МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

Специальность Информационные системы и технологии

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №10 НА ТЕМУ:**

**Сжатие/распаковка данных методом Лемпеля – Зива**

Выполнил студент 3 курса 1 группы

Халалеенко Андрей Николаевич

Минск 2024

**Задание 1:** Разработать авторское приложение в соответствии с целью лабораторной работы. При этом предусмотреть возможность оперативного изменения размеров окон (*n1*, *n2*).

Для выполнения этого задания был использован следующий код на языке *python*:

dictionary\_size = int(input('Длина буфера словаря:'))

lookahead\_size = int(input('Длина буфера кодирования:'))

Листинг 1 – Задание 1

**Задание 2:** С помощью приложения выполнить прямое и обратное преобразования произвольного текста длиной несколько килобайт. Формат представления параметров *p* и *q* выбрать по указанию преподавателя.

Для выполнения этого задания был использован следующий код на языке *python*:

def max\_substring(buffer, dictionary):

    shift = len(dictionary)

    substring = ''

    for character in buffer:

        substring\_temp = substring + character

        shift\_temp = dictionary.rfind(substring\_temp)

        if shift\_temp < 0:

            break

        substring = substring\_temp

        shift = shift\_temp

    return len(substring), len(dictionary) - shift

def compress\_lz77(message, lookahead\_size, dictionary\_size):

    dictionary = ''

    lookahead = message[:lookahead\_size]

    triads = []

    while len(lookahead) != 0:

        match\_length, offset = max\_substring(lookahead, dictionary)

        dictionary += message[:match\_length + 1]

        dictionary = dictionary[-dictionary\_size:]

        message = message[match\_length:]

        last\_character = message[:1]

        message = message[1:]

        lookahead = message[:lookahead\_size]

        triads.append((offset, match\_length, last\_character))

    return triads

def decompress\_lz77(compressed):

    message = ''

    for part in compressed:

        offset, match\_length, character = part

        message = message + message[-offset:][:match\_length] + character

    return message

Листинг 2 – Задание 2

**Задание 3:** Изменяя размеры окон, оценить скорость и эффективность (используя соотношения на с. 76) выполнения операций сжатия/ распаковки.

Для выполнения этого задания код, разработанный в предыдущем задании был дополнен следующим кодом:

def get\_compressed\_size(compressed):

    triads\_str = ''

    for part in compressed:

        triads\_str += str(part[0]) + str(part[1]) + str(part[2])

    return len(triads\_str)

dictionary\_size = 10

lookahead\_size = 5

while dictionary\_size < 10240:

    print('\nДлина буфера словаря:', dictionary\_size)

    print('Длина буфера кодирования:', lookahead\_size)

    start\_time = datetime.now()

    compressed = compress\_lz77(stringToEncode, dictionary\_size, lookahead\_size)

    compress\_time = datetime.now() - start\_time

    cipher\_size = get\_compressed\_size(compressed)

    ratio = cipher\_size / len(stringToEncode)

    print('Эффективность:', ratio)

    start\_time = datetime.now()

    decompress\_lz77(compressed)

    decompress\_time = datetime.now() - start\_time

    print('Время компрессии:', compress\_time)

    print('Время декомпрессии:', decompress\_time)

    dictionary\_size \*= 2

    lookahead\_size \*= 5

Листинг 3 – Задание 3

Результат работы итогового кода:

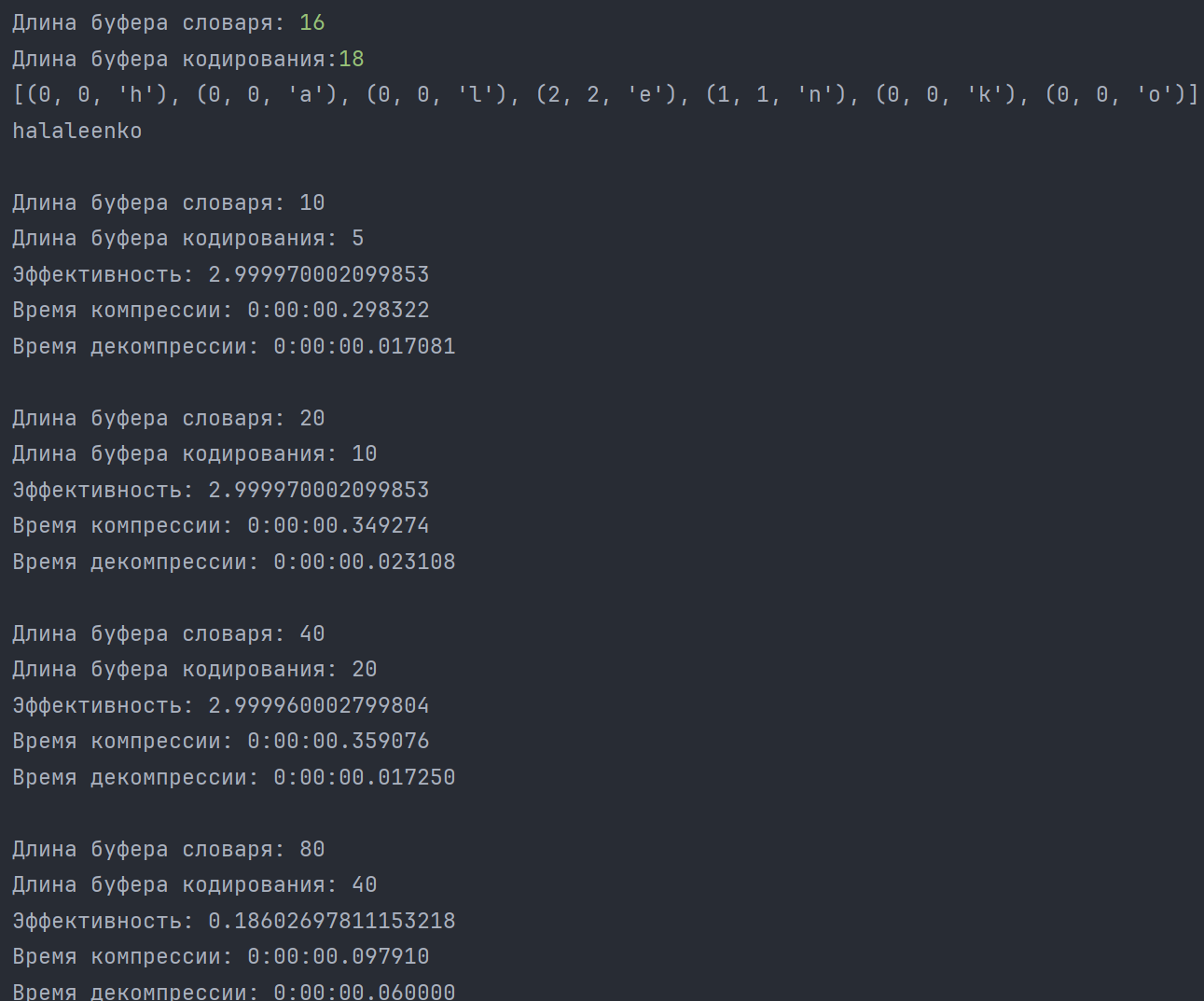


Рисунок 1 ­­­­– Результат работы программы

**Вывод:** в ходе лабораторной работы были приобретены практические навыки использования метода Лемпеля – Зива для сжатия/распаковки данных. Было выяснено, что при увеличении буферов понижается эффективность сжатия, увеличивается время, необходимое для компрессии.