Московский Авиационный Институт
(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Лабораторная работа №2 по курсу
«Дискретный анализ»

Сбалансированные деревья

Студент: Бойцов Иван Алексеевич
Группа: М8О–212Б–22
Вариант: 5. 0
Преподаватель: Н.Д.Глушин
Оценка:___
Дата:_____
Подпись: _____

Москва, 2024.

Условие

Вариант: 5. 0

Реализовать декартово дерево с возможностью поиска, добавления и удаления элементов.

Необходимо создать программную библиотеку, реализующую указанную структуру данных, на основе которой разработать программу-словарь. В словаре каждому ключу, представляющему из себя регистронезависимую последовательность букв английского алфавита длиной не более 256 символов, поставлен в соответствие некоторый номер, от 0 до 264 - 1. Разным словам может быть поставлен в соответствие один и тот же номер.

Программа должна обрабатывать строки входного файла до его окончания. Каждая строка может иметь следующий формат:

+ word 34 — добавить слово «word» с номером 34 в словарь. Программа должна вывести строку «OK», если операция прошла успешно, «Exist», если слово уже находится в словаре.

- word — удалить слово «word» из словаря. Программа должна вывести «OK», если слово существовало и было удалено, «NoSuchWord», если слово в словаре не было найдено.

word — найти в словаре слово «word». Программа должна вывести «OK: 34», если слово было найдено; число, которое следует за «OK:» — номер, присвоенный слову при добавлении. В случае, если слово в словаре не было обнаружено, нужно вывести строку «NoSuchWord».

Метод решения

В нулевом варианте требуется сделать декартово дерево. Сразу обозначим, что декартово дерево – это структура данных, которая объединяет в себе бинарную кучу и бинарное дерево.

Сначала разберу основные операции декартового дерева:

- **split** - делит дерево на два поддерева left и right, где все узлы left меньше key, а все узлы right больше или равны key.
- **merge -** функция объединяет два поддерева в одно дерево, сохраняя порядок ключей. Если одно из поддеревьев пустое, возвращается второе. Если левое поддерево имеет более высокий приоритет, оно остаётся корнем, и к его правому поддереву присоединяется правое поддерево. Если приоритет правого поддерева выше, оно становится корнем, и левое присоединяется к его левому поддереву.
- **insert -** добавляет узел с заданным ключом и значением в дерево. Если узел с этим ключом уже существует, операция отменяется. При добавлении узел может сдвигаться вверх в зависимости от приоритета. Если новый узел добавляется в левое поддерево, приоритет проверяется, и, при необходимости, выполняется поворот вправо. Если узел добавляется вправо, аналогично проверяется приоритет и выполняется поворот влево.
- **erase -** удаляет узел с заданным ключом из дерева. Если узел найден, его левое и правое поддеревья объединяются с помощью merge, замещая удалённый узел.
- **find** - находит узел с заданным ключом, возвращает указатель.
- **rotateRight -** выполняет правый поворот вокруг заданного узла. При этом левое поддерево узла становится его новым родителем, а узел становится правым дочерним узлом нового родителя.
- **rotateLeft -** аналогично предыдущей, но выполняет левый поворот. Здесь правое поддерево узла становится его родителем, а узел — его левым потомком.

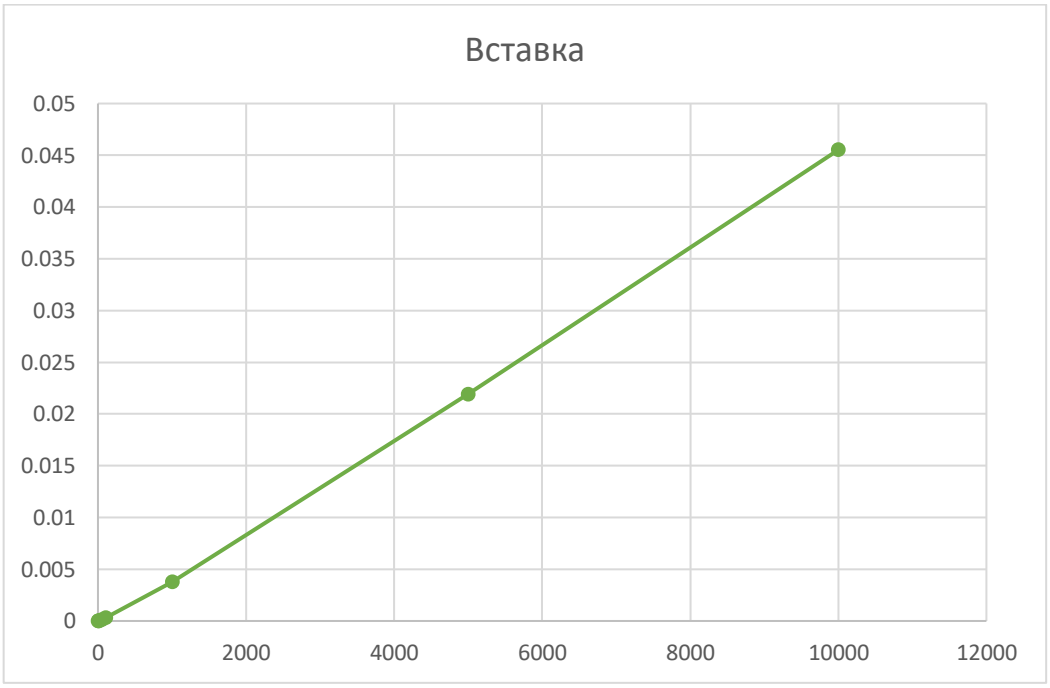
После реализации данных функций в private части класса дерева, остаётся в public вызвать данные функции, сделав ключи в нижнем регистре. А также правильно считать в основной части программы команды, вводимые в консоль.

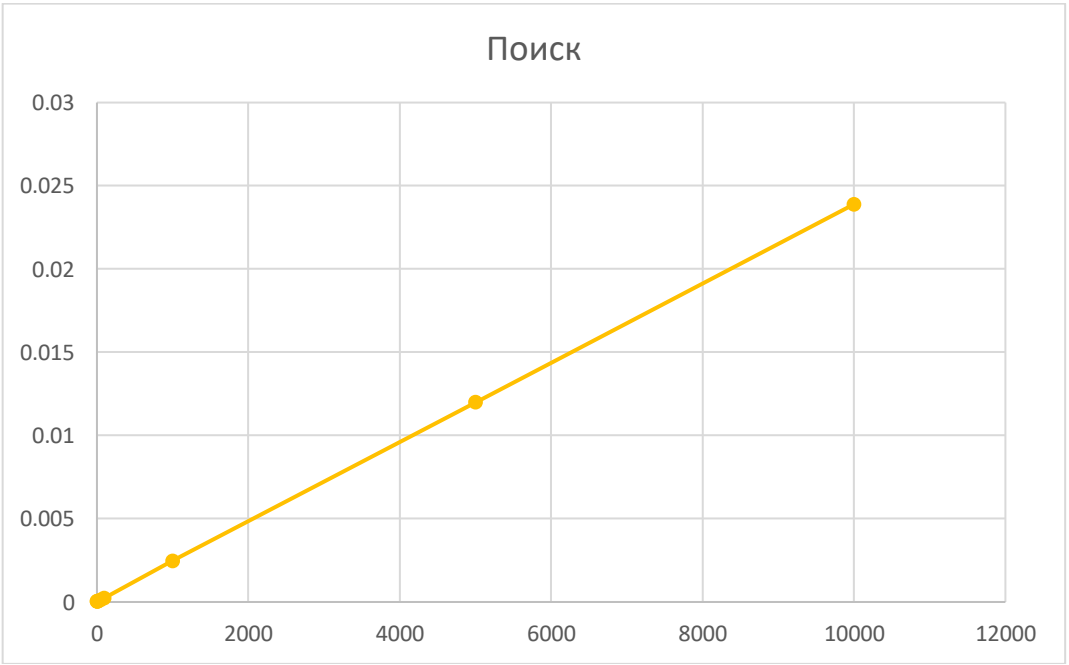
Описание программы

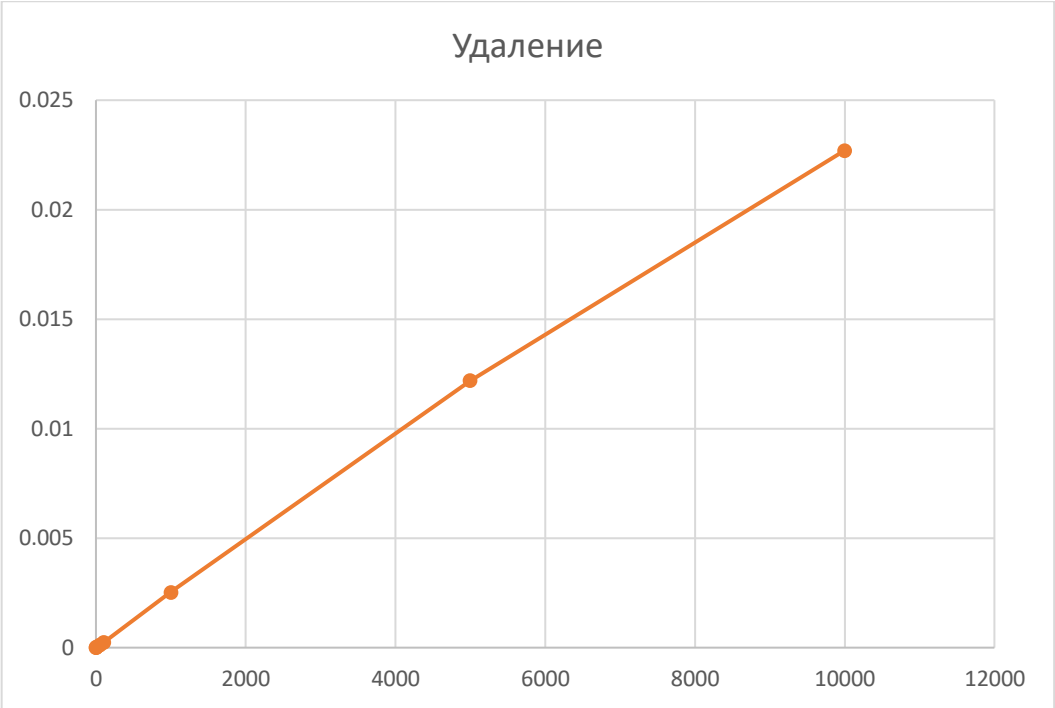
Реализована структура данных **Treap**, в которой реализованы функции добавления, поиска, удаления данных.

Тест производительности:

Производительность поиска, вставки, удаления, относительно входных данных (по y – время в сек, по x – количество данных):







Вывод:

В данной лабораторной работе я реализовал декартового дерево со всеми его основными функциями. Даже этот вариант был для меня не простым, так как я никогда не реализовывал самостоятельно деревья.