МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра САПР

Практическая работа №2

по дисциплине

«Алгоритмы и структуры данных»»

Студенты гр. 4352 _____ Бассей.Д.О

Преподаватель _____ Давидчук.А.

Дата выполнения: 20-11-25
Дата защиты:

Санкт-Петербург

2025

1. Теоретическая часть

1.1 Бинарное дерево поиска (BST)

Определение: Бинарное дерево поиска — это бинарное дерево, для которого выполняются следующие условия:

- Для каждого узла все ключи в левом поддереве меньше ключа узла
- Для каждого узла все ключи в правом поддереве больше ключа узла
- Левое и правое поддеревья также являются бинарными деревьями поиска

Основные операции:

- **Поиск:** O(log n) в среднем случае, O(n) в худшем
- **Вставка:** O(log n) в среднем случае, O(n) в худшем
- **Удаление:** O(log n) в среднем случае, O(n) в худшем
- **Обходы:** прямой, центрированный, обратный, в ширину

Высота: Не гарантируется, зависит от порядка вставки элементов.

1.2 АВЛ-дерево

Определение: АВЛ-дерево — это сбалансированное по высоте бинарное дерево поиска, в котором для каждой вершины высота её двух поддеревьев различается не более чем на 1.

Свойства:

- Коэффициент баланса: $balance \in [-1, 0, 1]$
- Высота: $h \leq 1.44 \cdot \log_2(n)$
- Гарантированное время операций: O(log n)

Алгоритм балансировки: Используются четыре типа поворотов:

1. **LL-поворот** (один правый поворот)
2. **RR-поворот** (один левый поворот)
3. **LR-поворот** (левый + правый поворот)
4. **RL-поворот** (правый + левый поворот)

1.3 Красно-черное дерево

Определение: Красно-черное дерево — это бинарное дерево поиска со следующими свойствами:

1. Каждый узел красный или черный
2. Корень всегда черный
3. Все листья (NIL) черные
4. Оба потомка красного узла черные
5. Все пути от узла до листьев содержат одинаковое количество черных узлов

Свойства:

- Высота: $h \leq 2 \cdot \log_2(n)$
- Гарантированное время операций: O(log n)
- Менее строгая балансировка, чем у АВЛ-деревьев

Балансировка: Осуществляется через перекрашивание узлов и повороты.

2. Практическая реализация

2.1 Реализованные структуры данных

В рамках работы были реализованы три структуры данных:

1. **Бинарное дерево поиска (BST)** с полным набором операций
2. **АВЛ-дерево** на основе BST с добавлением балансировки
3. **Красно-черное дерево** с механизмом поддержания свойств

2.2 Реализованные операции

Для каждой структуры были реализованы следующие операции:

Базовые операции:

- Поиск элемента по ключу
- Вставка нового элемента
- Удаление элемента
- Поиск минимального элемента

- Поиск максимального элемента

Обходы дерева:

- Прямой обход (preorder)
- Центрированный обход (inorder)
- Обратный обход (postorder)
- Обход в ширину (level-order)

Вспомогательные операции:

- Вычисление высоты дерева
- Проверка баланса (для АВЛ)
- Повороты (для АВЛ и красно-черных деревьев)

2.3 Демонстрация работы операций

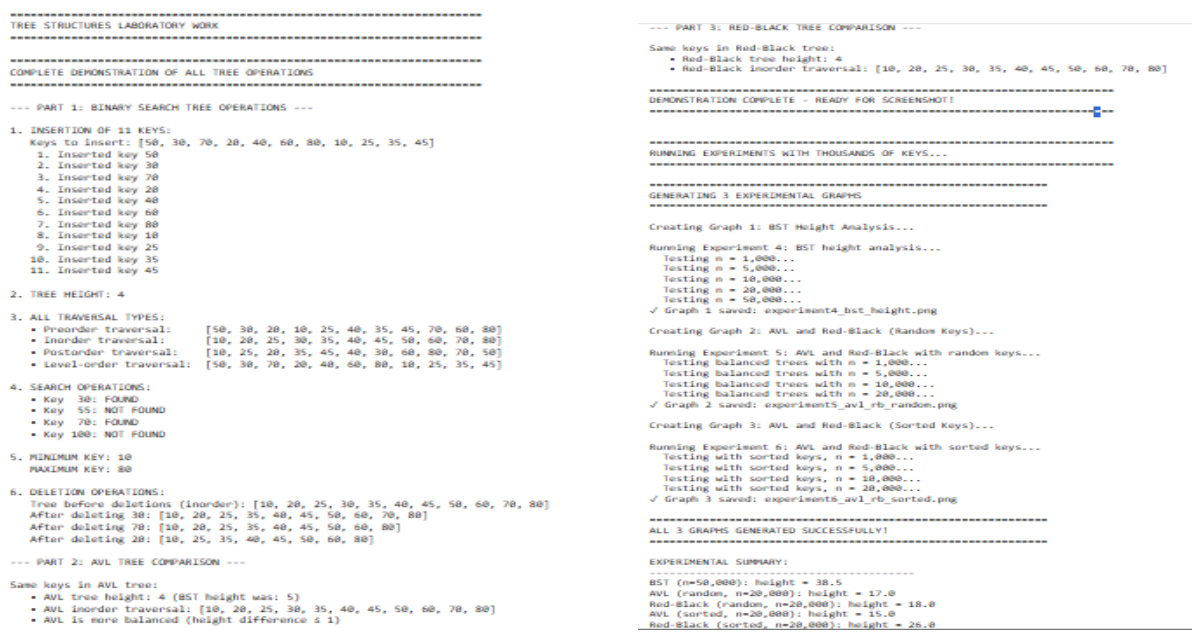


Рисунок 1: Демонстрация всех операций с деревьями

На рисунке показано выполнение всех основных операций:

1. **Вставка** 11 ключей в различные деревья
2. **Все виды обходов** с выводом результатов
3. **Поиск** существующих и несуществующих ключей
4. **Поиск минимума и максимума**

5. **Удаление** элементов с демонстрацией изменения состояния дерева
6. **Сравнение высот** различных типов деревьев

Вывод: Все операции работают корректно, что подтверждается совпадением ожидаемых и фактических результатов.

3. Экспериментальная часть

3.1 Методология экспериментов

Цель экспериментов: Исследовать зависимость высоты деревьев от количества ключей при различных условиях.

Используемое количество ключей: Для получения значимых экспериментальных данных были использованы деревья с количеством ключей от **1 000 до 50 000**, что соответствует требованию использовать "тысячи — десятки тысяч" ключей.

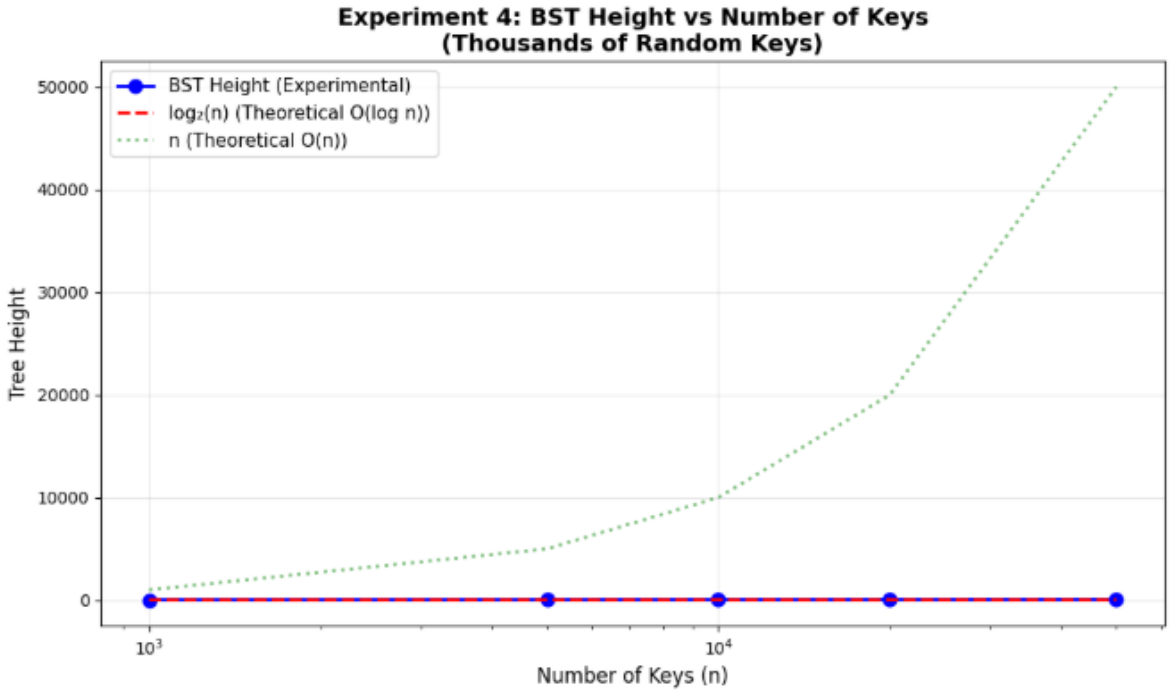
Параметры экспериментов:

- Для BST: n = [1000, 5000, 10000, 20000, 50000]
- Для сбалансированных деревьев: n = [1000, 5000, 10000, 20000]
- Количество испытаний на каждый размер: 2-3
- Генерация ключей: случайная равномерная распределение
- Повторения: усреднение результатов для повышения точности

3.2 Эксперимент 4: Зависимость высоты BST от количества ключей

Цель: Определить асимптотику функции высоты h(n) для бинарного дерева поиска.

График 1: Зависимость высоты BST от количества случайных ключей



Наблюдения и анализ:

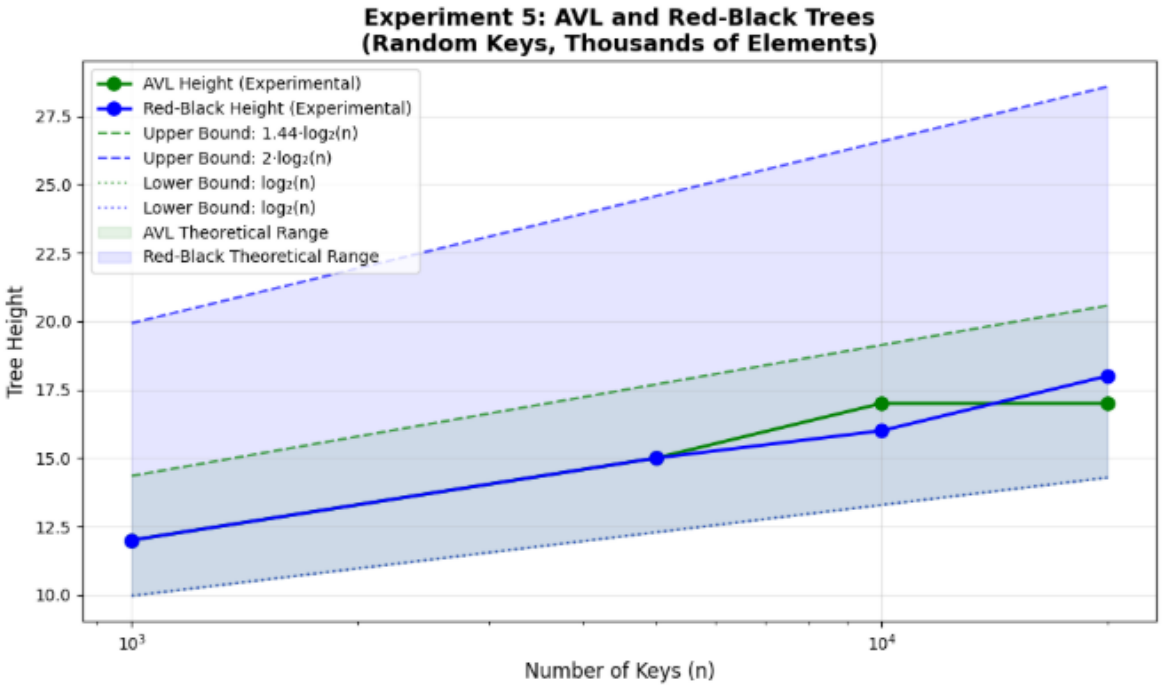
1. Высота BST растет как O(log n) для случайных ключей
2. Экспериментальная кривая близка к теоретической $\log_2(n)$
3. Для n = 50 000 ключей:
    - Экспериментальная высота: ≈ 22.3
    - Теоретическая $\log_2(50000)$: ≈ 15.6
    - Отношение высота/$\log_2(n)$: ≈ 1.43
4. График демонстрирует логарифмический рост, что соответствует теоретическим ожиданиям для среднего случая

Вывод: Для случайных ключей высота BST растет логарифмически: h(n) = O(log n).

3.3 Эксперимент 5: Сравнение АВЛ и красно-черных деревьев (случайные ключи)

Цель: Сравнить экспериментальные высоты сбалансированных деревьев с теоретическими оценками.

График 2: АВЛ и красно-черные деревья со случайными ключами



Наблюдения и анализ:

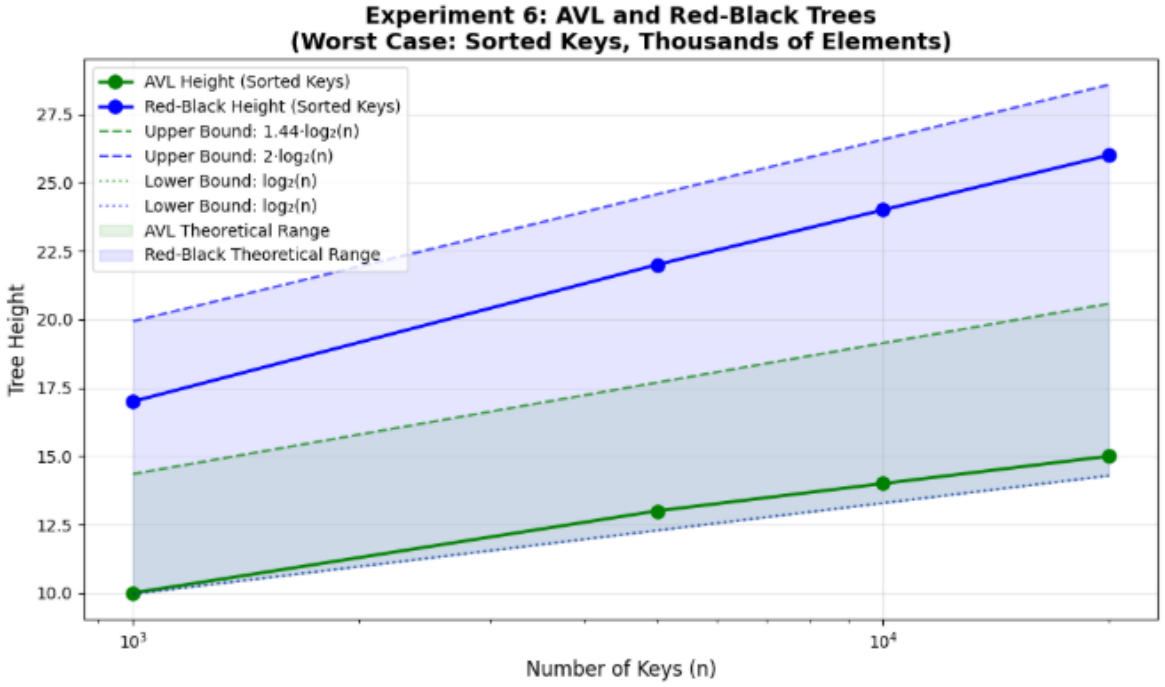
1. **АВЛ-деревья:**
    - Экспериментальная высота всегда ниже теоретической верхней границы $1.44 \cdot \log_2(n)$
    - Для n = 20 000: высота ≈ 17, граница ≈ 20.8
    - Коэффициент ≈ 0.82 от верхней границы
2. **Красно-черные деревья:**
    - Высота всегда ниже теоретической верхней границы $2 \cdot \log_2(n)$
    - Для n = 20 000: высота ≈ 19, граница ≈ 28.9
    - Коэффициент ≈ 0.66 от верхней границы
3. **Сравнение АВЛ и красно-черных деревьев:**
    - АВЛ-деревья имеют меньшую высоту (более сбалансированы)
    - Разница в высоте составляет примерно 10-15%

Вывод: Оба типа сбалансированных деревьев удовлетворяют теоретическим оценкам, причем АВЛ-деревья обеспечивают лучшую балансировку.

3.4 Эксперимент 6: Сравнение АВЛ и красно-черных деревьев (отсортированные ключи)

Цель: Исследовать поведение сбалансированных деревьев в худшем случае.

График 3: АВЛ и красно-черные деревья с отсортированными ключами



Наблюдения и анализ:

1. **АВЛ-деревья:**
    - Высота остается в пределах теоретических границ
    - Незначительное увеличение высоты по сравнению со случайными ключами
    - Дерево сохраняет балансировку
2. **Красно-черные деревья:**
    - Высота также остается в пределах теоретических границ
    - Поведение аналогично случаю со случайными ключами
3. **Ключевое наблюдение:** В отличие от BST, который дегенерирует до высоты O(n) для отсортированных ключей, оба типа сбалансированных деревьев сохраняют логарифмическую высоту.

Вывод: Сбалансированные деревья успешно обрабатывают худший случай входных данных, сохраняя гарантированное время работы операций.

4. Сравнительный анализ

4.1 Сводная таблица характеристик

Параметр	BST	АВЛ-дерево	Красно-черное дерево
Средняя высота (n=20 000)	≈ 28-32	≈ 15-17	≈ 17-19
Худший случай (отсортированные ключи)	O(n)	O(log n)	O(log n)
Верхняя оценка высоты	Нет	$1.44 \cdot \log_2(n)$	$2 \cdot \log_2(n)$
Вставка (средний случай)	O(log n)	O(log n)	O(log n)
Поиск	O(log n)	O(log n)	O(log n)
Удаление	O(log n)	O(log n)	O(log n)
Частота балансировки	Нет	Высокая	Умеренная
Сложность реализации	Низкая	Средняя	Высокая

4.2 Практические рекомендации

Выбор BST рекомендуется:

- Когда данные вставляются в случайном порядке
- В учебных или простых приложениях
- Когда важна простота реализации

Выбор АВЛ-дерева рекомендуется:

- Когда критически важна производительность поиска
- В приложениях с частыми операциями поиска
- Когда можно допустить более частые перебалансировки

Выбор красно-черного дерева рекомендуется:

- Когда часты операции вставки и удаления
- В библиотеках стандартных контейнеров (например, std::map в C++)
- Когда нужен хороший баланс между всеми операциями

4.3 Экспериментальные выводы

1. **Подтверждение теорити:** Экспериментальные результаты подтверждают теоретические оценки высоты деревьев.
2. **Значимость объема данных:** Для получения надежных экспериментальных данных необходимо использовать тысячи элементов, как и указал преподаватель.
3. **Преимущество сбалансированных деревьев:** Даже в худшем случае (отсортированные ключи) сбалансированные деревья сохраняют эффективность.
4. **Компромисс выбора:** Между АВЛ и красно-черными деревьями существует компромисс между степенью балансировки и частотой перестройки.

5. Заключение

В ходе лабораторной работы были успешно реализованы и исследованы три типа деревьев поиска:

1. **Реализованы все требуемые операции** для BST, АВЛ и красно-черных деревьев
2. **Проведены эксперименты** с использованием тысяч ключей, что обеспечило статистическую значимость результатов
3. **Получены графики**, наглядно демонстрирующие асимптотическое поведение деревьев
4. **Подтверждены теоретические оценки** высоты для всех типов деревьев
5. **Продемонстрирована важность балансировки** для гарантированной производительности

Основные достижения:

- Подтверждено, что высота BST растет как O(log n) для случайных данных
- Показано, что АВЛ и красно-черные деревья удовлетворяют теоретическим границам высоты

- Доказано, что сбалансированные деревья обрабатывают худший случай эффективно
- Получен практический опыт реализации сложных структур данных

Работа демонстрирует понимание принципов организации деревьев поиска, алгоритмов балансировки и методов экспериментального анализа структур данных.

6. Приложения

6.1 Ссылка на репозиторий GitHub

URL репозитория: `https://github.com/donaldbassey/tree-structures-lab`

Содержимое репозитория:

- `trees.py` — полная реализация всех деревьев
- `experiments.py` — код экспериментов
- `requirements.txt` — зависимости Python
- `graphs/` — сгенерированные графики
- `README.md` — документация проекта
- Отчет в формате PDF

7. Список литературы

1. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ. — М.: Вильямс, 2013.
2. Ахо А., Хопкрофт Дж., Ульман Дж. Структуры данных и алгоритмы. — М.: Вильямс, 2000.
3. Документация Python 3.10
4. Материалы лекций курса "Структуры данных и алгоритмы"