|  |
| --- |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МИРЭА – Российский технологический университет"**  **РТУ МИРЭА** |

Институт комплексной безопасности и специального приборостроения

Кафедра КБ-14 «Цифровые технологии обработки данных»

**ОТЧЕТ   
о выполнении лабораторной работы №3**

**«Деревья двоичного поиска»**

**по дисциплине   
«Алгоритмы и структуры данных»**

**Вариант № {{ task\_num }}**

Выполнил: студент 2 курса

группы **\_\_\_\_\_\_\_\_\_\_\_**

шифр \_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(фио студента)*

Проверил: *\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Москва 2022 г.

**Задание на лабораторную работу № 3.**

Реализовать в виде программы абстрактный тип данных «Дерево» согласно варианту (Номер варианта – две последние цифры шифра студента, номера зачетной книжки) с учетом заданного представления дерева.

Пусть А, В, С – деревья соответствующего типа, узлы которых могут содержать целочисленные значения. Требуется реализовать начальное формирование деревьев А и В, путем добавления некоторой последовательности значений (узлов) в пустое дерево.

После чего требуется по варианту реализовать заданную операцию над деревьями без использования каких-либо вспомогательных структур (списков, массивов и т.п.), работая только с узлами деревьев А и В.

Операция А=A ⋃прB означает, что элементы дерева В будут добавлены в дерево А в прямом порядке обхода дерева В, соответственно А=A ⋃обрB – в обратном, а А=A ⋃симB – симметричном обходе дерева В.

Операция А = A ⋂ B означает, что из дерева А исключаются узлы, отсутствующие в дереве В.

Операция А = A \ B означает, что из дерева А исключаются узлы, присутствующие в дереве В.

**Вариант № {{ task\_num }}**

**Тип дерева: Оптимальное дерево двоичного поиска**

**Реализация дерева: {{ tree\_realization }}**

**Операция: {{ tree\_operation }}**

**Порядок обхода: A – {{ tree\_a\_print }}, B – Симметричный**

**Теория о Деревьях двоичного поиска.**

Дерево двоичного поиска — это упорядоченное дерево, каждая вершина которого имеет не более двух поддеревьев, причем для каждого узла выполняется правило: в левом поддереве содержатся только ключи, имеющие значения, меньшие, чем значение данного узла, а в правом поддереве содержатся только ключи, имеющие значения, большие, чем значение данного узла. Бинарное дерево является рекурсивной структурой, поскольку каждое его поддерево само является бинарным деревом и, следовательно, каждый его узел в свою очередь является корнем дерева. Узел дерева, не имеющий потомков, называется листом.



Рисунок 1. – Иллюстрация дерева A.

Симметричный обход дерева идет в следующем порядке: левый потомок, корень, правый потомок. Пример обхода согласно дереву на рисунке 1: 40, 50, 60, 70, 80, 90, 100.

Прямой обход - корень, левый потомок, правый потомок. Пример обхода согласно дереву на рисунке 1: 70, 50, 40, 60, 90, 80,100.

Обратный обход - левый потомок, правый потомок, корень. Пример обхода согласно дереву на рисунке 1: 40, 60, 50, 80, 100, 90,70.



Рисунок 2. – Иллюстрация дерева B.



Рисунок 3. – Иллюстрация дерева C.

Иллюстрация дерева С, полученного добавлением элементов из дерева B на рисунке 2 в дерево A на рисунке 1 в прямом обходе дерева B, представлена на рисунке 3. Прямой обход полученного дерева C: 70, 50, 40, 35, 45, 60, 55, 65, 90, 80, 75, 85, 100, 95, 105.

**В качестве оптимального дерева выбрано АВЛ-дерево.**

**АВЛ-дерево** — сбалансированное по высоте двоичное дерево поиска: для каждой его вершины высота её двух поддеревьев различается не более чем на 1. **Балансировка**

Относительно АВЛ-дерева балансировкой вершины называется операция, которая в случае разницы высот левого и правого поддеревьев = 2, изменяет связи предок-потомок в поддереве данной вершины так, что разница становится <= 1, иначе ничего не меняет. Указанный результат получается вращениями поддерева данной вершины.

Используются 4 типа вращений:

[](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:AVL_LR.GIF)1.**Малое левое вращение** Данное вращение используется тогда, когда высота b-поддерева — высота L = 2 и высота С <= высота R.

[](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:AVL_BR.GIF)2.**Большое левое вращение** Данное вращение используется тогда, когда высота b-поддерева — высота L = 2 и высота c-поддерева > высота R.

[](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:AVL_LL.GIF)3.**Малое правое вращение** Данное вращение используется тогда, когда высота b-поддерева — высота R = 2 и высота С <= высота L.

[](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:AVL_BL.GIF)4.**Большое правое вращение** Данное вращение используется тогда, когда высота b-поддерева — высота R = 2 и высота c-поддерева > высота L.

В каждом случае достаточно просто доказать то, что операция приводит к нужному результату и что полная высота уменьшается не более чем на 1 и не может увеличиться. Также можно заметить, что большое вращение это комбинация правого и левого малого вращения. Из-за условия сбалансированности высота дерева О(log(N)), где N- количество вершин, поэтому добавление элемента требует O(log(N)) операций

**Листинг программы.**

{{ task\_code }}

**Скриншот работы программы.**

**{{ task\_out\_screenshot }}**

Рисунок 4. Вывод обхода деревьев

**Выводы.**

В результате выполнения данной работы была реализована структура АВЛ-дерева с операциями вставки, удаления и обхода дерева в прямом, симметричном и обратных порядках.

**Литература.**

1. Вирт Н. Алгоритмы и структуры данных. — М.: Мир, 1989.
2. Адельсон-Вельский Г. М., Ландис Е. М. Один алгоритм организации информации // Доклады АН СССР. — 1962.