**Задача 1. Хеш-таблица**

Реализуйте структуру данных типа “множество строк” на основе динамической хеш-таблицы с открытой адресацией. Хранимые строки непустые и состоят из строчных латинских букв. Начальный размер таблицы должен быть равным 8-ми. Перехеширование выполняйте в случае, когда коэффициент заполнения таблицы достигает 3/4.

Структура данных должна поддерживать операции добавления строки в множество, удаления строки из множества и проверки принадлежности данной строки множеству.

**1\_1.** Для разрешения коллизий используйте квадратичное пробирование. i-ая проба  
g(k, i)=g(k, i-1) + i (mod m). m - степень двойки.

**1\_2.** Для разрешения коллизий используйте двойное хеширование.

Формат входных данных

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.

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 выходных данных

Программа должна вывести для каждой операции одну из двух строк OK или FAIL, в зависимости от того, встречается ли данное слово в нашем множестве.

|  |  |
| --- | --- |
| stdin | stdout |
| + hello + bye ? bye  + bye - bye ? bye ? hello | OK  OK  OK  FAIL  OK  FAIL  OK |

**Задача 2. Порядок обхода**

Дано число N < 106 и последовательность целых чисел из [-231..231] длиной N.

Требуется построить бинарное дерево, заданное наивным порядком вставки.

Т.е., при добавлении очередного числа K в дерево с корнем root, если root→Key ≤ K, то узел K добавляется в правое поддерево root; иначе в левое поддерево root.

**2\_1. Выведите элементы в порядке in-order (слева направо).**

|  |  |
| --- | --- |
| in | out |
| 3  2 1 3 | 1 2 3 |
| 3  1 2 3 | 1 2 3 |
| 3  3 1 2 | 1 2 3 |

**Задача 3. Декартово дерево**

Дано число N < 106 и последовательность пар целых чисел из [-231..231] длиной N.

Построить декартово дерево из N узлов, характеризующихся парами чисел {Xi, Yi}.

Каждая пара чисел {Xi, Yi} определяет ключ Xi и приоритет Yi в декартовом дереве.

Добавление узла в декартово дерево выполняйте второй версией алгоритма, рассказанного на лекции:

* При добавлении узла выполняйте спуск по ключу до узла P с меньшим приоритетом. Затем разбивайте найденное поддерево. Получившиеся два дерева сделайте дочерними для нового узла. Новый узел вставьте на место узла P.

Построить также наивное дерево поиска по ключам Xi методом из задачи 2.

**3\_1.** Вычислить разницу глубин наивного дерева поиска и декартового дерева.

|  |  |
| --- | --- |
| in | out |
| 10  5 11  18 8  25 7  50 12  30 30  15 15  20 10  22 5  40 20  45 9 | 2 |
| 10  38 19  37 5  47 15  35 0  12 3  0 42  31 37  21 45  30 26  41 6 | 2 |

**Задача 4. АВЛ-дерево.**

Написать АВЛ-дерево.

Дана последовательность команд добавления или удаления натуральных чисел в АВЛ-дерево. Команда добавления числа A задается положительным числом A, команда удаления числа A задается отрицательным числом “-A”. Требуется вывести высоту АВЛ-дерева после выполнения всех команд.

|  |  |
| --- | --- |
| in | out |
| 2 4 6 -2 | 2 |

**5\_2. Порядковые статистики.** Дано число N и N строк. Каждая строка содержащит команду добавления или удаления натуральных чисел, а также запрос на получение k-ой порядковой статистики. Команда добавления числа A задается положительным числом A, команда удаления числа A задается отрицательным числом “-A”. Запрос на получение k-ой порядковой статистики задается числом k. Требуемая скорость выполнения запроса - O(log n).

|  |  |
| --- | --- |
| in | out |
| 5  40 0  10 1  4 1  -10 0  50 2 | 40  40  10  4  50 |

Решение всех задач данного раздела предполагает использование стека.

Способ реализации стека может быть любым (список/динамический массив).

6\_2. Стековые анаграммы.

Пара слов называется стековой анаграмой, если одно слово можно получить из другого, проведя последовательность стековых операций с его буквами (взять очередную букву исходного слова и поместить ее в стек; взять букву из стека и добавить ее в конец выходного слова).

Для заданной пары слов требуется определить, можно ли выполнить последовательность стековых операций, переводящую первое слово во второе.

Формат входных данных. Пара слов, являющихся анаграммой

Формат выходных данных. YES, если последовательность стековых операций существует и NO в противном случае.

in

STOL SLOT

out

YES

in

ABC CAB

out

NO

7\_4. Скользящий максимум.

Дан массив натуральных чисел A[0..n), n не превосходит 10^8.

Так же задан размер некотрого окна (последовательно расположенных элементов массива)

в этом массиве k, k<=n. Требуется для каждого положения окна (от 0 и до n-k) вывести

значение максимума в окне. Скорость работы O(n log n)

Формат входных данных. Вначале вводится n - количество элементов массива. Затем вводится

n строк со значением каждого элемента. Затем вводится k - размер окна.

Формат выходных данных. Разделенные пробелом значения максимумов для каждого положения окна.

in

3

1 2 3

2

out

2 3

in

9

0 7 3 8 4 5 10 4 6

4

out

8 8 8 10 10 10