МИНОБРНАУКИ РОСИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Отчет по выполнению практического задания №7.1
Тема:
Балансировка дерева поиска
Дисциплина: «Структуры и алгоритмы обработки данных»

Выполнил студент: Васильев Б.А.
Группа: ИКБО-20-23

Москва 2024

СОДЕРЖАНИЕ

ЦЕЛЬ РАБОТЫ

Получить опыт реализации самобалансирующихся двоичных деревьев поиска.

ХОД РАБОТЫ

Формулировка задачи

Составить программу создания двоичного дерева поиска и реализовать процедуры для работы с деревом согласно варианту.

Процедуры оформить в виде самостоятельных режимов работы созданного дерева. Выбор режимов производить с помощью пользовательского (иерархического ниспадающего) меню.

Провести полное тестирование программы на дереве размером n=10 элементов, сформированном вводом с клавиатуры. Тест-примеры определить самостоятельно. Результаты тестирования в виде скриншотов экранов включить в отчет по выполненной работе.

Сделать выводы о проделанной работе, основанные на полученных результатах.

Оформить отчет с подробным описанием созданного дерева, принципов программной реализации алгоритмов работы с деревом, описанием текста исходного кода и проведенного тестирования программы

Вариант	Тип значения узла	Тип дерева	Реализовать алгоритмы								
			Вставка элемента	Прямой обход	Обратный обход	Симметричный обход	Обход в ширину	Найти сумму значений листьев	Найти среднее арифметическое всех узлов	Найти длину пути от корня до заданного значения	Найти высоту дерева
1	Строка – имя	Красно-чёрное дерево	+ (и балансировка)			+	+	+			+

Рисунок 1 – Индивидуальный вариант задачи

Описание подхода к решению

Для реализации программы двоичного дерева поиска необходимо определить структуру узла дерева, в которой каждый узел содержит ключ, указатели на два дочерних узла (левый и правый), а также указатель на родительский узел. Корень дерева является отправной точкой для всех операций с деревом. Основные операции включают вставку новых элементов, симметричный обход для вывода элементов в отсортированном порядке с

использованием рекурсии, и обход в ширину, который требует использования очереди для прохождения узлов по уровням.

Методы для взаимодействия с деревом включают вставку и различные обходы дерева. Для удобства пользователя программа предоставляет иерархическое меню, позволяющее выбрать необходимый режим работы: вставка узлов, симметричный обход (рекурсивно) и обход в ширину. Указатель на родителя в каждом узле упрощает навигацию по дереву и управление его структурой при выполнении операций.\

Поскольку дерево является красно-чёрным, в каждом узле необходимо хранить дополнительное поле — цвет (красный или чёрный). Красно-чёрное дерево должно удовлетворять ряду ключевых свойств, таких как: корень всегда чёрный, красные узлы не могут иметь красных потомков, и каждый путь от корня до листа должен содержать одинаковое количество чёрных узлов. При добавлении новых элементов важно следить за выполнением этих условий и при необходимости производить балансировку дерева с помощью поворотов и перекрашивания узлов.

Методы для вставки новых узлов должны включать логику для восстановления свойств красно-чёрного дерева после каждой вставки. Если добавление нарушает баланс, программа должна автоматически применять балансирующие операции для поддержания корректности структуры дерева. Это гарантирует, что высота дерева остаётся логарифмической, что обеспечивает эффективное выполнение операций вставки и обхода.

Коды программ

Скриншоты с кодом программы вынесены в отдельный файл “Код_программ_ 7.1.pdf”.

Результаты тестирования

Выполним тестирование программы (рис. 2-8).

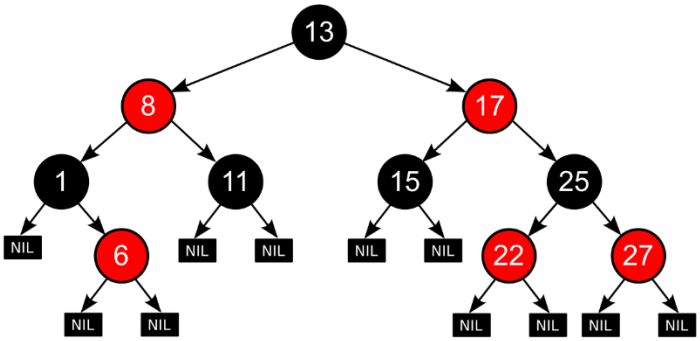

Рисунок 2 – Изначальная структура дерева

```
Initial Red-Black Tree created.


        -- Enter Your Command Code --
1: Insert
2: Display as a tree
3: In-Order print
4: Level-Order print
5: Sum of all leaf nodes
6: Print Tree Height
0: Exit
        Command Code: 2

Displaying Red-Black-Tree:
                                27-R

                    25-B

                                22-R

          17-R

                    15-B

13-B

                    11-B

          8-R

                                6-R

                    1-B
```

Рисунок 3 – Создание изначального дерева и его вывод

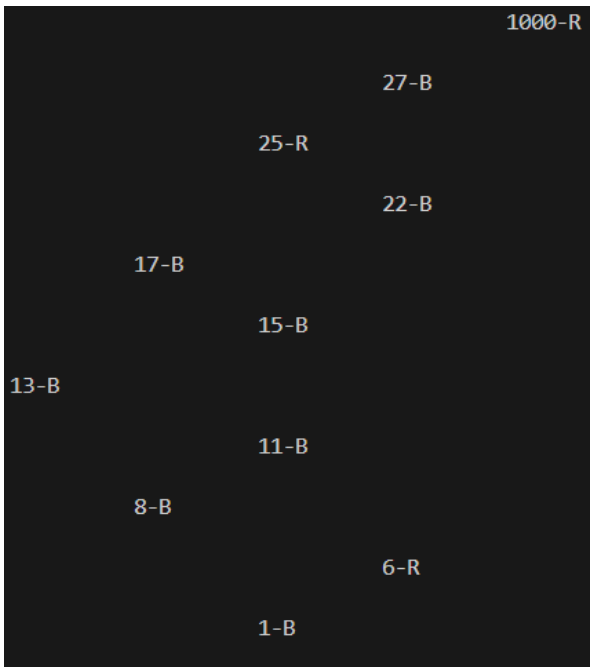

Рисунок 4 – Тестирование вставки нового элемента

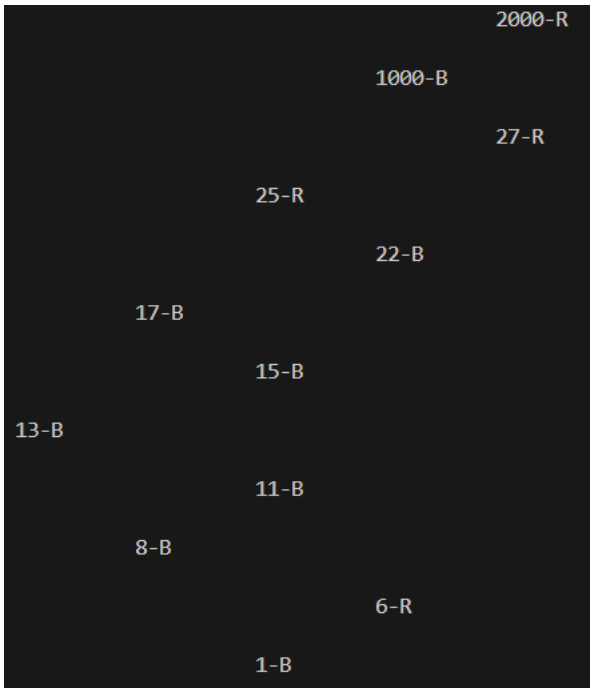

Рисунок 5 – Тестирование вставки нового элемента с балансировкой

```
        -- Enter Your Command Code --
1: Insert
2: Display as a tree
3: In-Order print
4: Level-Order print
5: Sum of all leaf nodes
6: Print Tree Height
0: Exit
        Command Code: 3

In-Order Print: 1 6 8 11 13 15 17 22 25 27

        -- Enter Your Command Code --
1: Insert
2: Display as a tree
3: In-Order print
4: Level-Order print
5: Sum of all leaf nodes
6: Print Tree Height
0: Exit
        Command Code: 4

Level-Order Print: 13 8 17 1 11 15 25 6 22 27
```

Рисунок 6 – Тестирование симметричного обхода и обхода в ширину

```
        -- Enter Your Command Code --
1: Insert
2: Display as a tree
3: In-Order print
4: Level-Order print
5: Sum of all leaf nodes
6: Print Tree Height
0: Exit
        Command Code: 5

Sum of all leaf nodes = 81
```

Рисунок 7 – Тестирование поиска суммы значений листьев

Рисунок 8 – Тестирование поиска высоты дерева.

Тестирование показало, что программа работает корректно.

ВЫВОД

В результате выполнения работы были освоены приёмы работы с самобалансирующимися деревьями. Был получен опыт реализации красно-чёрного бинарного дерева.

СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Рысин, М. Л. Введение в структуры и алгоритмы обработки данных : учебное пособие / М. Л. Рысин, М. В. Сартаков, М. Б. Туманова. — Москва : РТУ МИРЭА, 2022 — Часть 2 : Поиск в тексте. Нелинейные структуры данных. Кодирование информации. Алгоритмические стратегии — 2022. — 111 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/310826 (дата обращения: 05.10.2024).

2. Документация по языку C++ [Электронный ресурс]. URL: https://docs.microsoft.com/ruru/cpp/cpp/ (дата обращения 05.10.2024).