Отчет по Лабораторной работе №6

«Типы и Структуры Данных»

# Подготовил: Воякин А. Я.

**Группа ИУ7-34Б**

**Вариант 6**

**Цель работы**: построить дерево, вывести его на экран в виде дерева, реализовать основные операции работы с деревом: обход дерева, включение, исключение и поиск узлов, сбалансировать дерево, сравнить эффективность алгоритмов сортировки и поиска в зависимости от высоты деревьев и степени их ветвления; построить хеш-таблицу и вывести ее на экран, устранить коллизии, если они достигли указанного предела, выбрав другую хеш-функцию и реструктуризировав таблицу; сравнить эффективность поиска в сбалансированных деревьях, в двоичных деревьях поиска (ДДП) и в хеш-таблицах. Сравнить эффективность реструктуризации таблицы для устранения коллизий с эффективностью поиска в исходной̆ таблице.

**Задание:** построить ДДП, сбалансированное двоичное дерево (АВЛ) и хеш-таблицу по указанным данным. Сравнить эффективность поиска в ДДП в АВЛ дереве и в хеш-таблице (используя открытую или закрытую адресацию). Вывести на экран деревья и хеш-таблицу. Подсчитать среднее количество сравнений для поиска данных в указанных структурах. Произвести реструктуризацию хеш-таблицы, если среднее количество сравнений больше указанного. Оценить эффективность использования этих структур (по времени и памяти) для поставленной̆ задачи. Оценить эффективность поиска в хеш-таблице при различном количестве коллизий.

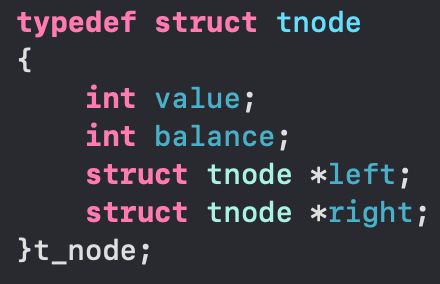
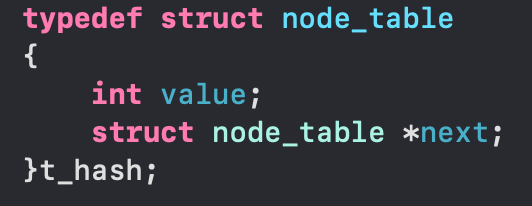
**Вариант:** В текстовом файле содержатся целые числа. Построить ДДП из чисел файла. Вывести его на экран в виде дерева. Сбалансировать полученное дерево и вывести его на экран. Построить хеш-таблицу из чисел файла. Использовать метод цепочек для устранения коллизий. Осуществить поиск введенного целого числа в ДДП, в сбалансированном дереве, в хеш-таблице и в файле. Сравнить время поиска, объем памяти и количество сравнений при использовании различных (4-х) структур данных. Если количество сравнений в хеш-таблице больше указанного (вводить), то произвести реструктуризацию таблицы, выбрав другую функцию.

**Входные данные:** Целые числа в файле, записанные через пробел.

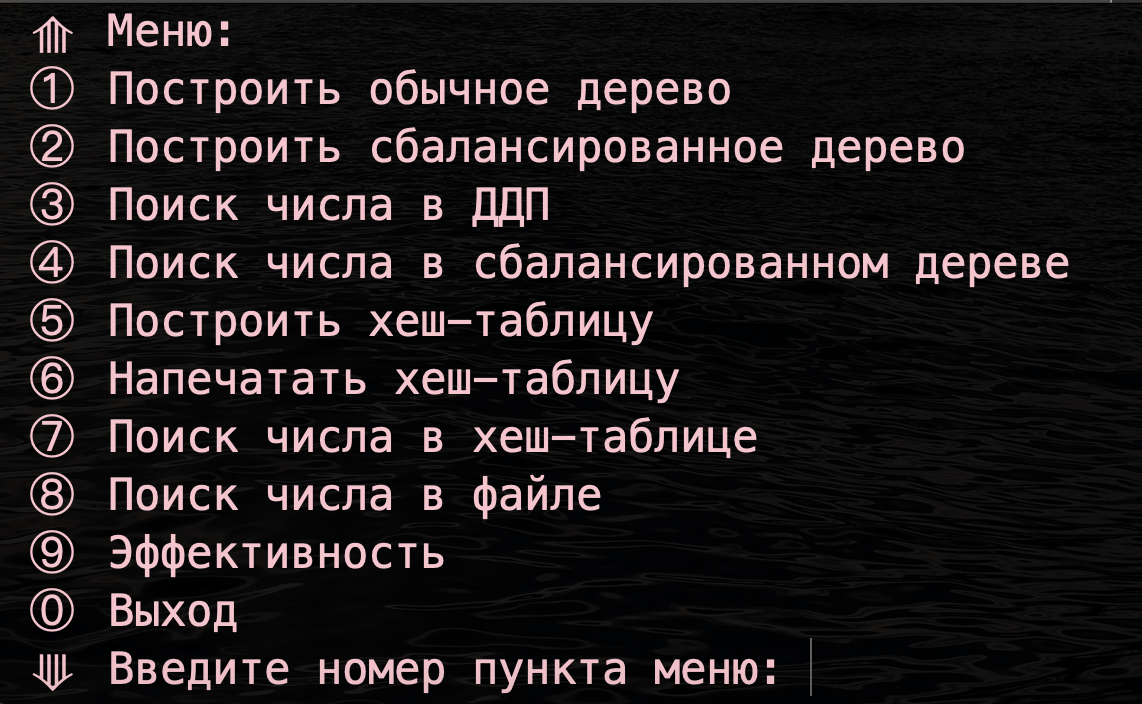
**Выходные данные:** ДДП (в виде png), АВЛ (в виде png), хеш-таблица (в виде png), оценка эффективности поиска элемента (для деревьев, хеш-таблицы и файла), кол-во сравнений при поиске отсутствующего числа.

**Внутренняя структура данных:**

*Дерево: Хеш-таблица:*

**Меню:**



**Аварийные ситуации:**

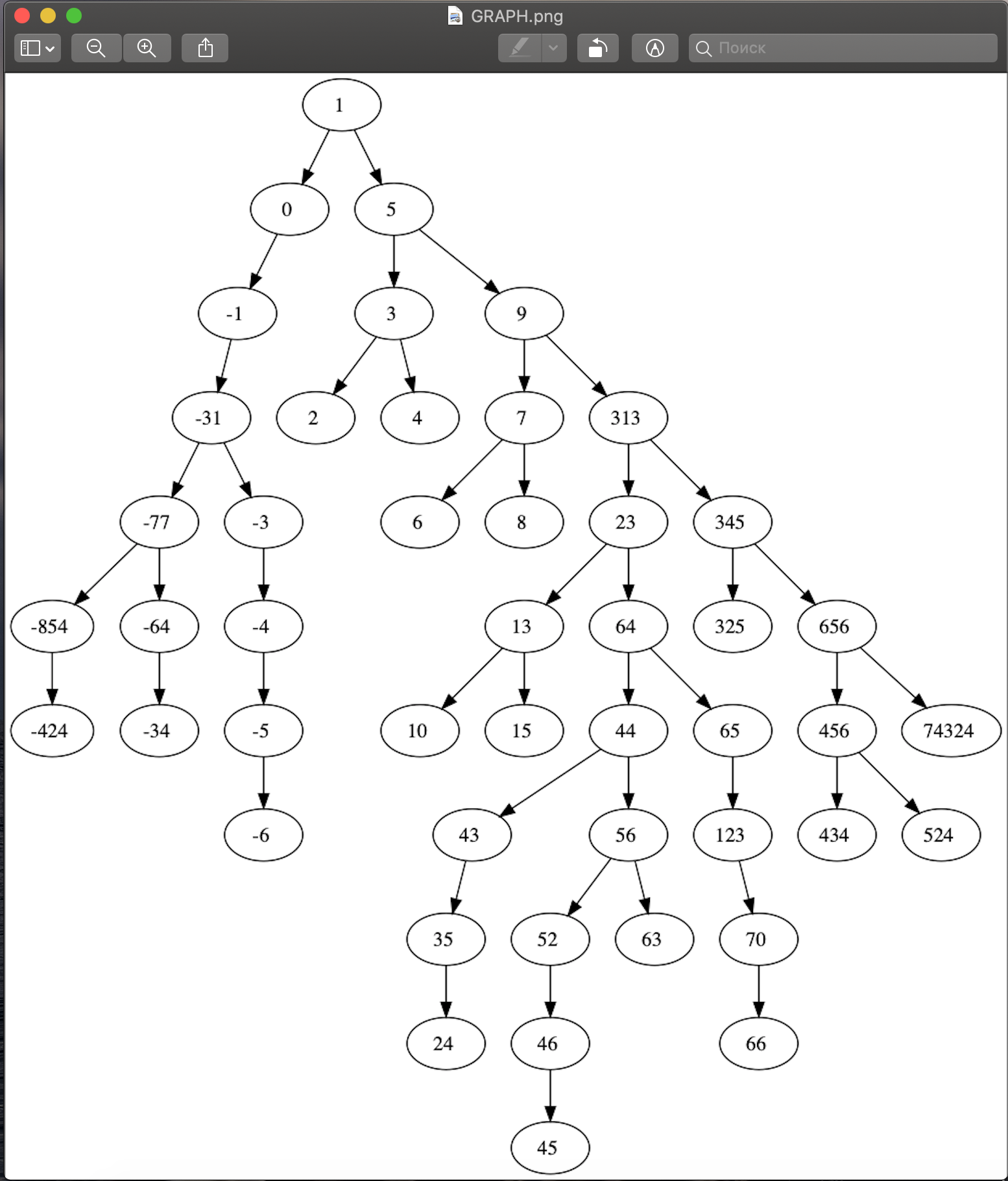
- Ввод неверного пункта меню.

- Вывод на экран хеш-таблицы без её создания.

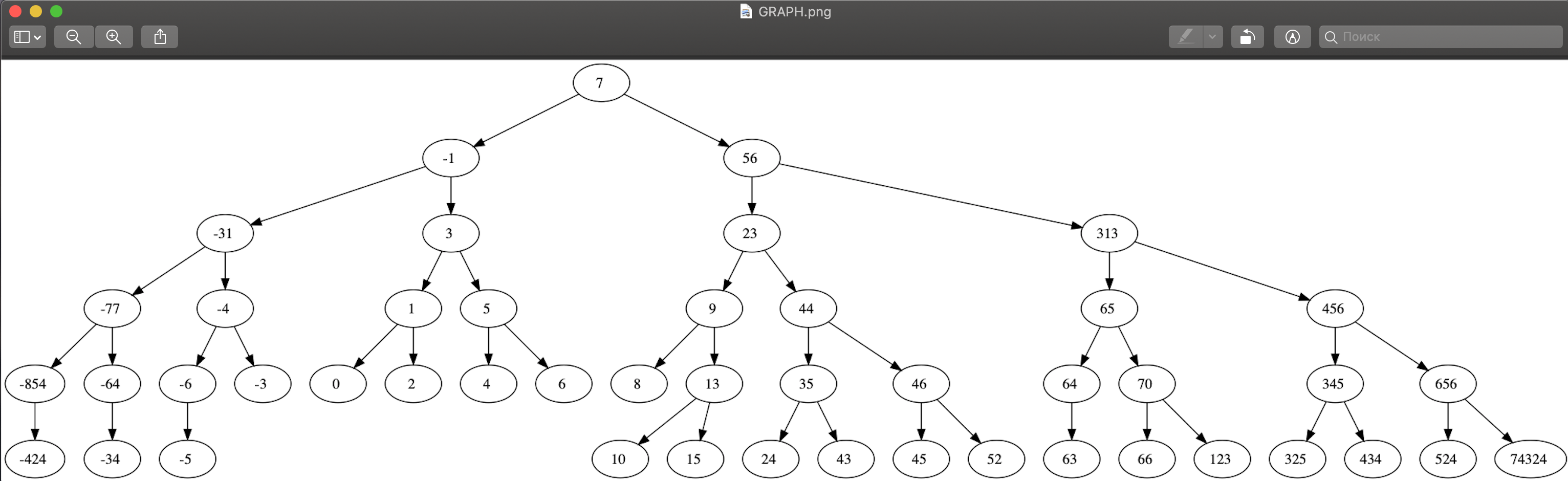
- Поиск числа в хеш-таблице без её создания.

**Демонстрация работы программы:**

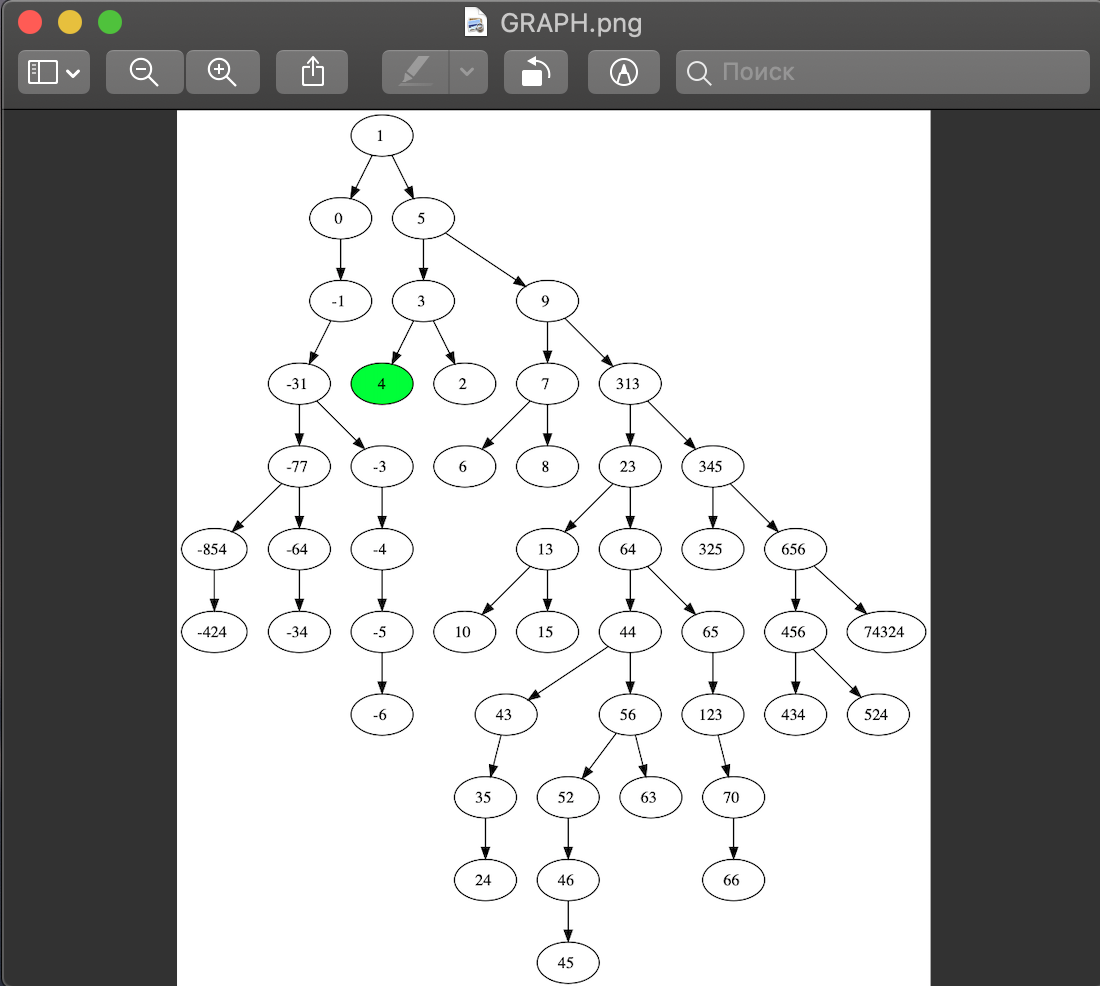
1. **Построить обычное дерево по файлу** (открывается png файл с построенным деревом)



1. **Построить сбалансированное дерево** (открывается png файл с построенным деревом)

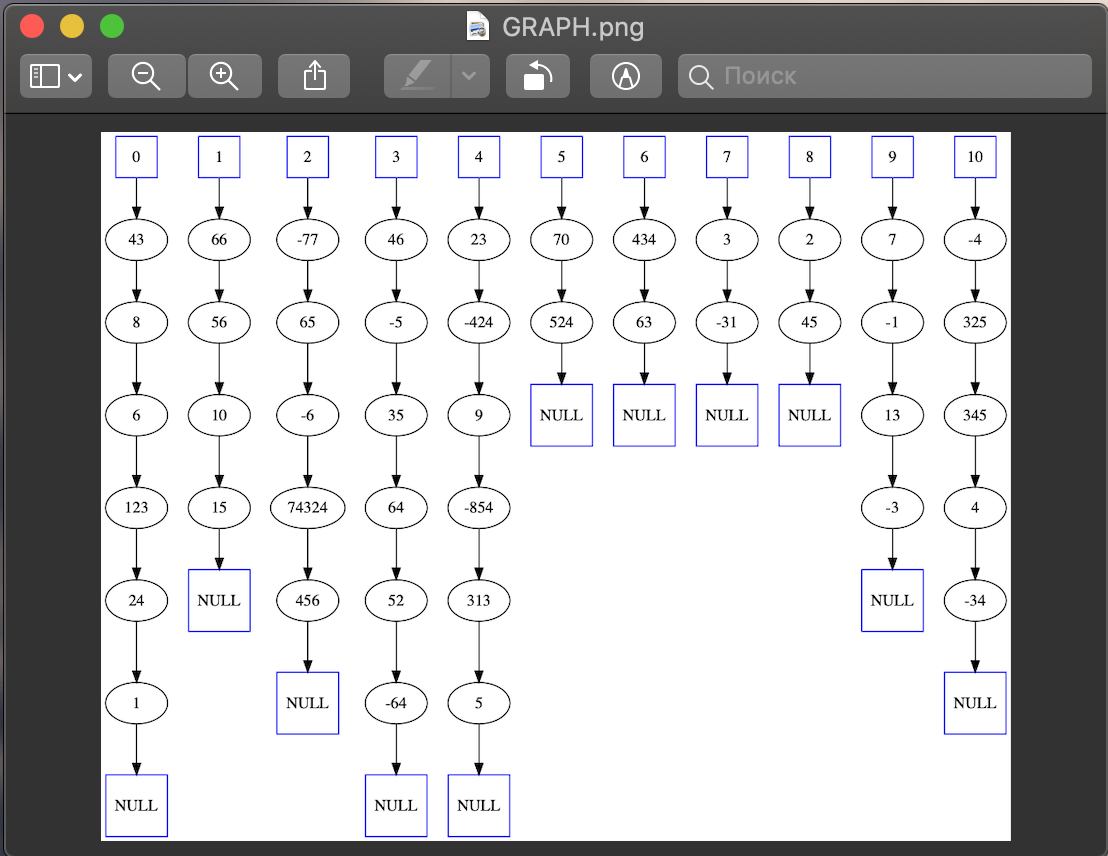
****

1. **Поиск числа в ДДП** - выводится сообщение в терминале и открывается png с найденным элементом.



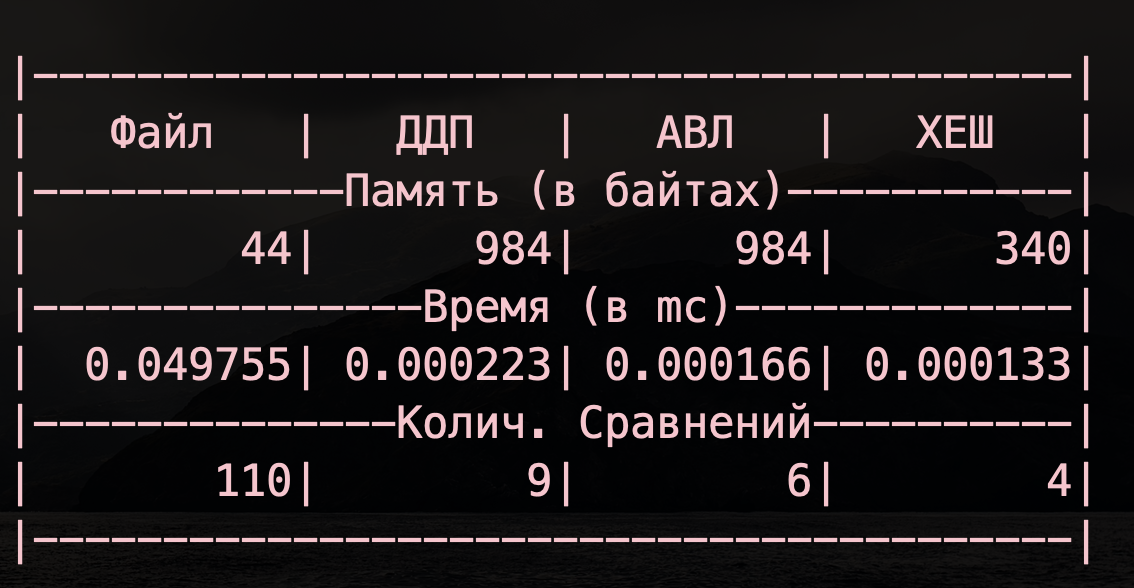
1. **Поиск числа в сбалансированном дереве** - аналогично третьему пункту.

**5, 6) Создание хеш-таблицы** **и вывод ее на экран в виде png**



**7, 8) Поиск числа в хеш-таблице, поиск числа в файле** - Вывод сообщения о нахождении о нахождении числа.

**9) Эффективность.**



Оценка эффективности

Для увеличения точности производился поиск числа, которого нет в файле 1000 раз.

Хеш-таблица наиболее эффективна по всем параметрам за исключением объёма памяти (Файл более эффективен).

ДДП по памяти соответствует АВЛ из-за использования одной и той же структуры данных,

ХЭШ на 65% эффективнее ДДП, АВЛ по памяти.

ХЭШ эффективнее по времени АВЛ на 7%, ДДП на 29%, Файла на 99%.

Для более быстрого поиска в хэш-таблице должно быть как можно меньше коллизий. Количество коллизий зависит от того, сколько было выделено памяти под хеш-таблицу, следовательно, для большей эффективности по времени придется выделить больше памяти. Количество коллизий также зависит от равномерности заполнения хеш-таблицы хеш-функцией.

Ответы на вопросы

1. Что такое дерево?

Дерево – это нелинейная структура данных, используемая для представления иерархических связей, имеющих отношение «один ко многим».

2. Как выделяется память под представление деревьев?

Выделение памяти под деревья определяется типом их представления. Это может быть таблица связей с предками или связный список сыновей. Оба представления можно реализовать в виде матрицы или списка. При реализации списком память выделяется динамически, при реализации матрицей статически.

3. Какие стандартные операции возможны над деревьями?

Основные операции с деревьями: обход дерева, поиск по дереву, включение в дерево, исключение из дерева.

4. Что такое дерево двоичного поиска?

Дерево двоичного поиска – это такое дерево, в котором все левые потомки моложе предка, а все правые – старше.

5. Чем отличается идеально сбалансированное дерево от АВЛ дерева?

Дерево, у которого число вершин в левом и правом поддеревьях отличается не более, чем на единицу называется идеально сбалансированным. Двоичное дерево, у каждого узла которого высота двух поддеревьев отличается не более чем на единицу называется АВЛ-деревом.

6. Чем отличается поиск в АВЛ-дереве от поиска в дереве двоичного поиска?

Поиск в АВЛ дереве имеет сложность О(log2n), в то время как в обычном ДДП сложность О(n).

7. Что такое хеш-таблица, каков принцип ее построения?

Массив, заполненный в порядке, определенным хеш-функцией, называется хеш- таблицей.

Хеш-функция – функция, которая ставит в соответствие каждому ключу индекс ячейки, где расположен элемент с этим ключом.

8. Что такое коллизии? Каковы методы их устранения.

Коллизия – ситуация, когда разным ключам соответствует одно значение хеш-функции. Существует несколько возможных вариантов разрешения коллизий: внешнее (открытое) хеширование (метод цепочек) и внутреннее (закрытое) хеширование (открытая адресация).

9. В каком случае поиск в хеш-таблицах становится неэффективен?

Поиск в хэш-таблице становится неэффективным при большом числе коллизий – сложность поиска возрастает.

При открытом хэшировании в случае, когда элемент таблицы с индексом, который вернула хеш-функция, уже занят, к нему присоединяется связный список. Таким образом, если для нескольких различных значений ключа возвращается одинаковое значение хеш- функции, то по этому адресу находится указатель на связанный список, который содержит все значения. Поиск в этом списке осуществляется простым перебором, так как при грамотном выборе хеш-функции любой из списков оказывается достаточно коротким.

При закрытом хэшировании в этом случае, если ячейка с вычисленным индексом занята, то можно просто просматривать следующие записи таблицы по порядку, до тех пор, пока не будет найден ключ K или пустая позиция в таблице.

10. Эффективность поиска в АВЛ деревьях, в дереве двоичного поиска и в хеш-таблицах.

В хэш-таблице минимальное время поиска О(1). В АВЛ дереве О(log2n). В дереве двоичного поиска О(h), где h – высота дерева.