Структура дерева кодирования

Node.java:

Этот класс представляет узел в дереве Хаффмана. Он реализует интерфейс **Comparable**, чтобы можно было сравнивать узлы по их частотам. Узел может иметь левого и правого потомка, а также хранит частоту.

Поля:

- **frequency**: Частота узла (или сумма частот для узла-родителя).
- **left**: Левый потомок узла.
- **right**: Правый потомок узла.

Методы:

- **compareTo**(Node n): Реализует интерфейс **Comparable** для сравнения узлов по их частотам.
- **fillCodeMap**(String character, HashMap<Byte, String> codeMap): Рекурсивно заполняет карту кодирования символов в дереве Хаффмана.

Leaf.java:

Этот класс представляет лист в дереве Хаффмана. Он наследуется от **Node** и добавляет символ и префикс. Класс также имеет метод **fillCodeMap**, который заполняет карту кодирования для символа.

Поля:

- **symbol**: Символ, представленный листом.
- **prefix**: Префикс (бинарный код) символа.

Методы:

- **toString**(): Возвращает строковое представление листа с символом и его префиксом.
- **fillCodeMap**(String character, HashMap<Byte, String> codeMap): Заполняет карту кодирования для символа.

Huffman.java:

Этот класс реализует алгоритм Хаффмана. Он содержит методы для кодирования и декодирования данных. Внутри класса используется приоритетная очередь для построения дерева Хаффмана. Метод **encode** принимает массив байт и возвращает сжатые данные в виде массива байт. Метод **decode** принимает сжатые данные и карту восстановления и возвращает исходные данные.

Поля:

- **codeMap**: Хранит соответствие байта (символа) его бинарному коду в виде строки.

Методы:

- **encode**(byte[] data): Сжимает исходные данные и возвращает массив байт с сжатыми данными.
- **decode**(String compressed, HashMap<String, Byte> recoveryMap): Декодирует сжатые данные и восстанавливает исходные.
- **padWithZeros**(String compressed): Добавляет нули в конец строки сжатых данных для выравнивания по байтам.

- **buildHuffmanTree**(PriorityQueue<Node> queue): Строит дерево Хаффмана из приоритетной очереди узлов.
- **buildWeightQueue**(HashMap<Byte, Integer> map): Строит приоритетную очередь узлов на основе карты весов символов.
- **calculateWeightMap**(byte[] data): Вычисляет карту весов символов в исходных данных.
- **compressData**(byte[] data): Преобразует исходные данные в их сжатое представление в виде строки бинарных кодов.
- **convertToByteArray**(String compressed): Преобразует строку бинарных кодов в массив байт.
- **convertListToArray**(ArrayList<Byte> list): Преобразует список байт в массив байт.

Main.java:

Этот класс содержит метод **main** и представляет точку входа в программу. В зависимости от переданной команды (**encode**, **decode**, **inform**), программа выполняет соответствующие действия с использованием классов **Huffman**, **DataIOManager** и **CompressionInfo**.

DataIOManager.java:

Этот класс предоставляет методы для работы с вводом/выводом данных. Он содержит методы для чтения заголовка архива, чтения данных из архива, чтения файла, записи архива и записи файла. Также есть метод для тестирования, который выводит информацию о размере архива, исходном файле и отображает карту кодирования.

Методы:

- **readArchiveHeader**(String filePath, CompressionInfo compressionInfo): Читает заголовок архива и заполняет информацию о сжатых данных в объекте CompressionInfo.
- **getAdditionalZerosNumber**(String compressed): Возвращает количество дополнительных нулей в сжатых данных.
- **readArchiveData**(String filePath, CompressionInfo compressionInfo): Читает сжатые данные из архива и заполняет информацию о сжатых данных в объекте CompressionInfo.
- **readArchive**(String filePath): Читает архив и возвращает объект CompressionInfo с информацией о сжатых данных.
- **readFile**(String filePath): Читает содержимое файла и возвращает массив байт.
- **writeArchive**(byte[] result, HashMap<Byte, String> codeMap, String originalFileName, String outputFilePath): Записывает сжатые данные и информацию о кодировании в файл архива.
- **writeFile**(byte[] result, String outputFilePath): Записывает массив байт в файл.
- **test**(CompressionInfo compressionInfo, byte[] result): Выводит информацию о сжатии, включая размер оригинального файла, размер архива, соотношение сжатия и карту кодирования.

CompressionInfo.java:

Этот класс представляет информацию о сжатых данных. Он содержит сжатые данные, оригинальное имя файла, карту восстановления (раскодирования), оригинальную длину данных и смещение данных в файле.

Поля:

- **compressedData**: Сжатые данные в виде строки бинарных кодов.
- **originalFileName**: Оригинальное имя файла.
- **recoveryMap**: Карта восстановления (раскодирования) бинарных кодов в символы.
- **originalLength**: Оригинальная длина данных.
- **dataOffset**: Смещение данных в файле архива.