**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

по дисциплине **«Алгоритмы и Структуры Данных»**

# на тему: «Архивация данных»

Выполнил: студент гр. ИП-31

Лепший Е.А.

Принял: ст. преподаватель

Косинов Г.П.

Дата сдачи отчета: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата допуска к защите: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата защиты: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Гомель 2020

# Лабораторная работа № 6

**Тема работы:** Архивация данных

**Цель работы:** Изучить основные методы архивации, сравнить их эффективность

**Теоретические сведения по изучению темы**

Приведены в соответствующем лекционном материале

**Индивидуальные задания**

Реализовать алгоритмы сжатия согласно варианту. Определить коэффициенты сжатия и время архивации.

|  |  |  |  |
| --- | --- | --- | --- |
| Вариант + доп требования | | Алгоритм 1 | Алгоритм 2 |
| 12 | С разбиением на тома | KWE BWT | Хаффмена |

**Листинг программы**

**package org.fekz115.labs6.ands**

**import HuffmanCompressor**

**import org.apache.commons.lang3.SerializationUtils**

**import java.io.File**

**import java.io.FileOutputStream**

**import java.io.PrintStream**

**import java.util.\***

**fun main() {**

**val scanner = Scanner(System.`in`)**

**println("Enter file name:")**

**val fileName = scanner.nextLine()**

**println("Enter compress algorithm:")**

**println("1. Huffman")**

**println("2. BWT KWE")**

**println("3. LZW")**

**val compressor = when(scanner.nextInt()) {**

**1 -> HuffmanCompressor()**

**2 -> BwtKweCompressor()**

**else -> LZW()**

**}**

**scanner.nextLine()**

**println("Enter password:")**

**var password = scanner.nextLine()**

**println("Enter archive name:")**

**val archive = scanner.nextLine()**

**println("Enter symbols count")**

**val count = scanner.nextInt()**

**var length = 0**

**var startTime = System.currentTimeMillis()**

**var compressedLength = 0**

**readFile(fileName)**

**.chunked(count)**

**.map { length+=it.length; compressor.compress(it) }**

**.map { Encoder(it, password).encode() }**

**.map { SerializationUtils.serialize(it) }**

**.forEachIndexed { index, bytes ->**

**compressedLength+=bytes.size**

**val fos = FileOutputStream(File("${archive}$index"))**

**fos.write(bytes)**

**}**

**println("Compressing: ${length / compressedLength.toFloat()}")**

**println("Time: ${System.currentTimeMillis() - startTime} ms")**

**println("Enter file to decompress")**

**scanner.nextLine()**

**val compressed = scanner.nextLine()**

**println("Enter file to save")**

**val output = scanner.nextLine()**

**println("Enter password:")**

**password = scanner.nextLine()**

**println("Enter compress algorithm:")**

**println("1. Huffman")**

**println("2. BWT KWE")**

**println("3. LZW")**

**val extractor = when(scanner.nextInt()) {**

**1 -> HuffmanCompressor()**

**2 -> BwtKweCompressor()**

**else -> LZW()**

**}**

**startTime = System.currentTimeMillis()**

**readEncoded(compressed)**

**.map { SerializationUtils.deserialize<Encoder.Encoded>(it) }**

**.map { Encoder.decode(it, password) }**

**.map {**

**extractor.decompress(it)**

**}**

**.joinToString("")**

**.apply {**

**PrintStream(FileOutputStream(File(output))).println(this)**

**}**

**println("Time: ${System.currentTimeMillis() - startTime} ms")**

**}**

**Вывод**

Были изучены основные методы архивации.