ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ _____

ПРЕПОДАВАТЕЛЬ

Доцент, канд. техн. наук		В.А. Миклуш
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3

Словарно-ориентированные методы кодирования

по курсу: Теория информации, данные, знания

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4329		Д.С. Шаповалова
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2025

1. Цель работы:

Изучение словарно-ориентированных методов кодирования (LZ77, LZ78, LZW).

2. Задание:

В соответствии с вариантом:

1. Построить алгоритм (блок-схему)
2. Написать программу, реализующую заданный метод кодирования
3. Провести ручную трассировку
4. Сравнить полученные результаты между собой
5. Рассчитать среднее число элементарных сигналов

3. Исходные данные

Вариант 23:

Текст: косарь касьян косой косит косо. не скосит косарь касьян покоса

Метод кодирования: LZ78

4. Теоретические сведения:

Алгоритм LZ78:

При старте алгоритма этот словарь содержит одну пустую строку длины ноль. Алгоритм считывает символы сообщения до тех пор, пока накапливаемая строка входит целиком в одну из фраз словаря. Как только эта строка перестанет соответствовать хотя бы одной фразе словаря, алгоритм генерирует код, состоящий из индекса строки в словаре, которая до последнего введенного символа содержала входную строку, и символа, нарушившего совпадение. Затем в словарь добавляется введенная строка. Если словарь уже заполнен, то из него предварительно удаляют менее всех используемую в сравнениях фразу.

Длина полученного двоичного кода будет округленным в большую сторону:

$$L_{\text{к}} = log(\text{размер словаря}) + 8\,, \tag{1}$$

(8 битами кодируются символы, например, ASCII+)

Расчет среднего числа элементарных сигналов:

Среднее число элементарных сигналов на один входной символ:

$$N_{\text{ср}} = \frac{\text{Длина исходного текста (в символах)}}{\text{Число фраз в выходе } LZ78}, \tag{2}$$

Средняя длина одной фразы в символах (обратная величина):

$$L_{\text{ср}} = \frac{\text{Число фраз}}{\text{Длина исходного текста}}, \tag{3}$$

4. Ход работы:

Для выполнения работы был выбран язык программирования высокого уровня Python. Была разработана программа (Приложение А), реализующая алгоритм LZ78, кодирующая поданную на вход последовательность.

Для наглядности работы программы и алгоритма были сделаны блок схемы (рисунки 1-2):

Рисунок 1 – Блок схема основной программы

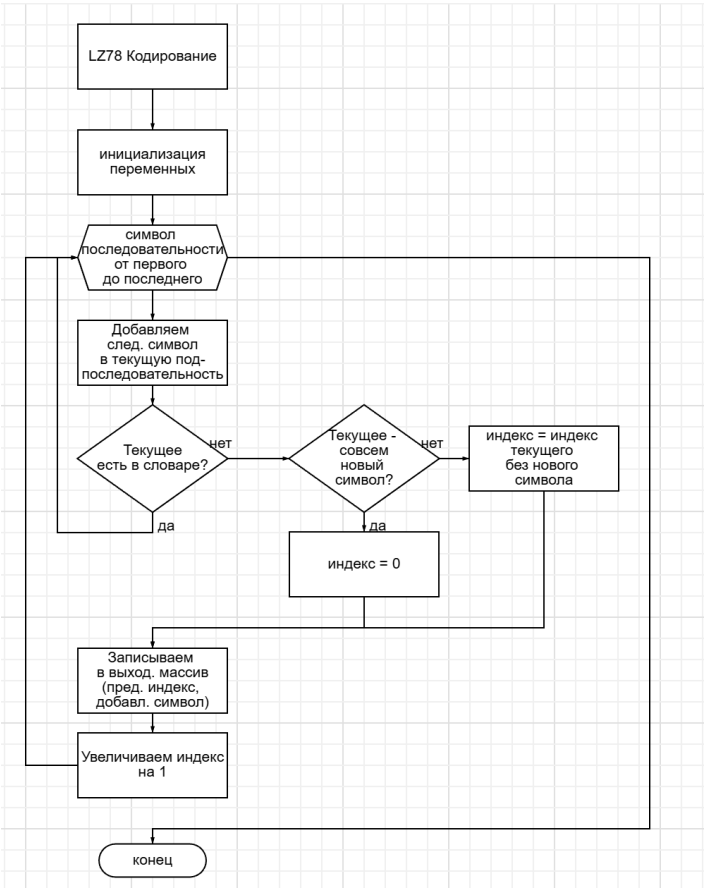


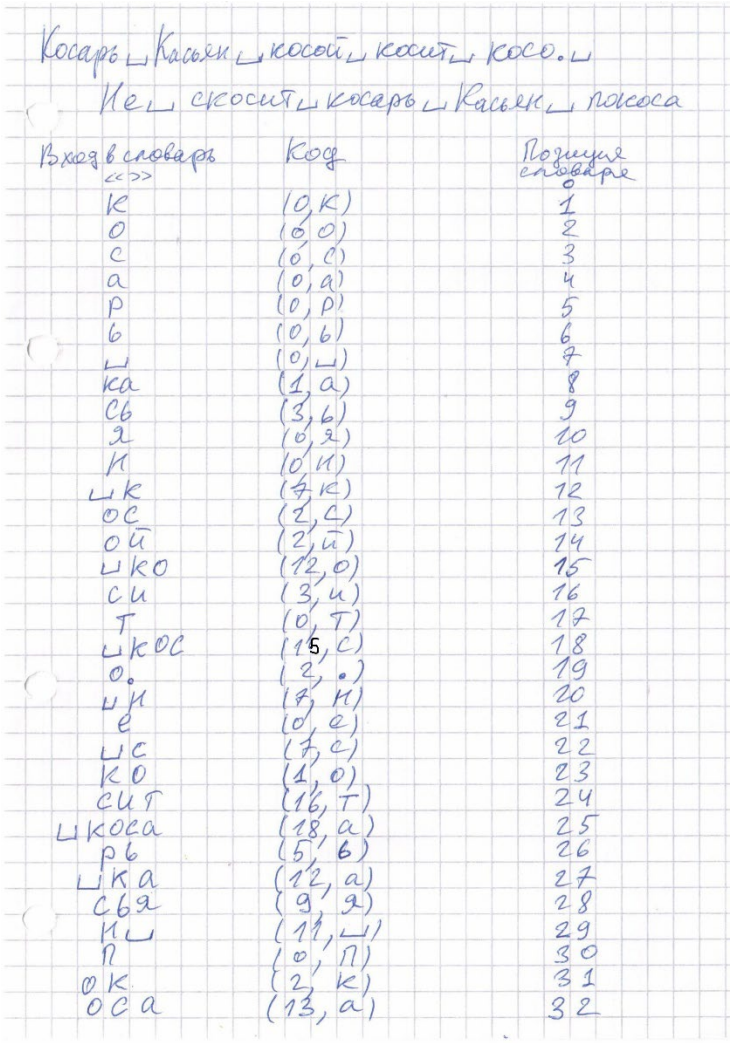
Рисунок 2 – Блок схема подпрограммы – алгоритма LZ78

Результат кодирования текста программой представлен на рисунке 3:

```
(0, 'к')
(0, 'о')
(0, 'с')
(0, 'а')
(0, 'р')
(0, 'ь')
(0, ' ')
(1, 'а')
(3, 'ь')
(0, 'я')
(0, 'н')
(7, 'к')
(2, 'с')
(2, 'й')
(12, 'о')
(3, 'и')
(0, 'т')
(15, 'с')
(2, '.')
(7, 'н')
(0, 'е')
(7, 'с')
(1, 'о')
(16, 'т')
(18, 'а')
(5, 'ь')
(12, 'а')
(9, 'я')
(11, ' ')
(0, 'п')
(2, 'к')
(13, 'а')
```

Рисунок 3 – Закодированная последовательность

Для проверки работы написанной программы и правильности результата была выполнена ручная трассировка, представленная на рисунке 4:


Косарь␣Касьян␣косой␣косит␣косо.␣
Не␣скосит␣косарь␣Касьян␣покоса

Вход в словарь << >>	Код	Позиция словаря
		0
К	(0,К)	1
о	(0,о)	2
с	(0,с)	3
а	(0,а)	4
р	(0,р)	5
ь	(0,ь)	6
␣	(0,␣)	7
ка	(1,а)	8
сь	(3,ь)	9
я	(0,я)	10
и	(0,и)	11
␣к	(7,к)	12
ос	(2,с)	13
ой	(2,й)	14
␣ко	(12,о)	15
си	(3,и)	16
т	(0,т)	17
␣кос	(15,с)	18
о.	(2,.)	19
␣и	(8,и)	20
е	(0,е)	21
␣с	(7,с)	22
ко	(1,о)	23
сит	(16,т)	24
␣коса	(18,а)	25
рь	(5,ь)	26
␣ка	(12,а)	27
сья	(9,я)	28
и␣	(11,␣)	29
п	(0,п)	30
ок	(2,к)	31
оса	(13,а)	32

Рисунок 4 – Ручная трассировка LZ78

Как видно, расхождения в результатах отсутствуют, что подтверждает правильность работы написанной функции.

Расчёт среднего числа элементарных сигналов:

Входных символов: 62

Элементарных сигналов (фраз): 32

Среднее число сигналов на символ: $N_{\text{ср}} = 0.5161$ по формуле (1)

Средняя длина фразы: $L_{\text{ср}} = 1.94$ символов по формуле (2)

Длина двоичного кода: $L_{\text{к}} = 32 * (\log(33) + 8) = 32 * (6 + 8) = 448$ бит по формуле (3)

Вывод

В ходе выполнения лабораторной работы был изучен метод кодирования текстовой информации алгоритмом LZ78. Была написана программа, реализующая кодирование по методу LZ78, а также для проверки её работы была проведена ручная трассировка и сравнение результатов: почти идентичны. Также была рассчитана длина двоичного кода закодированной последовательности = 448 бит.

ПРИЛОЖЕНИЕ А

Листинг программы

```
def lz78_encode(data):
    dictionary = {}         # словарь: {фраза: индекс}
    phrases = []            # результат: список кортежей (индекс, символ)
    current = ""
    index = 1               # индексы начинаются с 1

    for char in data:
        current_with_char = current + char
        if current_with_char in dictionary:
            current = current_with_char
        else:
            # current есть в словаре (или пуст), char — новый символ
            prev_index = dictionary[current] if current != "" else 0
            phrases.append((prev_index, char))
            dictionary[current_with_char] = index
            index += 1
            current = ""

    return phrases


def read_input(filename):
    with open(filename, 'r', encoding='utf-8') as f:
        return f.read()


def write_output(phrases, filename='output.txt'):
    with open(filename, 'w', encoding='utf-8') as f:
        for idx, char in phrases:
            # Записываем в формате: (индекс, символ)
            f.write(f"({idx}, '{char}')\n")


if __name__ == '__main__':
    data = read_input('text.txt')
    encoded = lz78_encode(data)
    write_output(encoded, 'output.txt')
    print("Сжатие LZ78 завершено. Результат в output.txt")
```