МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

по дисциплине
«Структуры и алгоритмы обработки данных»

Тема. Кодирование и сжатие данных методами без потерь

Выполнил студент группы ИКБО-25-22 Ракитин В.А.

Принял доцент Бузыкова Ю.С.

Москва 2023

Цель работы:

Задание 1 Исследование алгоритмов сжатия на примерах 1) Выполнить каждую задачу варианта, представив алгоритм решения в виде таблицы и указав результат сжатия. Примеры оформления решения представлены в Приложении1 этого документа.

Задание 2 Разработать программы сжатия и восстановления текста методами Хаффмана и Шеннона – Фано.

1) Реализовать и отладить программы.

Ход работы:

Метод Шенона-Фано

```
#include <iostream>
#include <map>
#include <string>
#include <vector>
#include <functional>
#include <algorithm>

using namespace std;

struct ShannonFanoNode {
    string symbol;
    double probability;
    string code;
};

vector<ShannonFanoNode> ShannonFanoEncoding(const string& data) {
    vector<ShannonFanoNode> symbols;

    // Подсчет частоты появления символов
    map<char, int> frequency;
    int totalSymbols = 0;
    for (char c : data) {
        frequency[c]++;
        totalSymbols++;
    }

    // Создание узлов для каждого символа
    for (const auto& pair : frequency) {
        ShannonFanoNode node;
        node.symbol = pair.first;
        node.probability = static_cast<double>(pair.second) / totalSymbols;
        symbols.push_back(node);
    }

    // Рекурсивная функция для назначения кодов каждому символу
    function<void(int, int)> assignCodes = [&](int start, int end) {
        if (start == end) {
            return;
        }

        double sum = 0.0;
        for (int i = start; i <= end; ++i) {
            sum += symbols[i].probability;
```

```
        }

        double halfSum = 0.0;
        int splitIndex = -1;
        for (int i = start; i <= end; ++i) {
            halfSum += symbols[i].probability;
            if (halfSum >= sum / 2) {
                splitIndex = i;
                break;
            }
        }

        // Назначение кодов символам до разделителя
        for (int i = start; i <= splitIndex; ++i) {
            symbols[i].code += '0';
        }

        // Назначение кодов символам после разделителя
        for (int i = splitIndex + 1; i <= end; ++i) {
            symbols[i].code += '1';
        }

        assignCodes(start, splitIndex);
        assignCodes(splitIndex + 1, end);
    };

    // Сортировка символов по вероятности (в убывающем порядке)
    sort(symbols.begin(), symbols.end(),
        [](const ShannonFanoNode& a, const ShannonFanoNode& b) {
            return a.probability > b.probability;
        });

    // Назначение кодов
    assignCodes(0, symbols.size() - 1);

    return symbols;
}

int main() {
    setlocale(LC_ALL, "rus");

    //====================Shennon-Fano coding=========================
    string data = "Кот пошёл за молоком, А котята кувырком. Кот пришёл без
молока, А котята ха - ха - ха.";
    vector<ShannonFanoNode> encodedSymbols = ShannonFanoEncoding(data);

    // Вывод закодированных символов и соответствующих кодов
    for (const auto& symbol : encodedSymbols) {
        cout << symbol.symbol << ": " << symbol.code << endl;
    }

    string message = "";

    for (char c : data) {
        for (const auto& symbol : encodedSymbols) {
            if (symbol.symbol.compare(string(1, c)) == 0)
                message += symbol.code + " ";
        }
    }

    cout << "Закодированное сообщение: " << message;
    system("pause");
    return 0;
}
```

Тестирование:

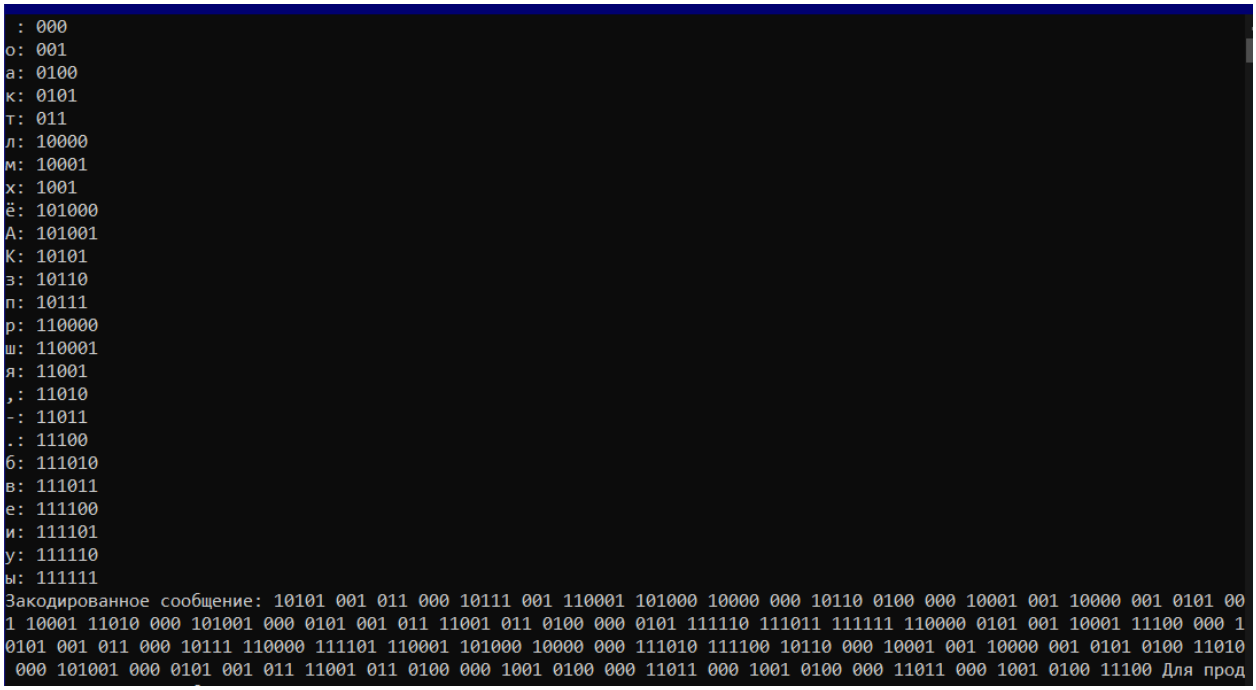

Рисунок 1. Тестирование работы программы кодирование методом Шенона-Фано

Метод LZ77

```
#include <iostream>
#include <string>
#include <vector>

using namespace std;

struct LZ77Token {
    int offset;
    int length;
    char nextChar;
};

vector<LZ77Token> LZ77Compress(const string& data, int windowSize) {
    vector<LZ77Token> compressedData;

    int dataSize = static_cast<int>(data.size());
    int currentIndex = 0;

    while (currentIndex < dataSize) {
        LZ77Token token;
        token.offset = 0;
        token.length = 0;
        token.nextChar = data[currentIndex];

        // Поиск наилучшего совпадения в окне
        int searchStart = max(currentIndex - windowSize, 0);
        int searchEnd = currentIndex;
        for (int i = searchStart; i < currentIndex; ++i) {
            int currentLength = 0;
```

```
            // Подсчет длины совпадения
            while (currentIndex + currentLength < dataSize &&
                data[i + currentLength] == data[currentIndex +
currentLength]) {
                ++currentLength;
            }

            // Обновление наилучшего совпадения
            if (currentLength > token.length) {
                token.offset = currentIndex - i;
                token.length = currentLength;
                token.nextChar = data[currentIndex + currentLength];
            }
        }

        compressedData.push_back(token);

        currentIndex += token.length + 1; // Добавляем длину совпадения +
следующий символ
    }

    return compressedData;
}

int main() {
   string data = "10101001101100111010";
   int windowSize = 6;
    vector<LZ77Token> compressedData = LZ77Compress(data, windowSize);

   // Вывод сжатых данных
   for (const auto& token : compressedData) {
        cout << "(" << token.offset << ", " << token.length << ", " <<
token.nextChar << ") ";
    }
    cout << endl;

    return 0;
      }
```

Тестирование:

```
(0, 0, 1) (0, 0, 0) (2, 4, 0) (5, 1, 1) (3, 4, 0) (4, 2, 1) (4, 2, 0)

D:\Учеба\3 семестр\СИАОД\ConsoleApplication10\x64\Debug\ConsoleApplication10.exe (процесс 14176) завершил работу с кодом
 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Ав
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рисунок 2. Тестирование работы программы кодирование методом LZ77

Метод LZ78

```
#include <iostream>
#include <map>
#include <string>
#include <vector>

using namespace std;

vector<pair<int, char>> LZ78Encode(const string& data) {
    vector<pair<int, char>> encodedData;
    map<string, int> dictionary;
```

```
    int currentIndex = 0;

    for (char c : data) {
        string currentPhrase = string(1, c);

        // Проверка, существует ли текущая фраза в словаре
        if (dictionary.count(currentPhrase) > 0) {
            currentIndex = dictionary[currentPhrase];
        }
        else {
            encodedData.emplace_back(currentIndex, c);
            dictionary[currentPhrase] = currentIndex + 1;
            currentIndex++;
        }
    }

    return encodedData;
}

string LZ78Decode(const vector<pair<int, char>>& encodedData) {
    string decodedData = "";
    map<int, string> dictionary;

    for (const auto& pair : encodedData) {
        int index = pair.first;
        char c = pair.second;

        string currentPhrase;

        // Проверка, существует ли текущий индекс в словаре
        if (dictionary.count(index) > 0) {
            currentPhrase = dictionary[index];
        }

        currentPhrase += c;
        decodedData += currentPhrase;
        dictionary[index + 1] = currentPhrase;
    }

    return decodedData;
}

int main() {
    setlocale(LC_ALL, "rus");
    string data = "bigbonebigborebigbo";
    vector<pair<int, char>> encodedData = LZ78Encode(data);

    // Вывод закодированных данных
    cout << "Закодированные данные: ";
    for (const auto& pair : encodedData) {
        cout << "(" << pair.first << ", " << pair.second << ") ";
    }
    cout << endl;

    // Раскодирование данных
    string decodedData = LZ78Decode(encodedData);
    cout << "Раскодированные данные: " << decodedData << endl;

    return 0;
      }
```

Тестирование:

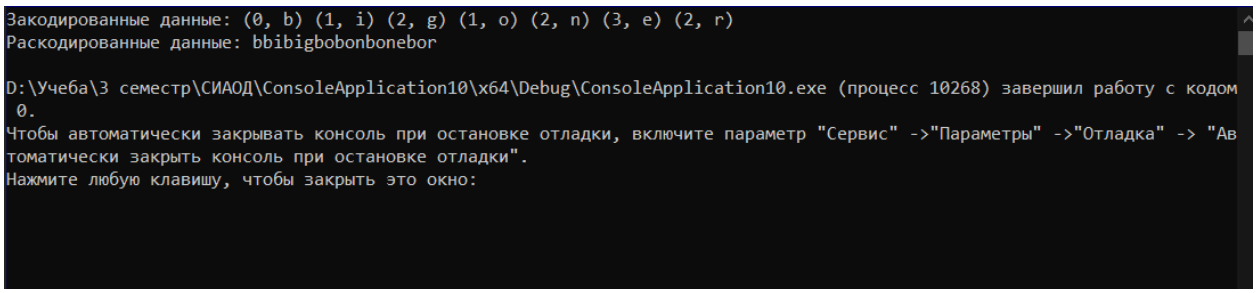

Рисунок 2. Тестирование работы программы кодирование методом LZ78