1

Стек с минимумом

2

Великое переселение

3

Гоблины и шаманы

4

Злые свинки или Anti Angry Birds

5

Большая политика

6

Банковские счета

7

Англо-русский словарь

Задача 1.

Средняя

**Стек с минимумом**

Вам требуется реализовать стек, который умеет добавлять, удалять элементы и выводить минимум из всех элементов стека. Во входных данных вам будут даваться операции трех типов: push x - добавить элемент x в стек pop - удалить самый верхний элемент из стека min - вывести минимум в стеке

ВХОДНЫЕ ДАННЫЕ

В первой строке вводится число N - количество операций с стеком В следующих N строках вводятся команды в указанном формате. Гарантируется, что при операциях pop и min в стеке есть хотя бы один элемент

ВЫХОДНЫЕ ДАННЫЕ

На каждый запрос типа min выведите минимальное число в стеке в отдельной строке

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 7 push 3 push 2 min pop min push 5 min | 2 3 3 |

Начало формы

Задача решена верно

OK. Your score is = 100, 51/51 tests passed

Решение задачи

В каждом элементе стека храните два числа - элемент и текущий минимум. Тогда при операции push добавляйте новый элемент, где минимум будет равен минимуму из предыдущего минимума и нового элемента. В операции pop просто удалите верхний элемент. В операции min выведите минимум из верхнего элемента

Конец формы

Задача 2.

Сложная

**Великое переселение**

**Ограничение по времени работы: 2 секунды**

Лайнландия представляет из себя одномерный мир, являющийся прямой, на котором располагаются N городов, последовательно пронумерованных от 0 до N− 1. Направление в сторону от нулевого города к первому названо восточным, а обратное — западным.

Когда в Лайнландии неожиданно начался кризис, все были жители мира стали испытывать глубокое смятение. По всей Лайнландии стали ходить слухи, что на востоке живётся лучше, чем на западе.

Так и началось Великое Лайнландское переселение. Обитатели мира целыми городами отправились на восток, покинув родные улицы, и двигались до тех пор, пока не приходили в город, в котором средняя цена проживания была меньше, чем в родном.

ВХОДНЫЕ ДАННЫЕ

В первой строке дано одно число N (2 ⩽N⩽ 105) — количество городов в Лайнландии. Во второй строке дано N чисел ai ( 0 ⩽ai⩽ 109) — средняя цена проживания в городах с нулевого по (N− 1)-ый соответственно.

ВЫХОДНЫЕ ДАННЫЕ

Для каждого города в порядке с нулевого по N− 1-й выведите номер города, в который переселятся его изначальные жители. Если жители города не остановятся в каком-либо другом городе, отправившись в Восточное Бесконечное Ничто, выведите - 1 .

ПРИМЕР

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 10  1 2 3 2 1 4 2 5 3 1 | -1 4 3 4 -1 6 9 8 9 -1 |

Начало формы

Загрузите исходный код программы

Python 3

Прикрепить файл

Попытка сдачи не засчитана

Partial Solution. Your score is = 7, 7/14 tests passed

**Подсказка**

Рассмотрим пример из условия. Будем двигаться по городам слева направо. Жители городов, в которых стоимость жизни равна 1, 2, 3 отправляются в путь направо.

После этого они встречают город, в котором стоимость жизни равна 2, и в этом городе остаются жители города, где стоимость жизни была равна 3. Но жители города со стоимостью 2 отправляются в путь, поэтому сейчас в пути будут жители городов, где стоимость жизни равна 1, 2, 2.

Затем они встречают город, где стоимость жизни равна 1. В нем остаются жители городов, где стоимость жизни была 2 (два города), далее в путь направо идут жители двух городов, где стоимость жизни равна 1.

То есть в очередном встреченном городе остаются все жители левых городов, где стоимость жизни была выше (если они продолжали путь, то есть не остановились раньше), а жители текущего города добавляются к тем, кто движется вправо.

**Подсказка**

Для решения задачи удобно использовать стек, в который будем хранить стоимость жизни в тех городах, жители которых сейчас находятся в пути.

При встрече нового города (пусть стоимость жизни в нем равна cost) из стека удаляются все верхние значения, которые больше cost - жители этих городов остаются в этом городе. Затем число cost добавляется в стек. Таким образом, в стеке всегда хранятся числа в порядке неубывания (наибольшее значение наверху стека).

**Подсказка**

На самом деле понадобится два стека, в одном стеке хранятся стоимости жизни (costs), в другом стеке хранятся номера городов (nums). Эти номера городов нужны для восстановления ответа — когда из стека costs удаляется стоимость жизни в каком-то городе, из стека nums удаляется номер этого города, при этом нужно в специальном массиве ans сделать пометку, что для данного удаленного города стал известен ответ (текущий номер города).

Удобно также в два имеющихся стека добавить числа, равные -1, чтобы не проверять стеки на пустоту при каждом обращении к верхнему элементу.

Конец формы

Задача 3.

Олимпиадная

**Гоблины и шаманы**

**Ограничение по времени работы: 2 секунды**

Гоблины Мглистых гор очень любят ходить к своим шаманам. Так как гоблинов много, к шаманам часто образуются очень длинные очереди. А поскольку много гоблинов в одном месте быстро образуют шумную толку, которая мешает шаманам проводить сложные медицинские манипуляции, последние решили установить некоторые правила касательно порядка в очереди.

Обычные гоблины при посещении шаманов должны вставать в конец очереди. Привилегированные же гоблины, знающие особый пароль, встают ровно в ее середину, причем при нечетной длине очереди они встают сразу за центром.

Так как гоблины также широко известны своим непочтительным отношением ко всяческим правилам и законам, шаманы попросили вас написать программу, которая бы отслеживала порядок гоблинов в очереди.

### ВХОДНЫЕ ДАННЫЕ

В первой строке входных данный записано число N (1⩽N⩽105) — количество запросов к программе. Следующие N строк содержат описание запросов в формате:

* "+ i" - гоблин с номером i (1⩽i⩽N) встает в конец очереди.
* "\* i" - привилегированный гоблин с номером i встает в середину очереди.
* "-" - первый гоблин из очереди уходит к шаманам. Гарантируется, что на момент такого запроса очередь не пуст.

### ВЫХОДНЫЕ ДАННЫЕ

Для каждого запроса типа "-" программа должна вывести номер гоблина, который должен зайти к шаманам.

### ПРИМЕР

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 7  + 1  + 2  - + 3 + 4 - - | 1 2 3 |
| 10  + 1  + 2  \* 3  -  + 4 \* 5 - - - - | 1 3 2 5 4 |

Начало формы

Загрузите исходный код программы

Python 3

Прикрепить файл

Попытка сдачи не засчитана

Partial Solution. Your score is = 27, 27/28 tests passed

**Подсказка**

Разобьем очередь на две части пополам. Размер первой части всегда должен быть равен размеру правой части или быть на 1 больше. Всегда будем поддерживать это свойство.

**Подсказка**

Тогда добавление обычного гоблина в конец очереди – это добавление его в конец второй части, добавление привилегированного гоблина – это добавление его в начало второй части, а удаление гоблина из очереди – это удаление его из начала первой части.

**Подсказка**

После каждой операции нужно выполнить "балансировку", поддерживая необходимые требования к двум частям. Если в первой части элементов оказалось меньше, чем во второй, то нужно первый элемент из второй части добавить в конец первой части.

Конец формы

Задача 4.

Средняя

**Злые свинки или Anti Angry Birds**

Вы никогда не задумывались, почему в Angry Birds у птиц нет крыльев? Тем же вопросом задались разработчики новой игры. В их версии смысл игры прямо противоположен Angry Birds: зеленая свинка стреляет по злым птицам из лазерного ружья.

Птицы в игре представляются точками на плоскости. Выстрел сбивает только ближайшую птицу находящуюся на линии огня. При этом сбитая птица падая сбивает всех птиц, находящихся ровно под ней. Две птицы не могут находиться в одной точке. По заданному расположению птиц необходимо определить, какое минимальное количество выстрелов необходимо, чтобы все птицы были сбиты.

### ВХОДНЫЕ ДАННЫЕ

Первая строка входного файла содержит единственное целое число N,1 ≤ N ≤ 1000 — количество птиц.

Следующие N строк содержат по два натуральных числа каждая xi,yi — координаты i-ой птицы (0 < x, y ≤ 109). Свинка находится в точке с координатами (0,0).

### ВЫХОДНЫЕ ДАННЫЕ

Единственная строка выходного файла должна содержать одно целое число — минимальное количество выстрелов, необходимое для того, чтобы сбить всех птиц.

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 6  1 1  2 2  3 3  2 1  3 2  3 1 | 3 |
| 6  1 1  2 2  3 3  2 1  3 2  3 4 | 3 |

Задача 5.

Сложная

**Большая политика**

В некотором королевстве есть N провинций. Король пожелал объединить все их под своей самодержавной властью. Естественно, чтобы никто не догадался об этих планах, он будет это делать поэтапно, а именно: раз в год он будет объединять какие-то две провинции в одну. Чтобы жителям обеих провинций не было обидно, новому территориальному образованию будет присвоено новое название, которое будет отличаться от обоих старых названий. Естественно, это потребует выпуска новых паспортов для жителей обеих провинций. Очевидно, что если в первой провинции pi жителей, а во второй – pj жителей, то для них надо выпустить pi+pj новых паспортов. На следующий год король объединяет еще какие-то две провинции. И так далее, до тех пор пока вся территория королевства не будет объединена в одну большую «провинцию». Определите, какое наименьшее количество новых паспортов придется выпустить, если король будет объединять провинции оптимально с этой точки зрения.

### ВХОДНЫЕ ДАННЫЕ

В первой строке вводится число N (натуральное, не превышает 105) – количество провинций. Затем вводится N чисел – количество жителей каждой провинции (натуральное, не превосходит 109). Гарантируется, что изначально в королевстве хотя бы две провинции.

### ВЫХОДНЫЕ ДАННЫЕ

Выведите единственное число – количество новых паспортов, которые придется выпустить.

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 2  2 6 | 8 |
| 3  6 2 4 | 18 |

Начало формы

Загрузите исходный код программы

Python 3

Прикрепить файл

**Подсказка**

Это жадный алгоритм

Конец формы

Задача 6.

Лёгкая

**Банковские счета**

У банка есть клиенты. Каждый клиент имеет ровно один счет. Напишите программу (обязательно с использованием ассоциативного массива), которая будет выполнять последовательность запросов таких двух видов: 1) начинается с числа 1, затем через пробел следует имя клиента (слово из латинских букв), далее через пробел идет сумма денег, которая добавляется к счету текущего клиента (целое число, не превышает по модулю 10000). 2) начинается с числа 2, через пробел следует имя клиента. На каждый такой запрос программа должна ответить какая сумма в данный момент есть на счету заданного клиента. Если такое имя клиента пока ни разу не упоминалось в запросах вида 1, выводите вместо числа слово ERROR. В начале работы программы у всех клиентов на счету 0. Затем суммы могут становиться как положительными, так и отрицательными. Обратите внимание, что в ситуации, когда клиент снял суммарно ровно столько же денег, сколько положил, сумма на счете становится равной 0; но, раз его имя уже встречалось, нулевое значение не является основанием выводить ERROR.

### ВХОДНЫЕ ДАННЫЕ

Первая строка стандартного входного потока — количество запросов N(1 < N < 100000). Далее следуют N строк в каждой из которых описан один из двух описанных выше видов запроса.

### ВЫХОДНЫЕ ДАННЫЕ

На каждый запрос 2-го вида нужно вывести текущее значение на счету заданного клиента (или слово ERROR).

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 7  1 asdf 3  1 zxcv 5  2 asdf  1 asdf -2  2 asdf  2 lalala  2 zxcv | 3  1  ERROR  5 |

Начало формы

Задача решена верно

OK. Your score is = 100, 3/3 tests passed

Решение задачи

Просто реализуйте данную задачу с помощью unordered\_map

Конец формы

Задача 7.

Олимпиадная

**Англо-русский словарь**

**Ограничение по времени работы: 2 секунды**

Однажды, разбирая старые книги на чердаке, школьник Вася нашёл англо-латинский словарь. Английский он к тому времени знал в совершенстве, и его мечтой было изучить латынь. Поэтому попавшийся словарь был как раз кстати.

К сожалению, для полноценного изучения языка недостаточно только одного словаря: кроме англо-латинского необходим латинско-английский. За неимением лучшего он решил сделать второй словарь из первого.

Как известно, словарь состоит из переводимых слов, к каждому из которых приводится несколько слов-переводов. Для каждого латинского слова, встречающегося где-либо в словаре, Вася предлагает найти все его переводы (то есть все английские слова, для которых наше латинское встречалось в его списке переводов), и считать их и только их переводами этого латинского слова.

Помогите Васе выполнить работу по созданию латинско-английского словаря из англо-латинского.

### ВХОДНЫЕ ДАННЫЕ

В первой строке содержится единственное целое число N — количество английских слов в словаре. Далее следует N описаний. Каждое описание содержится в отдельной строке, в которой записано сначала английское слово, затем отделённый пробелами дефис (символ номер 45), затем разделённые запятыми с пробелами переводы этого английского слова на латинский. Переводы отсортированы в лексикографическом порядке. Порядок следования английских слов в словаре также лексикографический.

Все слова состоят только из маленьких латинских букв, длина каждого слова не превосходит 15 символов. Общее количество слов на входе не превышает 100000.

### ВЫХОДНЫЕ ДАННЫЕ

Выведите соответствующий данному латинско-английский словарь, в точности соблюдая формат входных данных. В частности, первым должен идти перевод лексикографически минимального латинского слова, далее — второго в этом порядке и т.д. Внутри перевода английские слова должны быть также отсортированы лексикографически.

### ПРИМЕР

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 3 apple - malum, pomum, popula fruit - baca, bacca, popum punishment - malum, multa | 7 baca - fruit bacca - fruit malum - apple, punishment multa - punishment pomum - apple popula - apple popum - fruit |