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»

Выполнил: ст. группы ПВ-211

Шамраев Александр Анатольевич

Проверил:

Твердохлеб Виталий Викторович

Белгород 2023 г.

содержание отчета

[Задание 3](#_Toc129644482)

[1 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. 4](#_Toc129644483)

[2 Рассматривая каждую цепочку (2, 4 и 8 символов длиной) как отдельный символ, построить коды по методу Хаффмана и Шеннона-Фано 5](#_Toc129644484)

[3 Составить последовательности из полученных кодов символов для каждого случая. 9](#_Toc129644485)

[4 По результатам работы в п.3 сделать выводы по поводу полученных результатов для каждого из методов (простота, скорость, полученные результаты (рассчитать коэффициенты сжатия)). 17](#_Toc129644486)

[5 Написать программу, восстанавливающую последовательности, полученные в п.3 в исходный вид согласно вариантам, приведенным в п.2. 18](#_Toc129644487)

[Вывод: 22](#_Toc129644488)

Задание

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:
   1. 2 символа;
   2. 4 символа;
   3. 8 символов.

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

1. Рассматривая каждую цепочку (2, 4 и 8 символов длиной) как отдельный символ, построить коды по методу Хаффмана и Шеннона-Фано.
2. Составить последовательности из полученных кодов символов для каждого случая.
3. По результатам работы в п.3 сделать выводы по поводу полученных результатов для каждого из методов (простота, скорость, полученные результаты (рассчитать коэффициенты сжатия)).
4. Написать программу, восстанавливающую последовательности, полученные в п.3 в исходный вид согласно вариантам, приведенным в п.2.
5. Восстановить исходный текст из полученных последовательностей, пользуясь сервисом <https://onlineutf8tools.com/convert-binary-to-utf8>

# 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.

std::string ParseCode(  
 const std::string &str,  
 const size\_t n  
) {  
 std::string out;  
   
 size\_t l = str.length() / n;  
 for (::size\_t i = 0; i < l; i++) {  
 int id = 0;  
 for (size\_t j = 0; j < n; j++) {  
 id = id \* 2 + str[i \* n + j] - '0';  
 }  
   
 out.push\_back(id);  
 }  
   
 return out;  
}

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

Использован код из прошлой лабораторной работы.

Результат работы:

Для 2-х символьного алфавита

[FANO]

Table:

<☻> = 00

<☺> = 01

<♥> = 10

< > = 11

[HUFFMAN]

Table:

< > = 11

<♥> = 10

<☺> = 01

<☻> = 00

Для 4-х символьного алфавита

[FANO]

Table:

< > = 00000000

<♦> = 00000001

<♠> = 00000010

> = 11

< > = 10

<♂> = 011

<☺> = 010

<♫> = 00011

> = 0011

<☻> = 0010

<☼> = 0000011

<♣> = 00010

<♀> = 000011

<♥> = 000010

<

> = 0000010

<> = 00000011

[HUFFMAN]

Table:

<♂> = 111

> = 1101

<☻> = 1100

<☺> = 000

<♫> = 00101

<♣> = 00100

<♥> = 001100

< > = 00110100

<♦> = 00110101

<

> = 0011011

<☼> = 0011100

<♠> = 00111010

<> = 00111011

<♀> = 001111

< > = 01

> = 10

Для 8-и символьного алфавита

[FANO]

Table:

<й> = 000000000

<Т> = 00000001

<ч> = 000000100

<Ф> = 000000101

<в> = 01001

<я> = 0000101

<м> = 01011

<р> = 01101

<,> = 00110

<с> = 0111

<и> = 10000

<л> = 10001

<т> = 01100

<.> = 0000100

<е> = 1010

<ы> = 01010

<о> = 110

<н> = 1001

<а> = 1011

<ц> = 00000011

< > = 111

<д> = 000011

<у> = 01000

<к> = 00111

<б> = 00101

<ю> = 00000101

<п> = 001001

<г> = 001000

<ь> = 000111

<э> = 000000001

<В> = 00000100

<з> = 000110

<ж> = 000101

<х> = 000100

<ш> = 0000011

[HUFFMAN]

Table:

<п> = 111111

<б> = 111110

<с> = 11110

<о> = 1110

<р> = 11011

<т> = 11010

<у> = 01100

<в> = 01011

<я> = 000100

<к> = 01010

<н> = 0100

<и> = 0011

<,> = 00011

<л> = 0010

<ш> = 0001011

<д> = 000011

<ц> = 0001010

< > = 101

<.> = 000010

<г> = 00000

<з> = 011010

<х> = 011011

<ж> = 011100

<ю> = 0111010

<ч> = 01110110

<Ф> = 01110111

<м> = 01111

<е> = 1000

<ы> = 10010

<Т> = 10011000

<ь> = 100111

<В> = 10011001

<э> = 10011010

<а> = 1100

<й> = 10011011

# Составить последовательности из полученных кодов символов для каждого случая.

Для 2-х символьного алфавита

[FANO]

Coded:

100111110001110010011111001001011001110100111100100111110010010110011101001111111100111110011101001111011001111100101100

100111110010001110011101001111011001110100111100100111110010010110011111001000101100101111001111100111110010110010011111

001000111001111100100110100111110010010010011111001011111001111100100010110010111100111110011111001000001001110100111111

100111010011101010011101001101011001110100111100100111110010010110011111001000101100111110011111001000111100111110011111

001011101001110100111110100111110010011110011111001001011001111100100010110011111001111100101001100111110010111111001111

100111010011111110011111001011111001111100100110100111110010100110011101001100101001111100100101110011111001111100100010

100111110010111110011111001001111001110100110010110010001100111110011111000011001001111100101000110011111001111100100100

100111110010111110011111001000101001111100101000100111110010110110011111001010011001110100110010100111110010101111001111

100111010011110010011111001010001001111100101001100111110010010110011111001010011001110100111011100111110010000010011111

001000111001111100101011110011111001111100101110100111110010100010011111001000101001111100101000100111010011110110011111

001000001001111100101000100111110010101111001011110011111001110100111100100111110010100011001111100111110010111010011101

001111111001110100111110100111110010110110011101001100101001111100101011110011111001111100101101100111110010111110011101

001111011001111100101000100111110010110010011101001100101001111100101011110011111001110100111111100111110010111110011101

001111011001111100100000100111110010111110011101001111001001111100101000100111110010101111001111100111010011111110011111

001011111001110100111101100111110010101010011111001001011001111100101100100111110010111110011111001000101100111110011111

001010001001111100101001110011111001110100111101100111110010110010011111001010001001110100111000110011111001111100101101

100111110010100010011111001001011001111100101100100111010011111010011101001110001100111110011111001010101001111100100101

100111010011110110011111001001011001111100101001100111110010000010011101001111101100100011001111100111110000011110011111

001010001001111100101001100111110010111110011101001111111001111100100011110011111001110100110110100111010011111010011101

001111001001110100111011110011111001111100100110100111110010111110011111001010111001111100100101100111010011110010011111

001010011001111100101000110011111001111100101011100111110010001110011111001011101001111100101111100111110010001010011111

001000111100111110011101001111011001111100100000100111110010110010011111001010001001111100100110100111010011101111001111

100111110010100010011111001011101001110100111111100111110010100010011111001010111001111100101001100111010011001010011111

001001011100111110011111001011011001111100100101100111110010001010011101001100101001111100100101110011111001110100110101

100111110010001010011111001010001001111100101010100111010011101110011101001110101100111110011101001111011001111100101001

100111110010010110011111001011101001111100101111110010111100111110011111001010001001111100101101100111110010001110011111

001000101001110100111011100111110010100110011111001010001100111110011101001111011001110100110010100111110010101010011111

001011111001111100101100100111010011001110011111001000111001111100100101100111010011110110011101001110101100111110011111

001010011001111100101111110011111001110100111100100111010011111110011111001010001001110100111100100111010011111010011111

001011111001110100111111100111010011001011001011110011111001111100101001100111110010111111001111100111010011111010011111

001000101001111100100011100111010011010010011101001111101100101111001111100111110010100110011111001011111100111110011101

001110011001111100100000100111110010001110011111001010101001111100101111100111110010010010011111001000111100101111001111

100111110010001010011111001010001001110100110011100111110010111110011111001001111001111100100101100111110010000111001111

100111110010001111001111100111110010101010011101001111111001111100101000100111010011010110011111001010001001111100100100

1001111100100011100111010011010111001000

[HUFFMAN]

Coded:

100111110001110010011111001001011001110100111100100111110010010110011101001111111100111110011101001111011001111100101100

100111110010001110011101001111011001110100111100100111110010010110011111001000101100101111001111100111110010110010011111

001000111001111100100110100111110010010010011111001011111001111100100010110010111100111110011111001000001001110100111111

100111010011101010011101001101011001110100111100100111110010010110011111001000101100111110011111001000111100111110011111

001011101001110100111110100111110010011110011111001001011001111100100010110011111001111100101001100111110010111111001111

100111010011111110011111001011111001111100100110100111110010100110011101001100101001111100100101110011111001111100100010

100111110010111110011111001001111001110100110010110010001100111110011111000011001001111100101000110011111001111100100100

100111110010111110011111001000101001111100101000100111110010110110011111001010011001110100110010100111110010101111001111

100111010011110010011111001010001001111100101001100111110010010110011111001010011001110100111011100111110010000010011111

001000111001111100101011110011111001111100101110100111110010100010011111001000101001111100101000100111010011110110011111

001000001001111100101000100111110010101111001011110011111001110100111100100111110010100011001111100111110010111010011101

001111111001110100111110100111110010110110011101001100101001111100101011110011111001111100101101100111110010111110011101

001111011001111100101000100111110010110010011101001100101001111100101011110011111001110100111111100111110010111110011101

001111011001111100100000100111110010111110011101001111001001111100101000100111110010101111001111100111010011111110011111

001011111001110100111101100111110010101010011111001001011001111100101100100111110010111110011111001000101100111110011111

001010001001111100101001110011111001110100111101100111110010110010011111001010001001110100111000110011111001111100101101

100111110010100010011111001001011001111100101100100111010011111010011101001110001100111110011111001010101001111100100101

100111010011110110011111001001011001111100101001100111110010000010011101001111101100100011001111100111110000011110011111

001010001001111100101001100111110010111110011101001111111001111100100011110011111001110100110110100111010011111010011101

001111001001110100111011110011111001111100100110100111110010111110011111001010111001111100100101100111010011110010011111

001010011001111100101000110011111001111100101011100111110010001110011111001011101001111100101111100111110010001010011111

001000111100111110011101001111011001111100100000100111110010110010011111001010001001111100100110100111010011101111001111

100111110010100010011111001011101001110100111111100111110010100010011111001010111001111100101001100111010011001010011111

001001011100111110011111001011011001111100100101100111110010001010011101001100101001111100100101110011111001110100110101

100111110010001010011111001010001001111100101010100111010011101110011101001110101100111110011101001111011001111100101001

100111110010010110011111001011101001111100101111110010111100111110011111001010001001111100101101100111110010001110011111

001000101001110100111011100111110010100110011111001010001100111110011101001111011001110100110010100111110010101010011111

001011111001111100101100100111010011001110011111001000111001111100100101100111010011110110011101001110101100111110011111

001010011001111100101111110011111001110100111100100111010011111110011111001010001001110100111100100111010011111010011111

001011111001110100111111100111010011001011001011110011111001111100101001100111110010111111001111100111010011111010011111

001000101001111100100011100111010011010010011101001111101100101111001111100111110010100110011111001011111100111110011101

001110011001111100100000100111110010001110011111001010101001111100101111100111110010010010011111001000111100101111001111

100111110010001010011111001010001001110100110011100111110010111110011111001001111001111100100101100111110010000111001111

100111110010001111001111100111110010101010011101001111111001111100101000100111010011010110011111001010001001111100100100

1001111100100011100111010011010111001000

Для 4-х символьного алфавита

[FANO]

Coded:

111000000000001011100110001011010001100101110011000101101000111000101011010001101011100110010111001100111101000110101101

000110010111001100010111001101100100000110010101110011001011100110011111001100000011111001100000010111001110111001101100

100000110010101110011000001011010001110110100011000001111010001100010110100011001011100110001011100110110010101110011001

100101011100110000101101000110000101110011000000011110011000101110011011001010111001111111001110001010110100011101110011

101110011000000111110011111101000110111110011000100010101110011011111001110111001100000001110100011011001000011001010111

000000100010111001100011001010111001100000010111001110111001101111100110001111100110101110011111101000110111110011000011

001010110100011001011100110001111100111111100110001011100111111010001100001111100110000010111001100111110011000011001010

111001100001011100110001111100110111110011000111101000110101110011000001011100110001111100110000110010000011001010110100

011001011100110001100101011100110000101101000111011010001100001011100110101101000110111110011000011001010111001101011100

111011010001101011100110001111100110010110100011011111001100001100101011010001110111001110110100011010111001100000101110

011101101000110010111001100011111001100001100101011010001110111001110110100011010111001100000111110011000101110011001011

100111011100110110010101110011000111110011110010101101000110101110011001011100110001111010001100011001010111001101011100

110001111100110001011100110010110100011000010110100011000110010101110011000001111100110001011010001101011100110001011100

111111100110000010110100011000010001000011001010111000000100000000111100110001111100111111100111011010001110111001100110

010101101000110000001111010001100001011010001100101101000110000110010101110011000000111110011101110011000011111001100010

110100011001011100111111100110001100101011100110000111110011001111100110000101110011101110011011111001100110010101101000

110101110011000001011100110010111001100011111001100000011110100011000011001010111001100011111001100001011010001110111001

100011111001100001111100111111010001101111100110001000101011100110101110011000101110011011110100011011111001100010001010

110100011000101110011011111001100011111001100000111101000110000111101000110000011001010110100011010111001111111001100010

111001100001011100111000100000110010101110011000111110011010111001100111110011011110100011000011111001111111001100011001

010110100011010110100011011111001100000111110011101110011001011010001100111110011001111100110001011010001101011010001100

000110010101110011111110011100010101101000110010110100011101110011000111101000110010110100011000010111001110110100011101

101000110110010000011001010111001111111001110001010110100011000010111001101111100110011110100011000000101101000110000100

010000011001010111001111111001110001010110100011111110011000001011100110011111001100000111110011101110011000000101110011

001100100000110010101110011011111001100011110100011001111100111011100110000000111100110001011100110000000000101011100110

011001010111001100000111101000111011100110001111010001100010111001100011111001100000010111001100111101000110001000100001

1

[HUFFMAN]

Coded:

100100110100110010011110010010000110111001001111001001000011010111000110000110100010011111100100111111011000011010001000

011011100100111100100100111111111000011111100011001111110010011111101100111100111011100111100111010100111101100111111111

000011111100011001111001101110000110101100001101001110010000110100100100001101110010011110010010011111111100011001111110

111000110011110011001000011010011001001111001101011001111001001001111111110001100111110100111101110001100001101011001111

011001111001110111001111101000011011111001111001001100011001111111100111101100111100110101100001101111110000101110001100

100110111100100111100101110001100111100111010100111101100111111110011110010110011110001001111101000011011111001111001111

110001100001101110010011110010110011111010011110010010011111010000110100111110011110011011100111111011001111001111110001

100111100110010011110010110011111111001111001011000011010001001111001101110011110010110011110011111100001111110001100001

101110010011110010111000110011110011001000011010110000110100110010011110001000011011111001111001111110001100111100010011

110110000110100010011110010110011111100100001101111100111100111111000110000110101100111101100001101000100111100110111001

111011000011011100100111100101100111100111111000110000110101100111101100001101000100111100111001001111001001001111110010

011110110011111111100011001111001011001111101100011000011010001001111110010011110010110000110100101110001100111100010011

110010110011110010010011111100100001101001100100001101001011100011001111001110010011110010010000110100010011110010010011

111010011110011011100001101001100110000101110001100100110110011010110011110010110011111010011110110000110101100111111011

100011000011010011101110000110100110010000110111001000011010011111100011001111001110111001111011001111001111100111100100

100001101110010011111010011110010111000110011110011111001111110110011110011001001111011001111111100111111011100011000011

010001001111001101110011111100100111100101100111100111011100001101001111110001100111100101100111100110010000110101100111

100101100111100111110011111010000110111110011110010011000110011110001001111001001001111111100001101111100111100100110001

100001101001001001111111100111100101100111100111001000011010011111000011010011100110001100001101000100111110100111100100

100111100110010011110111000011111100011001111001011001111000100111111011001111111100001101001111100111110100111100101110

001100001101000100001101111100111100111001001111011001111110010000110111011001111110110011110010010000110100010000110100

111001100011001111101001111011100011000011011100100001101011001111001011000011011100100001101001100100111101100001101011

000011011111100001111110001100111110100111101110001100001101001100100111111110011111101100001101001110101000011010011001

100001111110001100111110100111101110001100001101101001111001101110011111101100111100111001001111011001111001110101001111

110111000011111100011001111111100111100101100001101110110011110110011110011010110011110010010011110011010011000110011111

101110001100111100111001000011010110011110010110000110100100100111100101100111100111010100111111011000011010010011000010

1

Для 8-и символьного алфавита

[FANO]

Coded:

000001001010011001010011011110111010011000001110110010101000100110111010011000000011000010110111000100110111001110110100

001010001000110010101000111110000111001000010000000111010100011111001101111101101101100011010010101010101111000110110000

110101000001001110000000111011100010110111000111000101100101010010111110110011010011010100100011100111100000101111100100

011010001110011100111110010110011011101100110111001000011010100000101010100101111100101101101111100100101010010111110110

110110111001111011011001100101111101101101101110010011010010011011100011111101001111011101001110000001011110010111010100

100101000000001011110010011010011110101001001110100000001001110000001011101001101101101100001110000001000100001100000111

111000110101101011101001100100111011101011100000010001011100011000011101110011101001110000110000111111110001000011011100

101110010101010101110010110101000101010101011100010010001110001001000111000010111101111001101000100010110011011111000101

100001000100011110011101110111010100010011011010010000011100001010011100001011111001101111101100011011100110001000101101

101010100011011110011011111010001000110000000000110100000110111100110111110000000010011110000001001101100010110000001101

11100011100000011101100001110100000000001111000011100100101101110000100110000101100000001000000100

[HUFFMAN]

Coded:

100110011000110101000110111011111001011001111110110101000001000011101010110011011010011100110000100001110101010110110001

000110111101010000010101001110100000011000000111000001010101001100101110111100011010010010010100010100101100000011100100

000101011001100011101010111001100001011101111100100100100111110111010111001001000010010011101010001101111101000001110001

011101111001010111001111000111011101011101010000011011011001111101001001111101111110110011110111001011100100111110111011

110011110010101100110101110011111011101111001111011111110000101111000010101111001001011111001011111001110101011111101110

100001011011000111010101111111100011110100001000101001100000010101011101111110010011001101100111010111011001100110101001

111010110101100011111000110100100111010101111001100000110000100011101111100101001011111001101010011110111100000011011111

001111010010010100010111111010000010100101000101011011001011101111111001110001001011111001001000000001100000111011110111

110001100101001110100111010111110100101111111100010110001011001110001111000010010101001100101110101101111101101001100110

011011100100001110101001100101011000010001100010100110000011101010011001011001101001010001111111111000111000011000111010

010111000010111100000011100010011011101001110111111111011111001101111100111000011011011000010

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

Для 2-х символьного алфавита

[FANO]

Initial length: 4120 --- Coded length: 4120 --- Coefficient: 1 --- Average code length: 2 --- Dispersion: 0

[HUFFMAN]

Table:

Initial length: 4120 --- Coded length: 4120 --- Coefficient: 1 --- Average code length: 2 --- Dispersion: 0

Для 4-х символьного алфавита

[FANO]

Initial length: 4120 --- Coded length: 3121 --- Coefficient: 1.32009 --- Average code length: 3.0301 --- Dispersion: 1.7923

[HUFFMAN]

Table:

Initial length: 4120 --- Coded length: 3121 --- Coefficient: 1.32009 --- Average code length: 3.0301 --- Dispersion: 1.7923

Для 8-и символьного алфавита

[FANO]

Initial length: 4120 --- Coded length: 1298 --- Coefficient: 3.17411 --- Average code length: 4.58657 --- Dispersion: 1.59233

[HUFFMAN]

Initial length: 4120 --- Coded length: 1293 --- Coefficient: 3.18639 --- Average code length: 4.5689 --- Dispersion: 1.16398

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

std::unordered\_map<std::vector<bool>, wchar\_t> GetDecodeTable(  
 const std::unordered\_map<wchar\_t,  
 std::vector<bool>> &code  
) {  
 std::unordered\_map<std::vector<bool>, wchar\_t> out;  
   
 for (auto &el : code) {  
 out[el.second] = el.first;  
 }  
   
 return out;  
}

std::wstring DecodeMessage(  
 const std::unordered\_map<std::vector<bool>, wchar\_t> &code,  
 const std::wstring &str  
) {  
 std::wstring out;  
 std::vector<bool> c;  
   
 int i = 0;  
 while (i < str.length()) {  
 c.clear();  
 while (!code.contains(c)) {  
 c.push\_back(str[i] - '0');  
 i++;  
 }  
   
 out.push\_back(code.find(c)->second);  
 }  
   
 return out;  
}

int main() {  
 setlocale(LC\_ALL, "");

std::string b\_code = "1101000010010010110100001011010111010001100000101101000010110101110100011000000000100000110100011000000111010000101100101101000010111000110100011000000111010001100000101101000010110101110100001011101100101100001000001101000010110010110100001011100011010000101101111101000010110110110100001011000011010000101110110010110000100000110100001011101011010001100000001101000110001111110100011000010111010001100000101101000010110101110100001011101100100000110100001011100000100000110100001011001111010001100000111101000010110100110100001011010111010000101110110010000011010000101111011101000010110000001000001101000110000000110100001011000011010000101101111101000010111101110100011000101111010000101101010010000011010000101110111101000010110000110100001011010011010001100010110010111000100000110100001010001011010000101111100010000011010000101101101101000010110000110100001011101111010000101111101101000010110001110100001011110111010001100010111101000010111100001000001101000110000010110100001011111011010000101111011101000010110101110100001011110111010001100011001101000010111010110100001011100011010000101111000010000011010000101100111101000010111110110100001011101111010000101111101101000110000001110100001011101011010000101111101101000010111100001011000010000011010001100000101101000010111110001000001101000010110011110100011000000011010001100000111101000010110001110100011000101111010000101111000010000011010000101100011101000010110000110100011000000111010000101111101101000010110010110100011000101111010000101111000010000011010001100000001101000010110000110100011000000111010000101110101101000010110000110100011000001011010000101111101101000010111100001000001101000110000000110100001011000011010001100000011101000010111111110100001011010111010000101100101101000010110000110100001011101100100000110100001011111011010000101111010010000011010001100000011101000010110010110100001011111011010001100011100010000011010000101100011101000010111110110100001011010111010000101100101101000110000011110100011000111000100000110100001011111111010000101101011101000110000001110100001011010111010000101111011101000010111010110100011000001100101110001000001101000010100100110100001011111011010000101111011101000010110000110100011000000011010000101110000010000011010001100001111101000110000011110100011000001011010001100011000010000011010000101101111101000010110000110100001011110011010000101101011101000110000010110100001011110111010000101111100010000011010000101111001101000010111000110100001011001111010000101100001101000010111011110100001011100000100000110100011000000111010000101110101101000010110010110100001011111011010000101101111101000110001100001000001101000010111110110100001011001111010001100000001101000010111110110100001011110011010000101111011101000110001011110100001011010100100000110100001011000111010000101101011101000010111011110100011000101111010000101101010010000011010001100001011101000010111011110100001011111011010000101111111101000110001100110100011000111100100000110100011000000111010000101111011101000010110101110100001011001111010000101100000010110000100000110100001011111011010000101100011101000010111000110100001011101111010001100011001101000010111101110100001011111000100000110100011000000111010001100010111101000010111111110100001011000011010000101100101101000110001000110100001011100011010000101101011101000110000001110100011000111100100000110100001011110111010000101100000010000011010001100000101101000110000000110100001011111011010001100000101101000110000011110100001011000011010001100000001101000110001011001011000010000011010000101111011101000010110000001000001101000110000011110100001011101111010000101110001101000110000110110100011000001100101100001000001101000010111101110100001011000000100000110100011000110111010000101110101101000010111000110100001011111111010000101100001101000010110110110100001011100000101100001000001101000010111011110100001011111011010001100010001101000010110000110100001011010011010000101101011101000010111001001000001101000010111000001000001101000010111111110100011000000011010000101111101101000110000101110100001011111011010000101101101101000010111000110100011000010100101110";  
   
 std::vector<int> k**{**2, 4, 8**}**;  
 for (auto &i : k) {  
 auto msg = utf8\_to\_utf16(ParseCode(b\_code, i));  
 auto table = GetTable(GetFanoCode(ParseString(msg)));  
 auto coded = CodeMessage(table, msg);  
 std::wcout << "[FANO]\n" << "Table:\n" << table << "Message:\n" << msg << "\nCoded:\n" << coded  
 << "\nInitial length: " << b\_code.length() << " --- Coded length: " << coded.length()  
 << " --- Coefficient: " << float(b\_code.length()) / coded.length() << " --- Average code length: "  
 << GetAvg(table, msg) << " --- Dispersion: " << GetDispersion(table, msg) << "\n";  
 std::wcout << "Decoded:\n" << DecodeMessage(GetDecodeTable(table), coded) << "\n";  
 }  
   
 for (auto &i : k) {  
 auto msg = utf8\_to\_utf16(ParseCode(b\_code, i));  
 auto table = GetTable(GetHuffmanCode(ParseString(msg)));  
 auto coded = CodeMessage(table, msg);  
 std::wcout << "[HUFFMAN]\n" << "Table:\n" << table << "Message:\n" << msg << "\nCoded:\n" << coded  
 << "\nInitial length: " << b\_code.length() << " --- Coded length: " << coded.length()  
 << " --- Coefficient: " << float(b\_code.length()) / coded.length() << " --- Average code length: "  
 << GetAvg(table, msg) << " --- Dispersion: " << GetDispersion(table, msg) << "\n";  
 std::wcout << "Decoded:\n" << DecodeMessage(GetDecodeTable(table), coded) << "\n";  
 }  
 return 0;  
}

Промежуточные результаты работы программы приведены в ходе работы. Декодированные сообщения (вместо двоичных кодов сразу выводятся их символьные представления):

Для 2-х символьного алфавита

♥☺ ☻☺ ☻♥☺ ☻♥☺☺♥☺ ☺☻ ☻♥☺ ☻♥☺☺♥☺ ☺☻ ☻ ♥☺ ☺☻ ☺♥☺ ☻♥ ☻♥☺ ☻♥☻ ♥☺ ☺☻ ☺♥☺ ☺☻ ☻♥☺ ☻♥☺☺♥☺ ☻♥☻♥ ☻♥ ☻ ♥☺ ☻♥ ☻♥☺

☻♥☻ ♥☺ ☻♥☺♥♥☺ ☻♥☺☻♥☺ ☻♥ ♥☺ ☻♥☻♥ ☻♥ ☻ ♥☺ ☻♥☻☻♥☺ ☺☻ ♥☺ ☺☻ ♥♥♥☺ ☺☻ ☺☺♥☺ ☺☻ ☻♥☺ ☻♥☺☺♥☺ ☻♥☻♥ ☻ ♥☺ ☻♥☻ ☻ ♥☺

☻♥ ♥♥☺ ☺☻ ♥♥☺ ☻♥☺ ♥☺ ☻♥☺☺♥☺ ☻♥☻♥ ☻ ♥☺ ☻♥♥☺♥☺ ☻♥ ☻ ♥☺ ☺☻ ♥☺ ☻♥ ♥☺ ☻♥☺♥♥☺ ☻♥♥☺♥☺ ☺☻ ☻♥♥☺ ☻♥☺☺ ☻ ♥☺ ☻♥☻♥

♥☺ ☻♥ ♥☺ ☻♥☺ ♥☺ ☺☻ ☻♥ ☻♥☻ ☻ ♥☺ ☻☻ ☻♥☺ ☻♥♥☻ ☻ ♥☺ ☻♥☺☻♥☺ ☻♥ ♥☺ ☻♥☻♥♥☺ ☻♥♥☻♥☺ ☻♥ ☺♥☺ ☻♥♥☺♥☺ ☺☻ ☻♥♥☺ ☻♥♥ ☻

♥☺ ☺☻ ☻♥☺ ☻♥♥☻♥☺ ☻♥♥☺♥☺ ☻♥☺☺♥☺ ☻♥♥☺♥☺ ☺☻ ♥ ♥☺ ☻♥☻☻♥☺ ☻♥☻ ♥☺ ☻♥♥ ☻ ♥☺ ☻♥ ♥♥☺ ☻♥♥☻♥☺ ☻♥☻♥♥☺ ☻♥♥☻♥☺ ☺☻ ☺♥☺

☻♥☻☻♥☺ ☻♥♥☻♥☺ ☻♥♥ ☻♥ ☻ ♥☺ ☺☻ ☻♥☺ ☻♥♥☻ ☻ ♥☺ ☻♥ ♥♥☺ ☺☻ ♥☺ ☺☻ ♥♥☺ ☻♥ ☺♥☺ ☺☻ ☻♥♥☺ ☻♥♥ ☻ ♥☺ ☻♥ ☺♥☺ ☻♥ ♥☺ ☺

☻ ☺♥☺ ☻♥♥☻♥☺ ☻♥ ☻♥☺ ☺☻ ☻♥♥☺ ☻♥♥ ☻ ♥☺ ☺☻ ♥☺ ☻♥ ♥☺ ☺☻ ☺♥☺ ☻♥☻☻♥☺ ☻♥ ♥☺ ☺☻ ☻♥☺ ☻♥♥☻♥☺ ☻♥♥ ☻ ♥☺ ☺☻ ♥☺

☻♥ ♥☺ ☺☻ ☺♥☺ ☻♥♥♥♥☺ ☻♥☺☺♥☺ ☻♥ ☻♥☺ ☻♥ ♥☺ ☻♥☻♥ ☻ ♥☺ ☻♥♥☻♥☺ ☻♥♥☺ ☻ ♥☺ ☺☻ ☺♥☺ ☻♥ ☻♥☺ ☻♥♥☻♥☺ ☺☻ ♥☻ ☻ ♥☺ ☻♥ ☺

♥☺ ☻♥♥☻♥☺ ☻♥☺☺♥☺ ☻♥ ☻♥☺ ☺☻ ♥♥☺ ☺☻ ♥☻ ☻ ♥☺ ☻♥♥♥♥☺ ☻♥☺☺♥☺ ☺☻ ☺♥☺ ☻♥☺☺♥☺ ☻♥♥☺♥☺ ☻♥☻☻♥☺ ☺☻ ♥ ☻♥☻ ☻ ♥☺ ☻☻☺ ♥☺

☻♥♥☻♥☺ ☻♥♥☺♥☺ ☻♥ ♥☺ ☺☻ ♥☺ ☻♥☻ ☻ ♥☺ ☺☻ ☺♥♥☺ ☺☻ ♥♥☺ ☺☻ ☻♥☺ ☺☻ ♥ ☻ ♥☺ ☻♥☺♥♥☺ ☻♥ ♥☺ ☻♥♥ ♥☺ ☻♥☺☺♥☺ ☺☻ ☻♥☺

☻♥♥☺♥☺ ☻♥♥☻ ☻ ♥☺ ☻♥♥ ♥☺ ☻♥☻ ♥☺ ☻♥ ♥♥☺ ☻♥ ♥☺ ☻♥☻♥♥☺ ☻♥☻ ☻ ♥☺ ☺☻ ☺♥☺ ☻♥☻☻♥☺ ☻♥ ☻♥☺ ☻♥♥☻♥☺ ☻♥☺♥♥☺ ☺☻ ♥ ☻

♥☺ ☻♥♥☻♥☺ ☻♥ ♥♥☺ ☺☻ ♥☺ ☻♥♥☻♥☺ ☻♥♥ ♥☺ ☻♥♥☺♥☺ ☺☻ ☻♥♥☺ ☻♥☺☺ ☻ ♥☺ ☻♥ ☺♥☺ ☻♥☺☺♥☺ ☻♥☻♥♥☺ ☺☻ ☻♥♥☺ ☻♥☺☺ ☻ ♥☺ ☺☻ ☺☺

♥☺ ☻♥☻♥♥☺ ☻♥♥☻♥☺ ☻♥♥♥♥☺ ☺☻ ♥ ♥☺ ☺☻ ♥♥ ☻ ♥☺ ☺☻ ☺♥☺ ☻♥♥☺♥☺ ☻♥☺☺♥☺ ☻♥ ♥♥☺ ☻♥ ☻♥ ☻ ♥☺ ☻♥♥☻♥☺ ☻♥ ☺♥☺ ☻♥☻ ♥☺

☻♥☻♥♥☺ ☺☻ ♥ ♥☺ ☻♥♥☺♥☺ ☻♥♥☻ ☻ ♥☺ ☺☻ ☺♥☺ ☺☻ ☻♥♥☺ ☻♥♥♥♥☺ ☻♥ ♥☺ ☻♥ ☻♥☺ ☺☻ ☻ ♥☺ ☻♥☻ ♥☺ ☻♥☺☺♥☺ ☺☻ ☺♥☺ ☺☻ ♥♥ ☻ ♥☺

☻♥♥☺♥☺ ☻♥ ☻ ♥☺ ☺☻ ☻♥☺ ☺☻ ♥☺ ☻♥♥☻♥☺ ☺☻ ☻♥☺ ☺☻ ♥♥☺ ☻♥ ♥☺ ☺☻ ♥☺ ☺☻ ☻♥ ☻♥ ☻ ♥☺ ☻♥♥☺♥☺ ☻♥ ☻ ♥☺ ☺☻ ♥♥☺

☻♥☻♥♥☺ ☻♥☻ ♥☺ ☺☻ ☺☻♥☺ ☺☻ ♥ ☻♥ ☻ ♥☺ ☻♥♥☺♥☺ ☻♥ ☻ ♥☺ ☺☻ ♥☺♥☺ ☻♥☻☻♥☺ ☻♥☻ ♥☺ ☻♥♥♥♥☺ ☻♥ ♥☺ ☻♥☺☻♥☺ ☻♥☻ ☻♥ ☻

♥☺ ☻♥☻♥♥☺ ☻♥♥☻♥☺ ☺☻ ☻ ♥☺ ☻♥ ♥☺ ☻♥☺ ♥☺ ☻♥☺☺♥☺ ☻♥☻☺ ☻ ♥☺ ☻♥☻ ☻ ♥☺ ☻♥♥♥♥☺ ☺☻ ♥☺ ☻♥♥☻♥☺ ☺☻ ☺☺♥☺ ☻♥♥☻♥☺ ☻♥☺☻

♥☺ ☻♥☻ ♥☺ ☺☻ ☺☺ ☻♥☻

Для 4-х символьного алфавита

♂♂☻♀☻ ☻

☻♫☻

♂♣☻

♂♫☻

♂ ☻ ☻

♂♫

♂♣☻

☻♫☻

♂ ☻

♂ ☻ ☻

♣☻♫ ☻

Для 8-и символьного алфавита

Ветер свистел, визжал, кряхтел и гудел на разные лады. То жалобным тоненьким голоском, то грубым басовым раскатом распев

ал он свою боевую песенку. Фонари чуть заметно мигали сквозь огромные белые хлопья снега, обильно сыпавшиеся на тротуары, на улицу, на экипажи, лошадей и прохожих.

Вывод:

Вывод: таким образом, эффективнее кодируются последовательности длиной в 8 бит. 2-битные символы сжать практически невозможно, поскольку при таком маленьком алфавите и относительно большом кол-ве символов будет наблюдаться низкая энтропия, при которой невозможно сжатие методами Хаффмана и Шеннона-Фано. Стоит отметить, что так как исходное сообщение было записано в кодировке utf-8, в которой один символ может кодироваться различным числом байт (русские символы кодируются двумя байтами), а обработка сообщений в программе производится в кодировке utf-16, у которой все символы одной длины, то итоговый коэффициент сжатия вышел примерно вдвое больше, чем, если бы каждый символ был закодирован одним байтом (вышло чисто случайно – без использования utf-16 не получалось вывести русский текст в консоль). Помимо всего прочего, на последнем наборе данных код Хаффмана получился немного оптимальнее кода Шеннона-Фано. Оба алгоритма имеют одинаковую вычислительную сложность декодирования (для этого достаточно в процессе кодирования коды, полученные обоими алгоритмами записать в виде хеш-таблицы). Алгоритм Хаффмана при кодировании требует дополнительной памяти, в отличие от алгоритма Шеннона-Фано, однако выполняется итеративно за линейное время, в то время, как алгоритм Шеннона Фано имеет квадратичную вычислительную сложность и выполняется рекурсивно.