Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ПРАКТИЧЕСКОЙ РАБОТЕ №7
дисциплины «Алгоритмизация»

Выполнил:
Мотовилов Вадим Борисович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Информатика и вычислительная
техника», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Роман Александрович
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Порядок выполнения работы:

1. Написал программу для кодирования и декодирования текста при помощи кодировки Хаффмана. В коде есть следующие функции: print tree() отрисовывает дерево в терминале; procedurehuffman(создает дерево хаффмана на основе словаря, в котором каждому символу присвоена его частота использования в предложении; codecreate(создает каждому символу определенный код возвращает созданное в виде словаря; replace sentence заменяет в предложении каждый символ на ранее созданные коды; huffman decode(с помощью дерева Хаффмана декодирует последовательность 0 и 1:

proalg1.py > ...

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from heapq import heappush, heappop

def print_tree(tree, level=0, levels=[]):
    if isinstance(tree, int):
        return

    for i, (node, child) in enumerate(tree.items()):
        if i == len(tree)-1 and level != 0:
            levels[level-1] = False
        branch = ''.join('    ' if lev else '    ' for lev in levels[:-1])
        branch += "└── " if i == len(tree) - 1 else "├── "
        if level == 0:
            print(str(node))
        elif isinstance(child, int):
            print(branch + f"'{node}'" + " — " + str(child))
        else:
            print(branch + str(node).split()[0])
        print_tree(child, level + 1, levels + [True])

def procedurehuffman(f):
    h = []
    l = len(f)
    visited_fs = set()  # Множество для хранения уникальных значений fs
    for i in f:
        heappush(h, (f[i], i))
    while len(h) > 1:
        f1, i = heappop(h)
        f2, j = heappop(h)
        fs = f1 + f2
        ord_val = ord('a')
```

Рисунок 1. Код программы

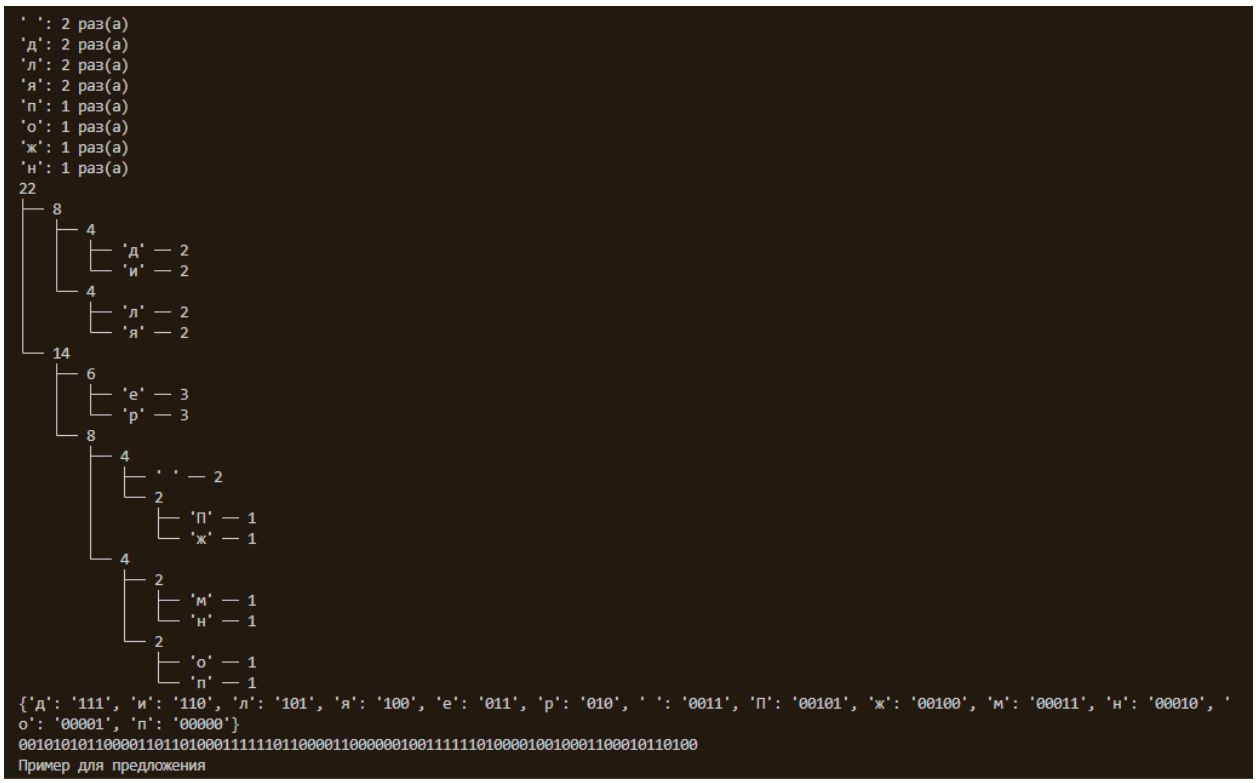

Рисунок 2. Результат выполнения программы

Вывод: в процессе выполнения лабораторной работы был изучен алгоритм кодирования Хаффмана, включая создание дерева Хаффмана и кодирование и декодирование текста с его помощью. Из этого следует, что метод кодирования Хаффмана представляет собой эффективный способ сжатия данных, особенно текстовых, в которых некоторые символы встречаются чаще, чем другие.