длину L.

В нашем случае вес длины 1 есть 2 (2*1), длины 2 есть 4 (2*2), длины 3 — 3 (1*3), длины 4 — 4 и длины 5 — 5.

Требуется найти наибольший из всех весов различных длин.

Примеры

Стандартный ввод	Стандартный вывод
aabaabaabaabaa	24
Abcab	5

Замечание

Длина входной строки превышает 10000 символов

Задача 13. Большая книжка

Ограничение по времени: 5 секунды

Ограничение по памяти: 4 мегабайта

Заказчику понравилось решение нашей задачи по созданию записной книжки и он предложил нам более сложную задачу: создать простую базу данных, которая хранит много записей вида ключ: значение. Для работы с книжкой предусмотрены 4 команды:

ADD KEY VALUE — добавить в базу запись с ключом KEY и значением VALUE. Если такая запись уже есть, вывести ERROR.

DDELETE KEY — удалить из базы данных запись с ключом KEY. Если такой записи нет — вывести ERROR.

UPDATE KEY VALUE — заменить в записи с ключом KEY значение на VALUE. Если такой записи нет — вывести ERROR.

PRINT KEY — вывести ключ записи и значение через пробел. Если такой записи нет — вывести ERROR.

Количество входных строк в файле с данными не превышает 300000, количество первоначальных записей равно половине количества строк (первые N/2 команд есть команды ADD).

Длины ключей и данных не превосходят 4096. Ключи и данные содержат только буквы латинского алфавита и цифры и не содержат пробелов.

Особенность задачи: все данные не поместятся в оперативной памяти и поэтому придется использовать внешнюю.

Формат входных данных

См. В примерах.

Формат выходных данных

См. В примерах.

Примеры

Стандартный ввод	Стандартный вывод
10 ADD JW SJXO ADD RZBR YMW ADD ADX LVT ADD LKFLG UWM PRINT ADX UPDATE HNTP JQPVG PRINT QURWB DELETE MB	ADX LVT ERROR QURWB MEGW ERROR
15 ADD RWJSN JFTF ADD ZDH GOON ADD FCDS TCAY ADD FCDS TCAY ADD HMGVI BWK ADD JTDU TLWWN ADD IXRJ ERF ADD IAOD GRDO PRINT IXRJ PRINT JTDU PRINT IXRJ UPDATE ZDH IOX PRINT ZDH ADD GVWU RTA DELETE ZDH ADD FCDS IVFJV	IXRJ ERF JTDU TLWWN IXRJ ERF ZDH IOX ERROR

Задача 14. Сопоставление по образцу

Ограничение по времени: 2 секунды

Ограничение по памяти: 64 мегабайта

Известно, что при работе с файлами можно указывать метасимволы * и ? для отбора нужной группы файлов, причем знак * соответствует любому множеству, даже пустому, в имени файла, а символ ? Соответствует ровно одному символу в имени.

Первая строка программы содержит имя файла, состоящее только из заглавных букв латинского языка (A-Z), а вторая — образец, содержащий только заглавные буквы латинского алфавита и, возможно, символы * и ?. Строки не превышают по длине 700 символов. Требуется вывести слова YES или NO в зависимости от того, сопоставляется ли имя файла указанному образцу.

Формат входных данных

SOMETEXT

PATTERN

Формат выходных данных

YES

или

NO

Примеры

Стандартный ввод	Стандартный вывод
ABRACADABRA ABRA*ABRA	YES
FOOBAR F??B??	YES
FOOBAR F*??O*	NO

Задача 15. Точные квадраты

Ограничение по времени: 1 секунды

Ограничение по памяти: 256 мегабайта

Можете ли вы по десятичному представлению натурального числа определить, является ли это число полным квадратом? А если в числе много десятичных знаков?

Формат входных данных

Первая строка содержит $5 \le N \le 10^6$ — количество тех чисел, которые нужно проверить. В последующих N строках — десятичные представления натуральных чисел количеством десятичных цифр в представлении не более 100.

Формат выходных данных

Для каждого из чисел, являющихся полным квадратом, вывести его номер. Нумерация начинается с единицы.

Примеры

Стандартный ввод	Стандартный вывод
10 215225	3 6

264996 136161 8809 189041 870489 203601 339456 917764 266156	9
10 690380618689483212662971952 64 480003042146133517019980195 84 276155065137459780897404545 96 555769327218234611076604674064 411163477046152583020160369764 432490597212800951537070222500 441051552494387213399506150481 837628563296051442316369723225 111788952233561308675908860 84 135743897738575637883685032100	3 4 5 8

Задача 16. Такси

Ограничение по времени: 2 секунды

Ограничение по памяти: 32 мегабайта

В некотором очень большом городе руководство осознало, что в автономные такси, то есть, такси без водителя — большое благо и решило открыть 10 станций по аренде таких такси. Были получены данные о том, откуда клиенты могут заказывать машины. Было замечено, что если станция находится от клиента на расстоянии, не большем, чем некое число R, то клиент будет арендовать машину именно на этой станции, причем, если таких станций несколько — клиент может выбрать любую. Для экономии, станции решено строить только в местах возможного расположения клиентов. Задача заключается в том, чтобы определить места наилучшей постройки, то есть такие, которые могут обслужить наибольшее количество клиентов.

Формат входных данных

В первой строке входного файла — два числа, количество клиентов и значение параметра R. В каждом из последующих N — два числа, координаты X_i и Y_i i-го клиента (нумерация ведется с нуля).

Формат выходных данных

В выходном файле должно присутствовать не более 10 строк. Каждая строка должна содержать номер клиента, у которого выгоднее всего строить станцию, и количество обслуживаемых этой станцией клиентов, отличное от нуля. Выводимые строки должны быть упорядочены от наибольшего количества обслуживаемых клиентов к наименьшему. Если две и более станции могут обслужить одинаковое количество клиентов, то выше в списке должна находиться станция с меньшим номером.

Примеры

Стандартный ввод	Стандартный вывод
5 3 0 0 2 -2 5 1	0 2 1 1 2 1 3 1 4 1
10 3.000000 3.168070 1.752490 0.500730 6.436580 0.089300 0.112720 2.275440 7.508780 0.779230 4.377090 0.644400 1.381650 1.844920 1.430420 8.079870 5.225030 7.823270 5.317290 1.788400 5.426120	5 4 1 3 4 3 6 3 9 3 0 2 2 2 3 2 7 1 8 1

Задача 17. Поисковая система

Ограничение по времени: 4 секунды (Python – 6,5 секунд).

Ограничение по памяти: 128 мегабайт

Тимофей пишет свою поисковую систему.

Имеется n документов, каждый из которых представляет собой текст из слов. По этим документам требуется построить поисковый индекс. На вход системе будут подаваться запросы. Запрос — некоторый набор слов. По запросу надо вывести 5 самых релевантных документов.

Релевантность документа оценивается следующим образом: для каждого уникального слова из запроса берётся число его вхождений в документ, полученные числа для всех слов из запроса суммируются. Итоговая сумма и является релевантностью документа. Чем больше сумма, тем больше документ подходит под запрос.

Сортировка документов на выдаче производится по убыванию релевантности. Если релевантности документов совпадают — то по возрастанию их порядковых номеров в базе (то есть во входных данных).

Формат ввода

В первой строке дано натуральное число n - количество документов в базе ($1 \leq n \leq 10^4$). Далее в n строках даны документы по одному в строке. Каждый документ состоит из нескольких слов, слова отделяются друг от друга одним пробелом и состоят из маленьких латинских букв. Длина одного текста не превосходит 1000 символов. Текст не бывает пустым.

В следующей строке дано число запросов - натуральное число m ($1 \leq m \leq 10^4$). В следующих m строках даны запросы по одному в строке. Каждый запрос состоит из одного или нескольких слов. Запрос не бывает пустым. Слова отделяются друг от друга одним

пробелом и состоят из маленьких латинских букв. Число символов в запросе не превосходит 100.

Формат вывода

Для каждого запроса выведите на одной строке номера пяти самых релевантных документов. Если нашлось менее пяти документов, то выведите столько, сколько нашлось. Документы с релевантностью 0 выдавать не нужно.

Примеры

Ввод	Вывод
3 i love coffee coffee with milk and sugar free tea for everyone 3 i like black coffee without milk everyone loves new year mary likes black coffee without milk	1 2 3 2 1
6 buy flat in moscow rent flat in moscow sell flat in moscow want flat in moscow like crazy clean flat in moscow on weekends renovate flat in moscow 1 flat in moscow for crazy weekends	4 5 1 2 3

Задача 18. Хеш-таблица

Ограничение по времени: 5 секунд (Python – 15 секунд).

Ограничение по памяти: 64 мегабайта

Тимофей, как хороший руководитель, хранит информацию о зарплатах своих сотрудников в базе данных и постоянно её обновляет. Он поручил вам написать реализацию хеш-таблицы, чтобы хранить в ней базу данных с зарплатами сотрудников.

Хеш-таблица должна поддерживать следующие операции:

- put key value – добавление пары ключ-значение. Если заданный ключ уже есть в таблице, то соответствующее ему значение обновляется.
- get key – получение значения по ключу. Если ключа нет в таблице, то вывести «None». Иначе вывести найденное значение.
- delete key – удаление ключа из таблицы. Если такого ключа нет, то вывести «None», иначе вывести хранимое по данному ключу значение и удалить ключ.

В таблице хранятся уникальные ключи.

Требования к реализации:

- Нельзя использовать имеющиеся в языках программирования реализации хеш-таблиц (std::unordered_map в C++, dict в Python, HashMap в Java, и т. д.)
- Число хранимых в таблице ключей не превосходит 10^5.

- Разрешать коллизии следует с помощью метода цепочек или с помощью открытой адресации.
- Все операции должны выполняться за O(1) в среднем.
- Поддерживать рехеширование и масштабирование хеш-таблицы не требуется.
- Ключи и значения, id сотрудников и их зарплата, – целые числа. Поддерживать произвольные хешируемые типы не требуется.

Формат ввода

В первой строке задано общее число запросов к таблице n ($1 \leq n \leq 10^6$).

В следующих n строках записаны запросы, которые бывают трех видов – get, put, delete – как описано в условии.

Все ключи и значения – целые неотрицательные числа, не превосходящие 10^9.

Формат вывода

На каждый запрос вида get и delete выведите ответ на него в отдельной строке.

Пример 1

Ввод	**Вывод**
10 get 1 put 1 10 put 2 4 get 1 get 2 delete 2 get 2 put 1 5 get 1 delete 2	None 10 4 4 None 5 None
8 get 9 delete 9 put 9 1 get 9 put 9 2 get 9 put 9 3 get 9	None None 1 2 3

Задача 19. Лемма о накачке

Ограничение по времени: 2 секунды.

Ограничение по памяти: 1024 мегабайта

Вам даны две строки s, t длины n, m, соответственно. Обе строки состоят из строчных букв латинского алфавита.

Подсчитайте тройки (x,y,z) строк, для которых справедливы следующие условия:

- $s = x + y + z$ (символ $+$ обозначает конкатенацию);
- $t = x + \underbrace{y + \cdots + y}_{k \text{ раз}} + z$ для некоторого целого числа k.

Входные данные

Первая строка содержит два целых числа nn и mm ($1 \le n < m \le 10^7$) — длины строк s и t, соответственно.

Вторая строка содержит строку s длины n, состоящую из строчных латинских букв.

Третья строка содержит строку t длины m, состоящую из строчных латинских букв.

Выходные данные

Выведите одно целое число: количество допустимых троек (x,y,z).

Примеры

Ввод	**Вывод**
4 8 abcd abcbcbcd	1
3 5 aaa aaaaa	5
12 16 abbababacaab abbababababacaab	8

Примечание

В первом наборе входных данных единственной подходящей тройкой является (x,y,z)=("a","bc","d")(x,y,z)=("a","bc","d"). Действительно,

- "abcd"="a"+"bc"+"d""abcd"="a"+"bc"+"d";
- "abcbcbcd"="a"+"bc"+"bc"+"bc"+"d""abcbcbcd"="a"+"bc"+"bc"+"bc"+"d".

Во втором наборе входных данных существует 55 подходящих троек:

- (x,y,z)=("","a","aa")(x,y,z)=("","a","aa");
- (x,y,z)=("","aa","a")(x,y,z)=("","aa","a");
- (x,y,z)=("a","a","a")(x,y,z)=("a","a","a");
- (x,y,z)=("a","aa","")(x,y,z)=("a","aa","");
- (x,y,z)=("aa","a","")(x,y,z)=("aa","a","").

В третьем наборе входных данных существует 88 подходящих троек:

- (x,y,z)=("ab","ba","babacaab")(x,y,z)=("ab","ba","babacaab");
- (x,y,z)=("abb","ab","abacaab")(x,y,z)=("abb","ab","abacaab");
- (x,y,z)=("abba","ba","bacaab")(x,y,z)=("abba","ba","bacaab");
- (x,y,z)=("ab","baba","bacaab")(x,y,z)=("ab","baba","bacaab");
- (x,y,z)=("abbab","ab","acaab")(x,y,z)=("abbab","ab","acaab");
- (x,y,z)=("abb","abab","acaab")(x,y,z)=("abb","abab","acaab");
- (x,y,z)=("abbaba","ba","caab")(x,y,z)=("abbaba","ba","caab");
- (x,y,z)=("abba","baba","caab")(x,y,z)=("abba","baba","caab").

Задача 20. Совпадения

Ограничение по времени: 2 с.

Ограничение по памяти: 64 мегабайта

Дана непустая строка S. Нужно найти такое наибольшее число k и строку t, что s совпадает со строкой t, выписанной k раз подряд.

Описание входных данных

Одна строка длины N, ($1 \leqslant N \leqslant 6*10^5$), состоящая только из маленьких латинских букв.

Описание выходных данных

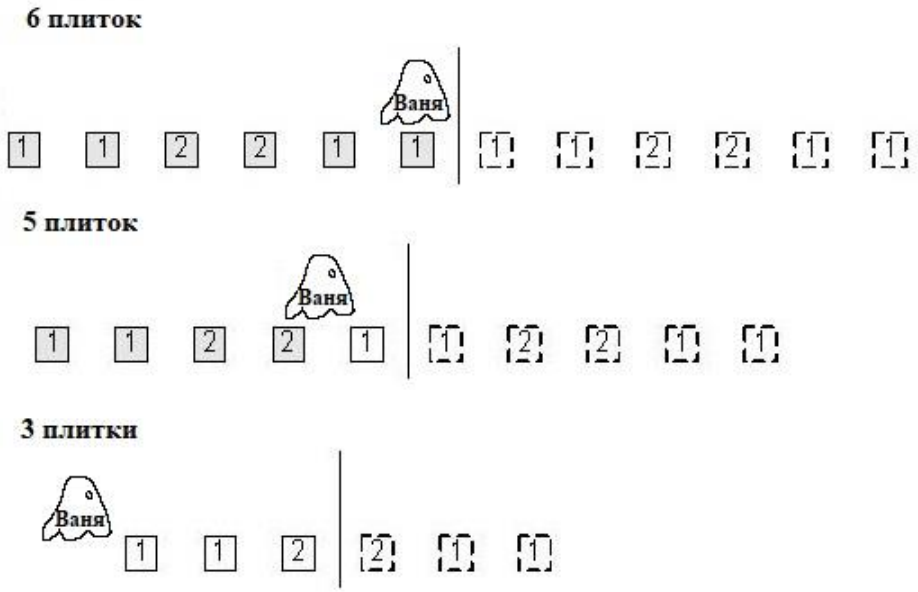
Вывести k и строку t.

Задача 21. Привидение Ваня

Ограничение по времени: 2 секунды.

Ограничение по памяти: 64 мегабайта

Привидение Ваня любит играть со своими плитками. Он любит выкладывать их в ряд и разглядывать свое творение. Однако недавно друзья решили подшутить над Ваней и поставили в его игровой комнате зеркало. Ведь всем известно, что привидения не отражаются в зеркале! А плитки отражаются. Теперь Ваня видит перед собой N цветных плиток, но не знает, какие из этих плиток настоящие, а какие — всего лишь отражение в зеркале. Помогите Ване! Выясните, сколько плиток может быть у Вани. Ваня видит отражение всех плиток в зеркале и часть плиток, которая находится перед ним. Часть плиток может быть позади Вани, их он не видит.



Описание входных данных

Первая строка входного файла содержит число N ($1 \leqslant N \leqslant 10^6$) и количество различных цветов, в которые могут быть раскрашены плитки — M ($1 \leqslant M \leqslant 10^6$). Следующая строка содержит N целых чисел от 1 до M — цвета плиток

Описание выходных данных

Выведите в выходной файл все такие K, что у Вани может быть K плиток в порядке убывания.

Задачи по теме 8. Строковые алгоритмы

Задача 22. Packed Prefix

Ограничение по времени: 0.5 с.

Ограничение по памяти: 64 Mb.

Вам даны строки в запакованном виде. Определим запакованную строку (ЗС) рекурсивно. Строка, состоящая только из строчных букв английского алфавита, является ЗС. Если A и B - корректные ЗС, то и AB является ЗС. Если A – ЗС, а n – однозначное натуральное число, то $n[A]$ тоже ЗС. При этом надпись $n[A]$ означает, что при распаковке строка A записывается подряд n раз. Найдите наибольший общий префикс распакованных строк и выведите его (в распакованном виде).

Иными словами, пусть сложение – это конкатенация двух строк, а умножение строки на число – повтор строки соответствующее число раз. Пусть функция f умеет принимать ЗС и распаковывать её. Если ЗС D имеет вид $D = AB$, где A и B тоже ЗС, то $f(D) = f(A) + f(B)$. Если $D = n[A]$, то $f(D) = f(A) \times n$.

Формат ввода

В первой строке записано число n ($1 \leq n \leq 1000$) – число строк. Далее в n строках записаны запакованные строки. Гарантируется, что эти строки корректны, то есть удовлетворяют указанному рекурсивному определению. Длина строк после распаковки не превосходит 10^5.

Формат вывода

Выведите наибольший общий префикс распакованных строк.

Примеры

Ввод	Вывод
3 2[a]2[ab] 3[a]2[r2[t]] a2[aa3[b]]	aaa
3 abacabaca 2[abac]a 3[aba]	aba

Примечания

Сложение подразумевается, как конкатенация двух строк. Умножение строки на число – повтор строки соответствующее число раз. Пусть функция f умеет принимать ЗС и распаковывать её. Если ЗС D имеет вид $D = AB$, где A и B тоже ЗС, то $f(D) = f(A) + f(B)$. Если $D = n[A]$, то $f(D) = f(A) \times n$.

Задача 23. Шпаргалка

Ограничение по времени: 0.7 с.

Ограничение по памяти: 256 Mb.

Вася готовится к экзамену по алгоритмам и на всякий случай пишет шпаргалки. Чтобы уместить на них как можно больше информации, он не разделяет слова пробелами. В итоге получается одна очень длинная строка. Чтобы на самом экзамене из-за нервов не запутаться в прочитанном, он просит вас написать программу, которая по этой длинной строке и набору допустимых слов определит, можно ли разбить текст на отдельные слова из набора.

Более формально: дан текст T и набор строк $s_1, \dots, s_n$. Надо определить, представим ли T как $s_{k_1} s_{k_2} \dots s_{k_\gamma}$, где k_i - индексы строк. Индексы могут повторяться. Строка s_i может встречаться в разбиении текста T произвольное число раз. Можно использовать не все строки для разбиения. Строки могут идти в любом порядке.

Формат ввода

В первой строке дан текст T, который надо разбить на слова. Длина T не превосходит 10^5. Текст состоит из строчных букв английского алфавита. Во второй строке записано число допустимых к использованию слов $1 \leq n \leq 100$. В последующих n строках даны сами слова, состоящие из маленьких латинских букв. Длина каждого слова не превосходит 100.

Формат вывода

Выведите «YES», если текст можно разбить на слова из данного словаря, или «NO» в ином случае.

Примеры

Ввод	Вывод
examiwillpasstheexam 5 will pass the exam i	YES
abacaba 2 abac caba	NO
abacaba 3 abac caba aba	YES

Задача 24. Книга Сандро

Источник задачи.

Задача 25. Палиндромы

Источник задачи.

Задача 26. Палиндром. Он же палиндром

Источник задачи.

Задача 27. Странный диалог

Источник задачи.

Задача 28. Басня о строке

Источник задачи.

Задача 28. Шифр Бэкона

Источник задачи.

Задача 30. Свобода выбора

Источник задачи.

Задача 31. Последнее слово Джека

Источник задачи.