**Руководство пользователя для класса BinarySearchTree**

**Описание класса**

Класс BinarySearchTree реализует структуру данных «бинарное дерево поиска» (Binary Search Tree, BST) на языке C++. BST — это двоичное дерево поиска, в котором каждый узел имеет ключ и left- и right-указатели на поддеревья, в которых ключи меньше и больше, чем ключ узла, соответственно.

Узлы BST хранятся в памяти в форме объектов класса Node. Каждый узел содержит ключ типа int и два указателя на поддеревья типа Node или NULL, если поддерево отсутствует.

**Методы**

1. Метод add

|  |
| --- |
| void add(int data) |

Добавляет новый узел с указанным ключом в дерево

1. Метод search

|  |
| --- |
| Node\* search(int data) |

Находит узел с указанным ключом в дереве и

возвращает указатель на этот узел.

1. Метод remove

|  |
| --- |
| void remove(int data) |

Удаляет узел с указанным ключом из дерева.

1. Метод print.

|  |
| --- |
| void print(): |

Выводит дерево в консоль в виде дерева со стрелочками.

**Технологии**

* Класс BinarySearchTree написан на языке программирования C++ и использует концепцию объектно-ориентированного программирования (ООП). В задаче реализации узлов дерева было использовано вложенное классовое объявление Node.
* Для обхода дерева используется алгоритмы работы с деревьями в глубину (DFS), которые реализуются рекурсивно.

**Руководство пользователя для класса MultiwayMerge**

Данный класс реализует алгоритм многофазного слияния, который является одним из способов внешней сортировки. Класс позволяет сортировать множество больших файлов, которые не могут поместиться целиком в оперативную память.

**Описание класса**

Конструктор для векторов

|  |
| --- |
| MultiwayMerge(std::vector<std::vector<int>> tapes) |

Конструктор класса MultiwayMerge принимает вектор лент tapes.

Конструктор для работы с файлами

|  |
| --- |
| MultiwayMerge(std::vector<std::string> filepaths, std::string outputFilePath) |

Конструктор класса MultiwayMerge принимает вектор строк filepaths с путями к файлам для сортировки и строку outputFilePath с путем для записи отсортированных данных.

**Методы:**

1. Метод merge

|  |
| --- |
| void merge() |

Метод merge выполняет многофазное слияние и сохраняет отсортированные данные во внутренней переменной класса output.

1. Метод getOutput

|  |
| --- |
| std::vector<int> getOutput() const |

Метод getOutput возвращает отсортированные данные в виде вектора.

1. Метод writeOutputToFile

|  |
| --- |
| void writeOutputToFile(const std::string& filepath) |

Метод writeOutputToFile записывает отсортированные данные в файл с указанным путем filepath.

**Технологии, использованные при создании класса**

Класс MultiwayMerge был написан на языке программирования C++ с использованием стандартной библиотеки языка. Для внутренней сортировки использовался алгоритм сортировки слиянием. Для организации многофазного слияния использовалась куча сортировки (std::priority\_queue).

**Описание класса**

Класс MultiwayMerge представляет собой реализацию алгоритма многофазного слияния. Он позволяет сортировать большие файлы, которые не помещаются целиком в оперативную память.

В конструкторе класса происходит чтение данных из файлов и заполнение лент соответствующими данными. Далее происходит сортировка лент и их последующее слияние. Результат записывается во внутренний вектор output.

Для внутренней сортировки использовался стандартный алгоритм std::sort, а для организации многофазного слияния была использована куча сортировки (std::priority\_queue).

Класс MultiwayMerge позволяет задавать входные и выходные файлы, а также предоставляет методы для чтения и записи данных в файлы. Класс можно использовать для сортировки больших файлов в файловой системе.

**Руководство** **пользователя для класса HashTable**

Хеш-таблицы с методом квадратичного рехэширования — это эффективный способ хранения и управления большим количеством данных.

Класс HashTable реализует хеш-таблицу с использованием метода квадратичного рехэширования. Он предоставляет интерфейс для добавления, удаления, поиска элементов и вывода таблицы на печать в консоль. В этом руководстве пользователя мы рассмотрим, как использовать каждый из этих методов.

**Описание класса**

Конструктор

|  |
| --- |
| HashTable(int size); |

Создает новую хеш-таблицу с заданной размерностью.

Параметр size - размер таблицы.

Деструктор

|  |
| --- |
| ~HashTable(); |

Удаляет хеш-таблицу.

**Методы**

1. Метод add

|  |
| --- |
| void add(int key); |

Добавляет элемент в хеш-таблицу.

Параметр key - ключ, который нужно добавить в таблицу.

1. Метод remove

|  |
| --- |
| void remove(int key); |

Удаляет элемент из хеш-таблицы.

Параметр key - ключ удаляемого элемента.

1. Метод search

|  |
| --- |
| bool search(int key); |

Ищет элемент в хеш-таблице и возвращает true, если элемент найден, и false, если нет.

Параметр key - ключ искомого элемен

1. Метод print

|  |
| --- |
| void print(); |

Выводит содержимое хеш-таблицы в консоль.

**Технологии**

Класс HashTable написан на языке программирования C++. Он использует следующие структуры данных и алгоритмы:

* Стандартный класс vector используется для хранения элементов таблицы.
* Хеш-функция основана на классе hash из библиотеки functional.
* Метод квадратичного рехэширования используется для разрешения коллизий.

**Примечание**

1. Для использования классов подключите к исполняемому файлу заголовочный «includes.h» и сами заголовочные файлы с классами
2. Исходный код классов с комментариями прилагается ниже
3. Исходный код программы для тестирования функционала классов прилагается ниже
4. Ссылка на скачивание исходных файлов (github): <https://github.com/JustGostuY/PractikaTvGU-Files>