**Санкт-Петербургский политехнический университет Петра Великого**

**Институт компьютерных наук и кибербезопасности**

**Высшая школа программной инженерии**

**Изображение выглядит как текст, Шрифт, логотип, Графика

Автоматически созданное описание**

**К У Р С О В А Я Р А Б О Т А**

**Методы разработки алгоритмов**

по дисциплине «**Алгоритмы и структуры данных**»

Выполнил Петров Константин

студент гр. 5130904/30003 Михайлович

Руководитель Шемякин Илья

Александрович

«22» Сентября 2024

Санкт-Петербург

2024 г

1. Введение. Общая постановка задачи:

Тема: Методы разработки алгоритмов

Вариант 3.2

Коды Шеннона-Фано.

Реализовать алгоритм кодирования и раскодирования текста по алгоритму Шеннона-Фано.

СОДЕРЖАНИЕ

**ОСНОВНАЯ ЧАСТЬ РАБОТЫ**

1. **Описание алгоритма решения и используемых структур данных**

Кодирование текста состоит из нескольких этапов:

1. Считывание текста в первый раз
2. Создание кодов для символов
3. Считывание текста во второй раз и его кодирование

Раскодирование текста тоже состоит из нескольких этапов:

1. Считывание кодов из файла
2. Считывание зашифрованного текста и его раскодирование

Для создания кодов используется алгоритм Шаннона-Фано, который можно описать как:

1. Создание списка букв, отсортированного по уменьшению частоты их нахождения в тексте
2. Рекурсивное разделение списка на две части так чтобы суммы частот до разделения и после были примерно равны
3. Так образуется дерево, где более частые буквы имеют более короткий код, а значит занимают меньше места на носителе.

Для хранения букв и их кодов используется структура данных бинарное дерево.

1. **Анализ алгоритма**

N - Количество символов в файле

K - Количество различных символов в файле

**1. read:** O(N)

**2. decode:**

Считывание дерева кодов:

O(K)

Чтение, декодирование и запись:

O(N \* K)

Суммарная сложность: O(N \* K)

**3. encode:**

Создание дерева кодов:

O(N \* K)

Чтение, кодирование и запись:

O(K \* N)

Суммарная сложность: O(N \* K)

**4. makeCodes:**

составление начального списка

O(N \* K)

mergeSort

O(K \* log(K))

listToTree

O(K \* log(K))

Суммарная сложность: O(N \* K)

1. **Описание спецификации программы (детальные требования)**

**Система команд**

1. Команда read <file>

Обязательные параметры: <file>

Выводит содержимое <file>

1. Команда decode <binary> <codes> <text>

Обязательные параметры: <binary> <codes>

Раскодирует <binary> с помощью <codes>

Записывает в файл <text>, если этот аргумент не был передан то

Записывает в файл с таким же названием как <binary>, но с приставкой «text»

1. Команда encode <text> <binary> <codes>

Обязательные параметры: <text>

Кодирует файл <text> и записывает созданные коды в <codes>

Результат кодирования записывает в <binary>

Если не указан <codes>, то записывает в файл с названием <text> с приставкой «codes».

Если не указан <binary>, то записывает в файл с названием <text> с приставкой «binary».

1. Команда makeCodes <text> <codes>

Обязательные параметры: <text>

Создает коды для файла <text> и записывает их в <codes>.

Если <codes> не указан то создает файл с названием <text> но с приставкой «codes».

1. **Описание программы**

Структура программы:

Main

* runCommandLoop – читает команды
* read – выводит содержимое файла
* decode
  + CodeTree::CodeTree(потокВвода, флагФайла) - создает кодовое дерево с помощью файла кодов
  + BinaryWriter::read – считывает биты из файла и представляет их как буквы с помощью кодового дерева
* encode – создает кодовое дерево с помощью анализа текста
  + CodeTree::CodeTree(потокВвода)
  + BinarzWriter::write – записывает буквы в файл в виде битов с помощью кодового дерево
* makeCodes
  + CodeTree::CodeTree(потокВвода)

Входные данные:

Выходные данные:

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЕ 1. ТЕКСТ ПРОГРАММЫ

ПРИЛОЖЕНИЕ 2. ПРОТОКОЛЫ ОТЛАДКИ