Лабораторная работа 1.

Контейнеры map и set.

Часть 1. (Максимум 5 баллов)

Выполните следующие задачи по ссылке:

https://informatics.msk.ru/mod/statements/view.php?id=81066#1

Часть 2.

Задача 1. Хеширование с удалением (1 балл)

Используйте unordered_set для реализации множества строк. Хранимые строки – непустые последовательности длиной не более 10 символов, состоящие из строчных латинских букв. Структура данных должна поддерживать операции добавления строки в множество, удаления строки из множества и проверки принадлежности данной строки множеству. Максимальное количество элементов в хранимом множестве не превосходит 10^6.

Формат ввода

Каждая строка входных данных задает одну операцию над множеством.

Запись операции состоит из типа операции и следующей за ним через пробел строки, над которой проводится операция. Тип операции – один из трех символов:

\+ означает добавление данной строки в множество;

\- означает удаление строки из множества;

? означает проверку принадлежности данной строки множеству.

Общее количество операций во входном файле не превосходит 10^6. Список операций завершается строкой, в которой записан один символ # – признак конца входных данных.

При добавлении элемента в множество НЕ ГАРАНТИРУЕТСЯ, что он отсутствует в этом множестве. При удалении элемента из множества НЕ ГАРАНТИРУЕТСЯ, что он присутствует в этом множестве.

Формат вывода

Программа должна вывести для каждой операции типа ? одну из двух строк YES или NO, в зависимости от того, встречается ли данное слово в нашем множестве.

Примеры.

ВВОД	ВЫВОД
+ hello + bye ? bye	YES NO YES

- bye ? bye ? hello #	
? qwertyuiop #	NO
? asdfghjklz + asdfghjklz ? asdfghjklz #	NO YES

Задача 2. (1 балл)

Профессор написал научную книгу и составил для неё предметный указатель. Это список ключевых слов, для каждого из которых указана страница, на которой это слово встречается. Теперь профессор хочет для каждой страницы выписать в алфавитном порядке все ключевые слова, которые на эту страницу попали (если такие вообще есть). Помогите профессору решить эту задачу.

Формат ввода

Сначала задано натуральное число *n*, не превосходящее 1000 — количество слов, которое требуется обработать. Далее идут *n* строк. В каждой строке сначала записано ключевое слово. Затем идёт натуральное число, также не превосходящее 1000, — номер страницы. Ключевые слова состоят из латинских букв, не бывают пустым и по длине не превосходят 16 символов. Слова в списке, конечно, могут повторяться.

Формат вывода

Выпишите в порядке возрастания все страницы, на которых присутствуют ключевые слова. После каждого номера страницы через пробел выпишите в алфавитном порядке сами эти слова. Если на какой-то странице слово встретилось несколько раз, то повторять его не нужно. Завершающего пробела в конце строк быть не должно.

Примеры.

ВВОД	ВЫВОД
5 derivative 10 function 2 function 10 function 10 limit 7	2 function 7 limit 10 derivative function

Задача 3. word2vec (1 балл)

Известная утилита word2vec, основанная на нейронной сети, умеет приписывать каждому слову из словаря некоторый числовой вектор размерности N. При этом оказывается, что семантически «близким» словам соответствуют «близкие» векторы. Эта утилита находит широкое применение в задачах обработки естественного языка и машинного обучения.

В качестве меры «близости» между словами обычно выбирают угол между их векторами, косинус которого вычисляют по известной формуле

$$\cos\alpha = \frac{\langle u, v\rangle}{\sqrt{\langle u, u\rangle}\sqrt{\langle v, v\rangle}}$$

Однако часто векторы слов нормируют (чтобы их длина стала равна константе), и тогда в качестве меры близости рассматривают просто скалярное произведение

$$\langle u, v\rangle = \sum_{i=1}^{N} u_i v_i$$

Чем это скалярное произведение больше, тем семантически «ближе» слова друг к другу.

Вам дан словарь из *M* слов, каждому из которых сопоставлен *N*-мерный вектор. Гарантируется, что все слова различны. Требуется найти в этом списке ближайшие слова к первому слову (кроме самого этого первого слова), используя в качестве метрики близости просто скалярное произведение.

Формат ввода

В первой строке через пробел записано 2 целых числа — *M* (*2 ≤ M ≤ 100*) и *N* (*3 ≤ N ≤ 1000*) — количество слов в словаре и размерность векторов соответственно.

Далее следует *M* строк, каждая из которых описывает очередное слово.

Описание состоит из самого слова, длиной не более 15 символов, состоящего из строчных латинских букв, и набора из *N* целых чисел, разделенных пробелами. Компоненты вектора по модулю не превосходят 1000.

Формат вывода

Выпишите слова (по одному на строке), скалярное произведение векторов которых с вектором первого слова максимально. При этом скалярное произведение первого слова с самим собой учитывать не нужно. Если таких слов несколько, то их следует вывести в том же порядке, в котором они были даны на входе.

ВВОД	ВЫВОД
4 3 sweden 4 6 3 queen 0 2 2	norway

norway 4 7 5 king -1 3 2	

Задача 4. (1 балл)

Рассмотрим последовательность целых чисел длины n. По ней с шагом 1 двигается «окно» длины k, то есть сначала в «окне» видны первые k чисел, на следующем шаге в «окне» уже будут находиться k чисел, начиная со второго, и так далее до конца последовательности. Требуется для каждого положения «окна» определить минимум в нём.

Формат ввода

В первой строке входных данных содержатся два натуральных числа n и k ($n \leq 150000$, $k \leq 10000$, $k \leq n$) – длины последовательности и «окна», соответственно. На следующей строке находятся n чисел – сама последовательность.

Формат вывода

Выходные данные должны содержать $n - k + 1$ строк – минимумы для каждого положения «окна».

ВВОД	ВЫВОД
7 3 1 3 2 4 5 3 1	1 2 2 3 1

Задача 5. (2 балла)

Реализуйте систему хранения автобусных маршрутов. Вам нужно обрабатывать следующие запросы:

- **NEW_BUS *bus stop_count stop1 stop2 ...*** — добавить маршрут автобуса с названием ***bus*** и ***stop_count*** остановками с названиями ***stop1***, ***stop2***, ...
- **BUSES_FOR_STOP *stop*** — вывести названия всех маршрутов автобуса, проходящих через остановку ***stop***.
- **STOPS_FOR_BUS *bus*** — вывести названия всех остановок маршрута ***bus*** со списком автобусов, на которые можно пересесть на каждой из остановок.
- **ALL_BUSES** — вывести список всех маршрутов с остановками.

Формат ввода

В первой строке ввода содержится количество запросов Q, затем в Q строках следуют описания запросов.

Гарантируется, что все названия маршрутов и остановок состоят лишь из латинских букв, цифр и знаков подчёркивания.

Для каждого запроса **NEW_BUS** ***bus stop_count stop1 stop2 ...*** гарантируется, что маршрут ***bus*** отсутствует, количество остановок больше 0, а после числа ***stop_count*** следует именно такое количество названий остановок, причём все названия в каждом списке различны.

Формат вывода

Для каждого запроса, кроме **NEW_BUS**, выведите соответствующий ответ на него:

- На запрос **BUSES_FOR_STOP** ***stop*** выведите через пробел список автобусов, проезжающих через эту остановку, в том порядке, в котором они создавались командами **NEW_BUS**. Если остановка ***stop*** не существует, выведите **No stop**.
- На запрос **STOPS_FOR_BUS** ***bus*** выведите описания остановок маршрута ***bus*** в отдельных строках в том порядке, в котором они были заданы в соответствующей команде **NEW_BUS**. Описание каждой остановки ***stop*** должно иметь вид **Stop** ***stop*****:** ***bus1 bus2 ...***, где ***bus1 bus2 ...*** — список автобусов, проезжающих через остановку ***stop***, в порядке, в котором они создавались командами **NEW_BUS**, за исключением исходного маршрута ***bus***. Если через остановку ***stop*** не проезжает ни один автобус, кроме ***bus***, вместо списка автобусов для неё выведите **no interchange**. Если маршрут ***bus*** не существует, выведите **No bus**.
- На запрос **ALL_BUSES** выведите описания всех автобусов в алфавитном порядке. Описание каждого маршрута ***bus*** должно иметь вид **Bus** ***bus*****:** ***stop1 stop2 ...***, где ***stop1 stop2 ...*** — список остановок автобуса ***bus*** в порядке, в котором они были заданы в соответствующей команде **NEW_BUS**. Если автобусы отсутствуют, выведите **No buses**.

Предупреждение

Условие задачи выше содержит много важных деталей. Если вы не уверены в том, что не упустили ни одной, перечитайте условие ещё раз.

ВВОД	ВЫВОД
10 ALL_BUSES BUSES_FOR_STOP Marushkino STOPS_FOR_BUS 32K NEW_BUS 32 3 Tolstopaltsevo Marushkino Vnukovo	No buses No stop No bus 32 32K Stop Vnukovo: 32 32K 950 Stop Moskovsky: no interchange

NEW_BUS 32K 6 Tolstopaltsevo Marushkino Vnukovo Peredelkino Solntsevo Skolkovo BUSES_FOR_STOP Vnukovo NEW_BUS 950 6 Kokoshkino Marushkino Vnukovo Peredelkino Solntsevo Troparyovo NEW_BUS 272 4 Vnukovo Moskovsky Rumyantsevo Troparyovo STOPS_FOR_BUS 272 ALL_BUSES	Stop Rumyantsevo: no interchange Stop Troparyovo: 950 Bus 272: Vnukovo Moskovsky Rumyantsevo Troparyovo Bus 32: Tolstopaltsevo Marushkino Vnukovo Bus 32K: Tolstopaltsevo Marushkino Vnukovo Peredelkino Solntsevo Skolkovo Bus 950: Kokoshkino Marushkino Vnukovo Peredelkino Solntsevo Troparyovo

Задача 6. (2 балла)

В этой задаче вам нужно присваивать номера автобусным маршрутам.

А именно, для каждого маршрута, заданного множеством названий остановок, нужно либо выдать новый номер (первому маршруту — 1, второму — 2 и т. д.), либо вернуть номер существующего маршрута, которому соответствует такое множество остановок.

В отличие от задачи «Автобусные остановки — 2», наборы остановок, которые можно получить друг из друга перестановкой элементов или добавлением/удалением повторяющихся, следует считать одинаковыми.

Формат ввода

Сначала вводится количество запросов Q, затем Q описаний запросов.

Каждый запрос представляет собой положительное количество остановок N, за которым следуют разделённые пробелом N названий остановок соответствующего маршрута (не обязательно различных). Названия остановок состоят лишь из латинских букв и символов подчёркивания.

Формат вывода

Выведите ответ на каждый запрос в отдельной строке.

Если маршрут с данным набором остановок уже существует, в ответ на соответствующий запрос выведите **Already exists for *i***, где ***i*** — номер маршрута с таким набором остановок. В противном случае нужно выделить введённому набору остановок новый номер ***i*** и вывести его в формате **New bus *i***.

ВВОД	ВЫВОД
5 2 Marushkino Kokoshkino	New bus 1 New bus 2

1 Kokoshkino 2 Marushkino Kokoshkino 2 Kokoshkino Marushkino 2 Kokoshkino Kokoshkino	Already exists for 1 Already exists for 1 Already exists for 2

Задача 7. (2 балла)

Номер варианта.

Варианты

1. Подсчитайте в заданном файле text1.txt количество неповторяющихся слов и выведите их в файл text2.txt.

2. Подсчитайте в заданном файле text1.txt все повторяющиеся слова и выведите их в файл text2.txt.

3. Определите, какие буквы алфавита не встречаются ни разу в заданном текстовом файле text1.txt, и выведите их в файл text2.txt.

4. В заданном текстовом файле text1.txt хранятся пары значений ФИО – номер телефона. При этом ФИО может повторяться несколько раз. Организуйте text2.txt в следующем виде: ФИО – список всех его телефонов. Отсортируйте по ФИО по возрастанию.

5. Дан текстовый файл text1.txt. Требуется отсортировать все входящие в него слова по алфавиту и записать результат в файл text2.txt.

6. Дан текстовый файл text1.txt. Исключите из него все дубликаты слов. Результат запишите в text2.txt.

7. Дан текстовый файл text1.txt. Найдите самые часто повторяющиеся в нем слова и запишите их в файл text2.txt.

8. Даны текстовые файлы text1.txt и text2.txt. Найдите все общие слова.

9. Дан текстовый файл text1.txt. Требуется записать его содержимое в файл text2.txt в обратном порядке.

10. В файле text1.txt имеется список процессов Windows с указанием количества памяти, которую они занимают. В файле text2.txt имеется список процессов, которые мы дополнительно запускаем, также с указанием количества занимаемой памяти. Запишите в файл text3.txt общий список процессов, отсортированный по количеству занимаемой памяти в порядке убывания.

11. Имеется текстовый файл text1.txt со следующей структурой: в каждой строке находится запись вида: «фамилия студента» – «предмет». Названия

предметов и фамилии могут повторяться. По заданной фамилии выведите все изучаемые им предметы.

12. Имеется текстовый файл text1.txt со следующей структурой: в каждой строке находится запись вида: «год» – «наименование статьи» – «вид публикации». Годы и виды публикации (тезисы, статья,монография…) могут повторяться в любом порядке. Отсортируйте информацию по принципу: самый ранний год, потом все монографии, затем статьи в реферируемых журналах, потом статьи и тезисы. Такую сортировку произведите для каждого года.

13. Дан текстовый файл text1.txt со следующей структурой: в каждой строке файла находится запись вида: «фамилия» – «имя» – «отчество» – «год рождения». Запишите в файл text2.txt количество однофамильцев, в text3.txt – всех ровесников (год задается с клавиатуры).

14. Дан файл text1.txt со следующей структурой: в каждой строке находится запись вида: «исполнитель» – «композиция» – «альбом» – «год выхода записи». В файл text2.txt запишите все композиции одного исполнителя (можно задать с клавиатуры какого). В файл text3.txt записать все композиции из одного альбома.

15. В заданном текстовом файле text1.txt хранятся строки вида: имя потока – приоритет – время выполнения. Определить, в каком порядке завершатся потоки.

16. Дан текстовый файл text1.txt со следующей структурой: в первой строке указано число n, меньшее числа строк в файле. В остальных строках записаны имена. Постройте программу-считалочку, которая выбрасывает каждый раз по одному имени на позиции n, если доходит до конца списка, то продолжает счет с начала. В файл text2.txt записать последовательность «выброса» имен и оставшегося игрока.

17. Дан текстовый файл text1.txt со следующей структурой: в каждой строке находится запись вида:

«масть карты» – «старшинство». Количество строк в файле не превосходит 8. Определите, сколько карт каждой масти на руках у игрока. Найдите количество карт одного достоинства для каждого вида старшинства карты. Результат запишите в файл text2.txt.

18. Дан текстовый файл text1.txt со следующей структурой: в каждой строке находится запись вида:

«масть карты» – «старшинство». В файле перечислена вся колода в произвольном порядке (52 карты…). Требуется отсортировать колоду по

принципу следования мастей: червы, трефы, бубны, пики. Внутри каждой масти отсортировать карты по старшинству по возрастанию.

19. В текстовом файле text1.txt хранится информация следующего вида: количество человек в очереди в кассу, количество касс, время обслуживания в каждой кассе. Определить, за какое время очередь будет обслужена.

20. Имеется текстовый файл text1.txt со следующей структурой: в каждой строке файла записаны пары «слово» – «синоним». Заменить в заданном текстовом файле text2.txt все слова, встречающиеся в text1.txt, на синонимы.

Задача 8 (1 балл).

1. **Реализация телефонного справочника:** Создайте программу для хранения ФИО и телефонных номеров. Программа должна позволять добавлять, удалять и искать номера телефонов по имени. Иногда ФИО людей могут совпадать полностью, но это два разных человека, подумайте как поступить в этой ситуации.
2. **Частотный анализ текста:** Напишите программу, которая считает частоту встречаемости каждого слова в тексте. Выведите слово, которое будет являться медианой частоты встречаемости в тексте, а также моду.
3. **Инвертирование словаря:** Создайте программу, которая инвертирует ключи и значения, предполагая, что значения уникальны. Отдельно рассмотрите случай когда значения не уникальны.
4. **Слияние словарей:** Реализуйте функцию, которая объединяет два словаря в один, обрабатывая случаи с одинаковыми ключами.
5. **Группировка элементов по категориям:** Предположим у вас есть категории товаров, вам необходимо выполнить сортировку по заданным категориям.
6. **Отслеживание уникальных и дублирующихся элементов:** Создайте программу, которая использует **std::map** для отслеживания уникальных элементов и **std::multimap** для дублирующихся элементов в векторе.
7. **Сортировка пользовательских объектов:** Реализуйте сортировку вектора пользовательских объектов (например, структур) по определенному полю, используя **std::map**.
8. **Реализация кэша:** Создайте простую реализацию кэша с использованием **std::unordered_map**, которая поддерживает операции добавления, удаления и поиска.

9. **Подсчет голосов:** Напишите программу для подсчета голосов в выборах, где ключи — это имена кандидатов, а значения — количество голосов, используя **std::map**.
10. **Словарь синонимов:** Создайте словарь синонимов, у одного слова может быть много синонимов.
11. **Поиск общих элементов:** Реализуйте функцию, которая находит общие элементы (пересечение) двух **std::map**.
12. **Иерархический список задач:** Используйте **std::multimap** для создания иерархического списка задач, где ключи представляют собой категории задач, а значения — конкретные задачи.
13. **Автодополнение:** Разработайте простую систему автодополнения, используя **std::map**, где ключи — это начальные буквы слов, а значения — векторы слов, начинающихся на эти буквы.
14. **Отображение индексов в массиве:** Используйте **std::unordered_map** для создания отображения, которое связывает элементы массива с их индексами.
15. **Дневник расходов:** Создайте приложение для ведения дневника расходов, используя **std::multimap**, где ключи — это даты, а значения — расходы за эти даты.
16. **Индексирование документов:** Реализуйте систему индексирования слов в документах, используя **std::unordered_map**, где ключи — это слова, а значения — списки документов, содержащих эти слова.
17. **Сортировка по частоте:** Напишите программу, которая сортирует элементы по частоте их встречаемости, используя **std::map** для подсчета и **std::multimap** для сортировки по частоте.
18. **Библиотека книг:** Используйте **std::unordered_map** для создания каталога книг, где ключи — это ISBN номера, а значения — информация о книгах.
19. **Поиск подстроки:** Разработайте функцию поиска всех вхождений подстроки в строку и возвращения их индексов с использованием **std::unordered_map**.
20. **Удаление дубликатов из массива:** Используйте **std::unordered_set** для удаления всех дубликатов из массива, сохраняя только уникальные элементы.