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 № 2

МЕТОДЫ КОДИРОВАНИЯ. КОДЫ ШЕННОНА-ФАНО, ХАФФМАНА

по курсу:

ТЕОРИЯ ИНФОРМАЦИИ, ДАННЫЕ, ЗНАНИЯ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. № 4326		Г. С. Томчук
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2025

1 Цель работы

Цель работы: изучение методов статистического кодирования, алгоритмов Шеннона-Фано, Хаффмана.

2 Краткое описание задания

В соответствии с вариантом, таблица 1, построить дерево, представить алгоритм (блок-схему) и написать программу, реализующую заданный метод кодирования. Провести ручную трассировку и сравнить полученные результаты между собой. Рассчитать среднее число элементарных сигналов.

3 Вариант задания

В таблице 1 представлен вариант задания.

Таблица 1 — Вариант работы

№ п/п	Метод кодирования	Алфавит	Текстовое сообщение
17.	Шеннона-Фано	Итальянский	Si mangia per vivere, non si vive per mangiare

4 Ход работы

4.1 Определение частот появления символов

В таблице 2 приведены данные по частотному распределению символов в заданном сообщении.

Таблица 2 — Частоты символов

Символ	**Частота**	**Вероятность**
пробел	8	0.1739
e	6	0.1304
i	6	0.1304
a	4	0.0870
n	4	0.0870
r	4	0.0870
v	4	0.0870
g	2	0.0435
m	2	0.0435
p	2	0.0435
s	2	0.0435
,	1	0.0217
o	1	0.0217

4.2 Построение кодов

В таблице 3 отображена произведенная вручную трассировка алгоритма разбиения символов на подгруппы и построения кодов.

Таблица 3 — Частоты символов

Символ	Вероятность	Разбиение на подгруппы						Кодовое обозначение
пробел	0.1739	I	I	I				000
e	0.1304			II				001
i	0.1304		II	I				010
a	0.0870			II				011
n	0.0870	II	I	I				100
r	0.0870			II	I			1010
v	0.0870				II			1011
g	0.0435		II	I	I			1100
m	0.0435				II			1101
p	0.0435			II	I			1110
s	0.0435				II	I		11110
,	0.0217					II	I	111110
o	0.0217						II	111111

На рисунке 1 изображено получившееся дерево Шеннона-Фано.

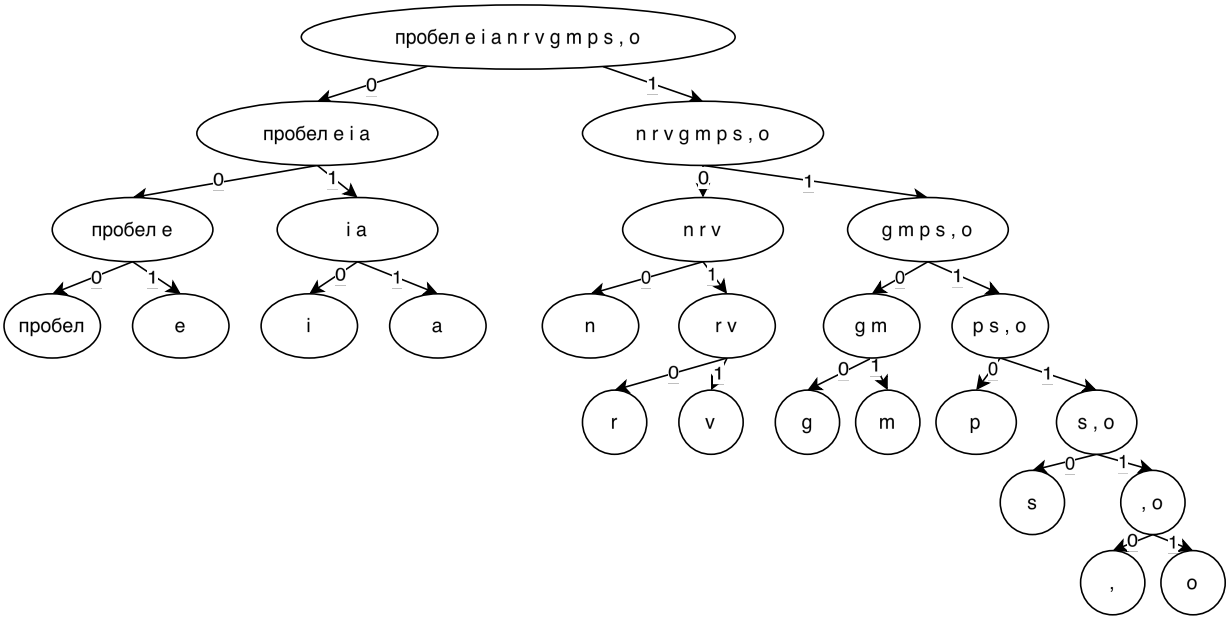

Рисунок 1 — Дерево Шеннона-Фано

4.3 Среднее число элементарных сигналов

Среднее число элементарных сигналов вычислим по формуле:

$$L = \sum p_i \cdot l_i, \quad (1)$$

где p_i — вероятность появления символа, l_i — длина кода символа. Подставим полученные данные в (1):

$$L \approx 3.5217 \text{ (бит на символ)}$$

4.4 Алгоритм и реализация программы

На рисунке 2 изображена блок-схема алгоритма кодирования.

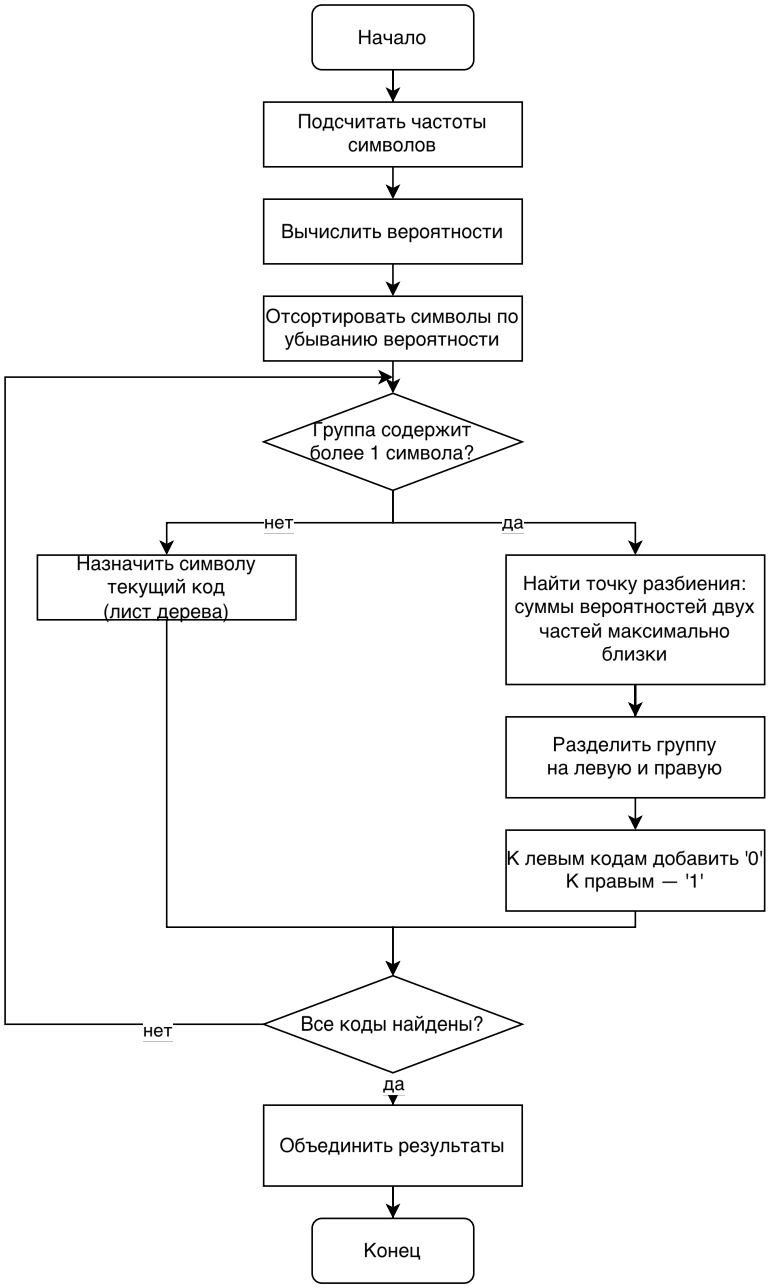

Рисунок 2 — Блок-схема

В Приложении А приведен листинг реализованной программы на языке Python.

5 Вывод

В результате работы были изучены статистические методы кодирования и, в частности, метод Шеннона—Фано. Для варианта 17 посчитаны частоты символов, построено дерево методом Шеннона—Фано и получены кодовые слова. Средняя длина кода составила $L \approx 3.5217$ бит на символ, а весь текст кодируется примерно в 162 бита.

Код является префиксным. Алгоритм прост в реализации, но не гарантирует глобально оптимального результата (в отличие от метода Хаффмана в общем случае).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Миклуш, В.А. Основы теории информации: Учебно-методическое пособие / В.А. Миклуш, В.А. Ушаков. — СПб: ГУАП, 2024. — 41 с.
2. Шеннон, К. Э. Работы по теории информации и кибернетике / К. Э. Шеннон. — М.: Иностранная литература, 1963. — 830 с.
3. Колодуб, В. Д. Теория информации: учебник для вузов / В. Д. Колодуб. — М.: ФОРУМ, 2019. — 352 с.
4. Ковалёв, В. А. Теория информации и кодирование: учебное пособие / В. А. Ковалёв. — СПб.: Питер, 2020. — 368 с.

ПРИЛОЖЕНИЕ А

```
from collections import Counter


def shannon_fano(sorted_items):
    codes = {sym: "" for sym, _ in sorted_items}
    splits = []

    def split_group(items):
        if len(items) <= 1:
            return
        probs = [p for _, p in items]
        total_p = sum(probs)
        # ищем лучший индекс разбиения
        best_idx = 1
        best_diff = abs(sum(probs[:1]) - (total_p - sum(probs[:1])))
        for i in range(2, len(items)):
            left_sum = sum(probs[:i])
            diff = abs(left_sum - (total_p - left_sum))
            # ищем такое разбиение, чтобы суммы вероятностей слева и справа были
максимально близкими
            if diff < best_diff:
                best_diff = diff
                best_idx = i
        left = items[:best_idx]
        right = items[best_idx:]
        for sym, _ in left:
            codes[sym] += "0"
        for sym, _ in right:
            codes[sym] += "1"
        splits.append((items, left, right))
        split_group(left)
        split_group(right)

    split_group(sorted_items)
    return codes, splits


text = "Si mangia per vivere, non si vive per mangiare"
text = text.lower()

# частоты
freq = Counter(text)
total = sum(freq.values())

# сортируем по убыванию вероятности
items = sorted([(ch, freq[ch] / total) for ch in freq], key=lambda x: (-x[1], x[0]))

codes, splits = shannon_fano(items)

# вывод
print("Символ  Кол-во  p       Код       Длина")
for ch, p in items:
    print(f"{repr(ch):6}  {freq[ch]:6d}  {p:.4f}  {codes[ch]:8}  {len(codes[ch]):d}")

avg_len = sum(p * len(codes[ch]) for ch, p in items)
print("L =", avg_len)
print("Всего бит:", round(total * avg_len))
```