МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»

(БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа № 3

по дисциплине: Теория информации

тема: «Исследование возможности применения методов энтропийного кодирования для обработки двоичных последовательностей»

Вариант № 12

Выполнил: ст. группы ПВ-211

Павленко Станислав Вячеславович

Проверили:

Твердохлеб Виталий Викторович

Белгород 2023 г.

Содержание отчета

1. Аналитика касательно построения кодов для исходной двоичной

последовательности

2. Примеры кодовой реализации п.3, п.3, п.4 и п.6.

3. Результаты обработки кодов, полученных в п.5.

4. Текстовая последовательность, восстановленная к читаемому виду.

5. Общие выводы

Ход работы:

1. Открыть файл Лабораторная работа 3 (задание).txt. Рассмотреть возможность построения кода по методам Хаффмана и Шеннона-Фано для бинарной последовательности. Сделать выводы.

2. Рассмотреть варианты обработки цепочек символов, а именно:

- 2 символа;

- 4 символа;

- 8 символов.

Для этого разработать консольное приложение, разбивающее сплошной массив символов на цепочки заданной длины.

Код модуля:

```
std::vector<simb> breakToParts(std::vector<char> &massage, int size) {
    std::vector<simb> res;
    for (int i = 0; i < massage.size(); i += size) {
        std::string latter;

        for (int j = 0; j < size; ++j)
            latter.push_back(massage[j + i]);
        res.push_back(simb(latter));
    }

    return res;
}
```

3. Рассматривая каждую цепочку (2, 4 и 8 символов длиной) как отдельный символ, построить коды по методу Хаффмана и Шеннона-Фано.

Кодировка сообщений при разбиении на 2:

```
input massage: Shenon
00-00-754
11-01-518
01-10-395
10-11-393
hoffman
00 - 11 - 754
11 - 10 - 518
01 - 01 - 395
10 - 00 - 393
```

Кодировка сообщения при разбиении на 4:

```
input massage: Shenon
1101-000-248
0000-001-236
1011-01-183
0001-1000-87
1000-1001-85
0010-101-71
1110-11000-29
0101-11001-22
1100-1101-21
0011-111000-14
1010-111001-9
1111-11101-9
0111-111100-5
0110-111101-5
0100-111110-4
1001-111111-2
hoffman
1101 - 10 - 248
0000 - 01 - 236
1011 - 111 - 183
0001 - 000 - 87
1000 - 1101 - 85
0010 - 1100 - 71
1110 - 00101 - 29
0101 - 00100 - 22
1100 - 001111 - 21
0011 - 001100 - 14
1010 - 0011100 - 9
1111 - 0011011 - 9
0111 - 00111011 - 5
0110 - 00111010 - 5
0100 - 00110101 - 4
1001 - 00110100 - 2
```

Кодировка сообщения при разбиении на 8:

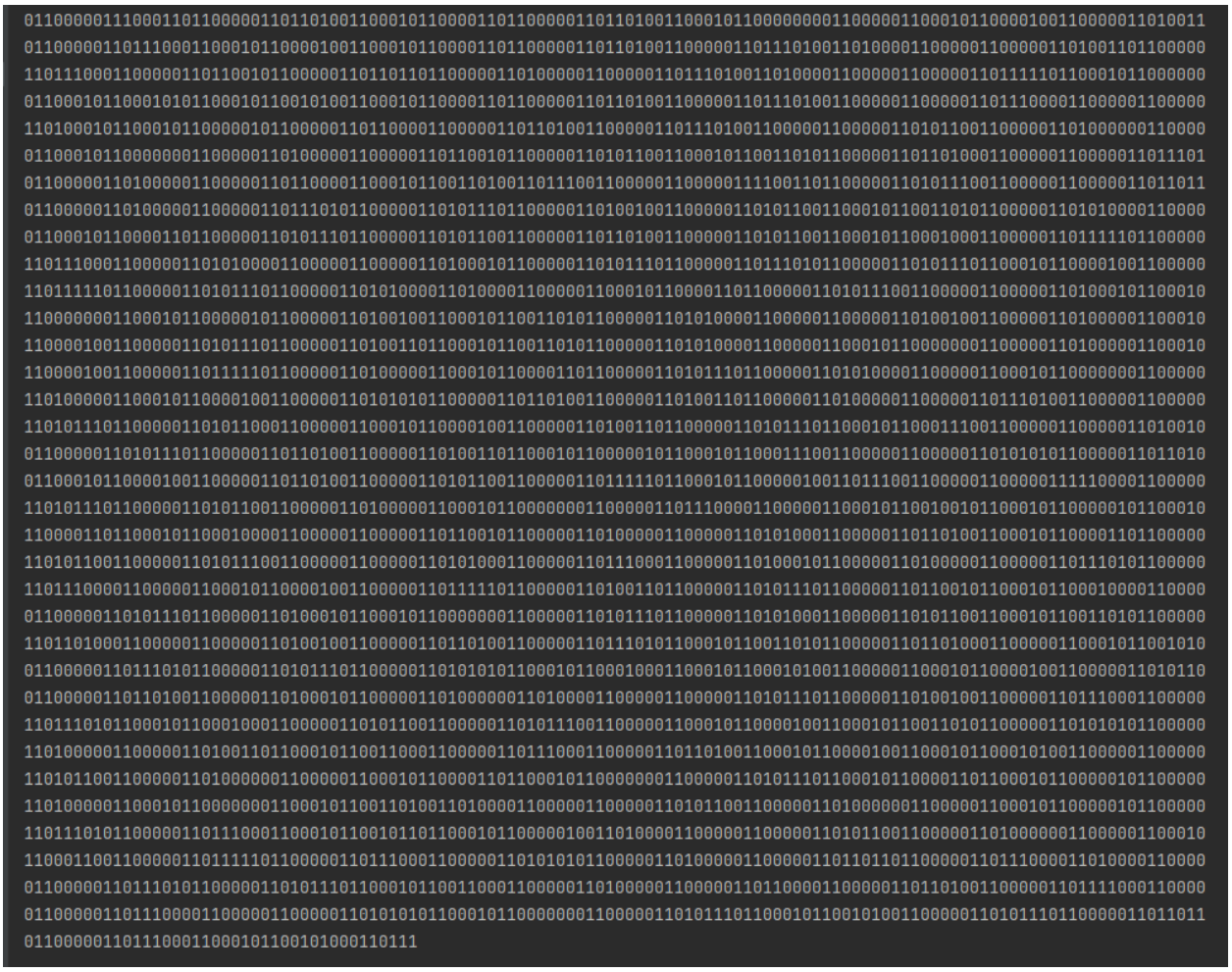
```
input massage: Shenon
11010000-00-163
11010001-010-69
00100000-011-41
10111110-10000-24
10110000-10001-21
10110101-10010-18
10111101-10011-15
10111000-10100-14
10111011-10101-14
10000001-10110-12
10000000-10111-11
10000010-110000-10
10001011-110001-9
10111100-110010-9
10110010-110011-8
10000011-110100-8
00101100-110101-7
10111010-11011-7
10110011-1110000-6
10110001-1110001-6
10111111-111001-6
10001100-1110100-5
10110111-1110101-4
10110110-111011-4
10000101-1111000-4
10001111-1111001-3
10110100-1111010-3
00101110-1111011-3
10001110-11111000-2
10001000-11111001-2
10010010-11111010-1
10100010-11111011-1
10100100-111111000-1
10000111-111111001-1
10000110-11111101-1
10001101-11111110-1
10111001-11111111-1
```

```
hoffman
11010000 - 11 - 163
11010001 - 100 - 69
00100000 - 1011 - 41
10111110 - 0010 - 24
10110000 - 0000 - 21
10110101 - 10100 - 18
10111101 - 01100 - 15
10111000 - 01011 - 14
10111011 - 01010 - 14
10000001 - 00110 - 12
10000000 - 00011 - 11
10000010 - 00010 - 10
10001011 - 101010 - 9
10111100 - 011111 - 9
10110010 - 011100 - 8
10000011 - 011011 - 8
00101100 - 011010 - 7
10111010 - 010011 - 7
10110011 - 010000 - 6
10110001 - 001111 - 6
10111111 - 001110 - 6
10001100 - 1010111 - 5
10110111 - 0111100 - 4
10110110 - 0111011 - 4
10000101 - 0111010 - 4
10001111 - 0100100 - 3
10110100 - 0100011 - 3
00101110 - 0100010 - 3
10001110 - 01111010 - 2
10001000 - 01001011 - 2
10010010 - 01001010 - 1
10100010 - 101011011 - 1
10100100 - 101011010 - 1
10000111 - 101011001 - 1
10000110 - 101011000 - 1
10001101 - 011110111 - 1
10111001 - 011110110 - 1
```

4. Составить последовательности из полученных кодов символов для каждого случая.

Метод Шеннона-Фано:

Разбиение на 2:



Разбиение на 4:

```
0000011111111010000010111001000100010011010000010111001000100010010011010010001000100110000000010110100000101100100010001000
1001100000001000100110100000101110010000010101101110110100100000101101000001011001000001011111000000010111110100000101001
0000010101101110110100100000101111001000100010010010001000100111101000100010011100100010001001101000001011100100000010101
1010010000010110011010010000010111100000001000100111000000001011111100000010111001000001010110100100000101000000000101001
1010010001000100100100000101001000001011111000000001010000001000100101000000101110011010010000010101000000101001000001011111
1100001000100101101110001010010000011110011010000010111000101001000001011111010000010100100000101010000001011100000000101
1000000001010000001000100101000000101110110100100010001001101000001011100000000101000000001011100100000101000000010001001 1
1010000010111100100000101100100000101110110100100000101111000000001011100000000101010000001011100000001000100110000000000101
1110010000010111000000000101110110111011010010001000100110100000101110001010010000010111100000001000100100100010001001111 0
0000000101100000001000100101000000101110110100100000101100000000010100100010001001100000000010111000000000101101000100010 0101
0000010111011010010001000100100100000101001000100010011000000000101111001000001010010001000100110100000101110000000001011 1
0110100100010001001001000001010010001000100110000000001011110100000101110010000010110100000101001000001010110100100000101
1100000000101000101001000100010011000000000101101000001011100000001000100111000101001000001011000000000101110000000001011100
1000001011010001000100111100000001000100111000101001000001011110100000101110010001000100110000000010111001000001010000000
0101111001000100010011110001011100010100100000111100111111000000101110000000001010000000010100100010001001001000001011001
101001000100010011111000000100010011110000000100010011010001000100111011010010000010111110000000101001000001011101000000101
1100100010001001101000001010000000010111000101001000001011101000001011001000001011110000000001010010000010101000001011001
1010010001000100110000000010111100100000101101000001011100000000101111100000010001001110110100100000101110000000001011110 0
0000100010010010000010111000000000101110100000101000000100010010100000101110011010010000010110000000001011100100000101010 0
0100010010100000101110011010010001000100111001000001010100000101110000000001011110100010001001110100010001001111011010010
0010001001100000000010100000000010111001000001011110000000001010011011101101001000001011100000000010110000000001011001000001 0
1010001000100111010000010100000000101110001010010001000100110000001000100101000001011110100000101001000001011010001000 10
0110010000010110010000010111001000100010011000000010001001111011010010000010100000000010100110100100010001001101000100010 0
1001000001011100000001000100110100010001001111000000000101001000100010010010001000100101101110110100100000101000000000101 00
1101001000100010011110000000001010100000101100100010001001111101000100010011110001011101101001000001010000000001010011010 0
1000100010010000000010111100100000101100100000101111010000010100100000101111101000001011001011101101001000001010100000 1
0111000000010001001100100000101001000001011111100000001011100100000101111111010010000010110011010010000010111101000100010
010010000010111000000010001001110010000010111000000000101111010000010110010001000100111001101110 00
```

Разбиение на 8:

```
001111101000100100101100000001001001010111011010101100011001100101000101011001011000000100100010101110101011001100110010 1
0000111010100111011001000100101011101010110011011010101110101111001010111100001011000000100100010101011001010001100111 00
0001011010000111101000100100010101011001001100100010110101011100100010011101010010011010110001001001001100101010010001 00
1111010010110001111101101100111110110010000011001110110010001001010100100000011100010010011010110001001100100110101100 00
0010000000100110010010001001101011101000011011001010000110010011001110000000000001010100100000101011000110110010000001 10
0101101010110101100000001000001100111000000101011101011010000111000101011000100110010011001110001001000101010110001000 0001
1001101011000100110010011010101110010001010101100011011001000101011000000010000001100100110101011100100010101011000111 001
0010010001100110010001001010101100100000001001101101010110001100110010000001011111000011001110001001000000100100011001 1010
1101000101111100001100111001001001001010110001001000100110011011010110100111101101100111111000000100000001001100100010101 0
1110010100011010111111001010110100010110000010111010001100111010100100010011001000100100101100000010011001000000110011001
0001010000111000000100010010101001010001101010110001101100110011001000000011101010101110100011001000000111000001010111001
0000001100100010011010110001001001001100111000100100100010101010110001001001001101011110000010101001000000011100101011101
0001011110010110101010110001001100100100011100000001000111010101100100000011100010010100001010101011101000010011001000000110
1010110010110001001110010010001001100110101111100100101000010010010101100101111001011001001100100010110101100000101011 10
0100000101100000101101000010001010101110101100011101010110010011001000101101011010000101010010100010111111010101101001 10
1010110010011001000101101011111111000110110010100001110010010001001110110010100110101011001010100100000101111100100100010
0111101000100100011111111011001010001100111001010101110010000001011110000001000000111011001010001011110001111011
```

Метод Хаффмана:

Разбиение на 2:

Разбиение на 4:

```
1001001101001100100111100100100001101110010011110010010000110101110001100001101000100111111001001111110110000110100010000
0110111001001111001001001111111110000111111000110011111100100111111011001111001110111001111001110101001111011001111111111
0000111111000110011110011100100001101011000011010011011100001101001001000011011100100111100100100111111111000110011111110
1110001100111100110010000110100110010011110011010110011110010010011111111100011001111010011110111000110000110101100111 1
0110011110011101110011111010000110111111001111001001100011001111111100111101100111100110101100001101111110000101110001100
1001110011001001111001011100011001111001110101001111011001111111100111100101100111100010011111010000110111111001111001111
1100011000011011100100111100101100111110100111100100100111110100001101001111100111100111001001111110110011110011111110001
1001111001100100111100101100111111110011110010110000110100010011110011100100111100101100111100111111000011111100011000001
1011100100111100101110001100111100110010000110101100001101001100100111100010000110111111001111001111110001100111100010011
1101100001101000100111100101100111111001000011011111100111100111111000110000110101100111101100001101000100111100111001001
1110110000110111001001111001011001111001111110001100001101011001111011000011010001001111001101110011110010010011111110010
0111101100111111111000110011110010110011111011000110000110100010011111100100111100101100001101001011100011001111000010011
1100101100111100100100111111001000011010011001000011010010111000110011110011011100111100100100001101000100111100100100011
1110100111100111001000011010011001100001011100011001001110000110101100111100101100111110100111101100001101011001111111011
1000110000110100111011100001101001100100001101110010000110100111111000110011110011101110011110110011110011111001111100100
1000011011100100111110100111100101110001100111100111110011111101100111100110010011110110011111111001111110111000110000011
0100010011110011100100111111001001111001011001111001110111000011010011111100011001111001011001111001100100001101011001111
1001011001111001111100111110100001101111110011110010011000110011110001001111001001001111111100001101111110011110010011000 1
1000011010010010011111111001111001011001111001101110000110100111110000110100110111100011000011010001001111101001111100100
1001111001100100111101110000111111000110011110010110011110001001111110110011111111000011010011111001111101001111100101110
0011000011010001000011011111100111100110111001111011001111110010000110111011001111110110011110010010000110100010000110100
1101111000110011111010011110111000110000110111001000011010110011110010110000110111001000011010011001001111011000011010 11
0000110111111000011111100011001111101001111011100011000011010011001001111111100111111011000011010011101010000110100110 01
1000011111100011001111101001111011100011000011011010011110011100100111110110011110011011100111101100111100111010100111 1
1101110000111111000110011111111001111001011000011011101100111101100111100110101100111100100100111100110100110001100111 11
1011100011001111001101110000110101100111100101100001101001001001111001011001111001110101001111110110000110100100110000 10
1
```

Разбиение на 8:

```
1101001010111010010000001011101001000000111011100001101101110011010111000011010000001011101001101010011010101111011100 11010
1111011110011011101111000011010100110101011110100111000000111000100100100011101010000001011101001101010101111010111011 11101
0000100011011110100011111010011010101011110110011000010111000000111100001101111001101100100101010111010010111101010110000
1101000111001010100100010101111101011011110010101111011101111000011010101100101100111111011001001010101101111111011100000
1011001011011001110100110110010010101111101001111010111101111110111101000011001011010101100101000011011010011110010110 11
1110110101011100000101100101011110100001000001110001101111001111100101010110111111011110011111100001000011011001011011 10
0100101010110111111011100000111100001000011011010011110000100000101100101101111110111000001111000010000110110011101110 10
0110111001100001101010101111001011011001011100001101101110011001010001111010101111001111110010111010011011100100011011 10
0011110101011110011101111010010000110111010011011001101001110001101101000101011111010110101100101101100110000100000111101
0111011100101011001100011011100000010100101011110111101111001100001101111111101001000000101101100110010101111011111110101 1
1101000011000011010101101011101110000110110100111101110011001011011110010010101111011110010110100001000001111001011011 11
1111011001001010101110100101111001111111010011010101001010101110100101110001110101101010110010110011101001010111100010010
0101110000110110110011101001101000011000001101010111100101100111111010111101010100101011111011001100101011100001101001 01
0101100111011000011011100100010010111101011111010010000110100010010010111101100110000101110000001010000011110010100000101
0001101111000010000001110010101001101010111101100110000101110001101111010101101011100101011000100011011011010101111011001
1000010111000111101111110100111101011110011101100001101110111101011011010101111010101100101000100101111000011010001111101
0011011110110101111010111011110011101000000111100101000111010110010110111011110101110001110100100010
```

5. По результатам работы в п.3 сделать выводы по поводу полученных результатов для каждого из методов (простота, скорость, полученные результаты (рассчитать коэффициенты сжатия)).

	N = 2	N = 4	N = 8
Хаффмана	1	1.32	2.04
Шеннона-Фано	1	1.23	2.03

6. Написать программу, восстанавливающую последовательности, полученные в п.3 в исходный вид согласно вариантам, приведенным в п.2.

Модуль для декодирования:

```
std::vector<simb> createAlp() {
    std::cout << "input size of alp: ";
    int size;
    std::cin >> size;

    std::vector<simb> alp;

    for (int i = 0; i < size; ++i) {
        simb sb;
        std::string s;
        std::cin >> s;

        int j = 0;
        std::string liter;
        while (s[j] != '-') {
            liter.push_back(s[j]);
            j++;
        }
        sb.liter = liter;
        j++;
        liter = "";
        while (s[j] != '-') {
            liter.push_back(s[j]);
            j++;
        }
        sb.code = liter;
        j++;
        int n = 0;
        while (j < s.size()) {
            n += s[j] - '0';
            j++;
        }
        sb.ratio = n;

        alp.push_back(sb);
    }

    return alp;
}

void decodeMassage(std::string &massage, std::vector<simb> &alp) {
    int iStart = 0;
    int iEnd = 0;

    std::string liter;
    while (iStart < massage.size() - 1) {
        liter.push_back(massage[iEnd]);
```

```
        int i = -1;
        bool isFound = false;
        while (i != alp.size() - 1 && !isFound) {
            i++;
            if (liter == alp[i].code)
                isFound = true;
        }

        iEnd++;

        if (isFound) {
            std::cout << alp[i].liter;
            liter = "";
            iStart = iEnd;
        }
    }
}
```

Ввод программы:

```
C:\Users\HP\CLionProjects\TI3\cmake-build-debug\TI3.exe
00111110100010010010110000001001001010111011010101100011001100101000101011001011000
input massage: input alph:input size of alp: 37
11010000-00-163
11010001-010-69
00100000-011-41
10111110-10000-24
10110000-10001-21
10110101-10010-18
10111101-10011-15
10111000-10100-14
10111011-10101-14
10000001-10110-12
10000000-10111-11
10000010-110000-10
10001011-110001-9
10111100-110010-9
10110010-110011-8
10000011-110100-8
00101100-110101-7
10111010-11011-7
10110011-1110000-6
10110001-1110001-6
10111111-111001-6
10001100-1110100-5
10110111-1110101-4
10110110-111011-4
10000101-1111000-4
10001111-1111001-3
10110100-1111010-3
00101110-1111011-3
10001110-11111000-2
10001000-11111001-2
10010010-11111010-1
10100010-11111011-1
10100100-111111000-1
10000111-111111001-1
10000110-11111101-1
```

Результат работы программы:

11010000100100101101000010110101111010001100000101101000010110101111010001100000000010000001101000110000001110100001011001011010000101110001101000110000001110100011000001011010001101011101000010111011001011000010000011010000101100101101000010111000

10100001011011111010000101101101101000010110000110100001011101100101100001000001101
00001011101011010001100000001101000110001111110100011000010111010001100000101101000
01011010111010000101110110010000011010000101110000010000011010000101100111101000110
00001111010000101101001101000010110101110100001011101100100000110100001011110111010
00010110000001000001101000110000000110100001011000011010000101101111101000010111101
11010001100010111101000010110101001000001101000010111011110100001011000011010000101
10100110100011000101100101110001000001101000010100010110100001011111000100000110100
00101101101101000010110000011010000101110111101000010111110110100001011000111010000 1
01111011101000110001011110100001011110000100000110100011000001011010000101111101101
00001011110111010000101101011101000010111101111010001100011001101000010111010110100 0
01011100011010000101111000010000011010000101100111101000010111110110100001011101111
01000010111110110100011000000111010000101110101101000010111110110100001011110000101
10000100000110100011000001011010000101111100010000011010000101100111101000110000000
11010001100000111101000010110001110100011000101111010000101111000010000011010000101
10001110100001011000011010001100000011101000010111110110100001011001011010001100010
11110100001011110000100000110100011000000011010000101100001101000110000001110100001
01110101101000010110000011010001100000010111010000101111110110100001011110000100000 1
00011000000011010000101100001101000110000001110100001011111111010000101101011101000
01011001011010000101100001101000010111011001000001101000010111110110100001011101000
10000011010001100000011101000010110010110100001011111011010001100011100001000001101 0
00010110001110100001011111011010000101101011101000010110010110100011000001111010001
10001110001000001101000010111111110100001011010111010001100000011101000010110101110
10000101111011101000010111010110100011000001100101110001000001101000010100100110100
00101111101101000010111101110100001011000011010001100000001101000010111000001000001
10100011000011111010001100000111101000110000010110100011000110000100000110100001011
01111101000010110000110100001011110011010000101101011101000011000001011010000101110
11101000010111110001000001101000010111100110100001011100011010000101100111101000010
11000011010000101110111110100001011100000100000110100011000000111010000101110101101 0
00010110010110100001011111011010000101101111101000110001100001000001101000010111110
11010000101100111101000110000000110100001011111010110100011000010111010000101111101110
10001100010111101000010110101001000001101000010110001110100001011010111010000101110
11110100011000101111010000101101010010000011010001100001011101000010111011110100001
01111101101000010111111110100011000110011010001100011110010000011010001100000011101
00001011110111010000101101011101000010110011110100001011000000101100001000001101000
01011111011010000101100011101000010111000110100001011101111010001100011001101000010
11101111010000101111100010000011010001100000011101000110001011110100001011111111010
00010110000110100001011001011010001100010001101000010111000110100001011010111010001
10000001110100011000111100100000110100001011110111010000101100000010000011010001100
00010110100011000000011010000101111101101000110000010110100011000001111010000101100
00110100011000000011010001100010110010110000100000110100001011110111010000101100000
01000001101000110000011110100001011101111010000101110001101000110000110110100011000
00110010110000100000110100001011110111010000101100000010000011010001100011011101000
01011101011010000101110001101000010111111110100001011000011010000101101101101000010
11100000101100001000001101000010111011110100001011111011010001100010001101000010110
00011010000101101001101000010110101110100001011100100100000110100001011110000010000
11010000101111111101000110000000110100001011111011010001100001011101000010111110110
10000101101101101000010111000110100011000010100101110

Результат совпадает с исходным сообщением.

7. Восстановить исходный текст из полученных последовательностей, пользуясь сервисом https://onlineutf8tools.com/convert-binary-to-utf8.

Исходный текст:

Ветер свистел, визжал, кряхтел и гудел на разные лады. То жалобным тоненьким голоском, то грубым басовым раскатом распевал он свою боевую песенку. Фонари чуть заметно мигали сквозь огромные белые хлопья снега, обильно сыпавшиеся на тротуары, на улицу, на экипажи, лошадей и прохожих.