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-233
Ситников Алексей Павлович
Проверил: Твердохлеб Виталий
Викторович

Белгород 2025 г.

Ход работы

Задание 1.

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

Бинарная последовательность уже ужата, в бинарной последовательности используются только два символа, значит для кодирования символа нужен один бит, это уже минимум.

Задание 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:

```
int main(){
    FILE *f = fopen("text.txt", "r");
    int size;
    scanf("%d", &size);
    char letters[size+1];
    char D[4120][size+1];
    int count = 0;
    while (fread(letters, 1, size, f) == size) { // Считываем по 2 символа
        letters[size] = '\0';
        for(int i = 0; i < size+1; i++){
            D[count][i] = letters[i];
        }
        count++;
    }

    fclose(f);
    for(int i = 0; i < count; i++){
        printf("%s ", D[i]);
    }

    return 0;
}
```

Вывод программы:

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
8
11010000 10010010 11010000 10110101 11010001 10000010 11010000 10110101 11010001 10000000 00100000 11010001 10000001 110
10000 10110010 11010000 10111000 11010001 10000001 11010001 10000010 11010000 10110101 11010000 10111011 00101100 001000
00 11010000 10110010 11010000 10111000 11010000 10110111 11010000 10110110 11010000 10110000 11010000 10111011 00101100
00100000 11010000 10111010 11010001 10000000 11010001 10001111 11010001 10000101 11010001 10000010 11010000 10110101 110
10000 10111011 00100000 11010000 10111000 00100000 11010000 10110011 11010001 10000011 11010000 10110100 11010000 101101
01 11010000 10111011 00100000 11010000 10111101 11010000 10110000 00100000 11010001 10000000 11010000 10110000 11010000
10110111 11010000 10111101 11010001 10001011 11010000 10110101 00100000 11010000 10111011 11010000 10110000 11010000 101
10100 11010001 10001011 00101110 00100000 11010000 10100010 11010000 10111110 00100000 11010000 10110110 11010000 101100
00 11010000 10111011 11010000 10111110 11010000 10110001 11010000 10111101 11010001 10001011 11010000 10111100 00100000
11010001 10000010 11010000 10111110 11010000 10111101 11010000 10110101 11010000 10111101 11010001 10001100 11010000 101
11010 11010000 10111000 11010000 10111100 00100000 11010000 10110011 11010000 10111110 11010000 10111011 11010000 101111
10 11010001 10000001 11010000 10111010 11010000 10111110 11010000 10111100 00101100 00100000 11010001 10000010 11010000
10111110 00100000 11010000 10110011 11010001 10000000 11010001 10000011 11010000 10110001 11010001 10001011 11010000 101
11100 00100000 11010000 10110001 11010000 10110000 11010001 10000001 11010000 10111110 11010000 10110010 11010001 100010
11 11010000 10111100 00100000 11010001 10000000 11010000 10110000 11010001 10000001 11010000 10111010 11010000 10110000
11010001 10000010 11010000 10111110 11010000 10111100 00100000 11010001 10000000 11010000 10110000 11010001 10000001 110
10000 10111111 11010000 10110101 11010000 10110010 11010000 10110000 11010000 10111011 00100000 11010000 10111110 110100
00 10111101 00100000 11010001 10000001 11010000 10110010 11010000 10111110 11010001 10001110 00100000 11010000 10110001
11010000 10111110 11010000 10110101 11010000 10110010 11010001 10000011 11010001 10001110 00100000 11010000 10111111 110
10000 10110101 11010001 10000001 11010000 10110101 11010000 10111101 11010000 10111010 11010001 10000011 00101110 001000
00 11010000 10100100 11010000 10111110 11010000 10111101 11010000 10110000 11010001 10000000 11010000 10111000 00100000
11010001 10000111 11010001 10000011 11010001 10000010 11010001 10001100 00100000 11010000 10110111 11010000 10110000 110
10000 10111100 11010000 10110101 11010001 10000010 11010000 10111101 11010000 10111110 00100000 11010000 10111100 110100
00 10111000 11010000 10110011 11010000 10110000 11010000 10111011 11010000 10111000 00100000 11010001 10000001 11010000
10111010 11010000 10110010 11010000 10111110 11010000 10110111 11010001 10001100 00100000 11010000 10111110 11010000 101
10011 11010001 10000000 11010000 10111110 11010000 10111100 11010000 10111101 11010001 10001011 11010000 10110101 001000
00 11010000 10110001 11010000 10110101 11010000 10111011 11010001 10001011 11010000 10110101 00100000 11010001 10000101
11010000 10111011 11010000 10111110 11010000 10111111 11010001 10001100 11010001 10001111 00100000 11010001 10000001 110
10000 10111101 11010000 10110101 11010000 10110011 11010000 10110000 00101100 00100000 11010000 10111110 11010000 101100
01 11010000 10111000 11010000 10111011 11010001 10001100 11010000 10111101 11010000 10111110 00100000 11010001 10000001
11010001 10001011 11010000 10111111 11010000 10110000 11010000 10110010 11010001 10001000 11010000 10111000 11010000 101
10101 11010001 10000001 11010001 10001111 00100000 11010000 10111101 11010000 10110000 00100000 11010001 10000010 110100
01 10000000 11010000 10111110 11010001 10000010 11010001 10000011 11010000 10110000 11010001 10000000 11010001 10001011
00101100 00100000 11010000 10111101 11010000 10110000 00100000 11010001 10000011 11010000 10111011 11010000 10111000 110
10001 10000110 11010001 10000011 00101100 00100000 11010000 10111101 11010000 10110000 00100000 11010001 10001101 110100
00 10111010 11010000 10111000 11010000 10111111 11010000 10110000 11010000 10110110 11010000 10111000 00101100 00100000
11010000 10111011 11010000 10111110 11010001 10001000 11010000 10110000 11010000 10110100 11010000 10110101 11010000 101
11001 00100000 11010000 10111000 00100000 11010000 10111111 11010001 10000000 11010000 10111110 11010001 10000101 110100
00 10111110 11010000 10110110 11010000 10111000 11010001 10000101 00101110
Process finished with exit code 0
```

Задание 3, 4 и 5

Рассматривая каждую цепочку (2, 4 и 8 символов длиной) как отдельный символ, построить коды по методу Хаффмана и Шеннона-Фано. Составить последовательности из полученных кодов символов для каждого случая. По результатам работы в п.3 сделать выводы по поводу полученных результатов для каждого из методов (простота, скорость, полученные результаты (рассчитать коэффициенты сжатия)).

Код, метод Шеннона-Фано:

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
```

```c
typedef struct node{
    int count;
    char letter[15];
    char code[30];
}node;
void sort(int n, node *array[n]){
    for(int i = 0; i < n; i++){
        int is_sort = 1;
        for(int j = n - 1; j > i; j--){
            if(array[j]->count > array[j-1]->count){
                node *temp = array[j];
                array[j] = array[j-1];
                array[j-1] = temp;
                is_sort = 0;
            }
        }
        if(is_sort){
            break;
        }
    }
}
void putChar(char *arr, char c){
    char *begin = arr;
    while (*begin!='\0'){
        begin++;
    }
    *begin = c;
    *(++begin) = '\0';
}
void fano(int size, node *array[size]){
    int sum1 = 0, sum2 = 0;
    node *part1[size];
    node *part2[size];
    int size1 = 0;
    int size2 = 0;
    for(int i = 0; i < size; i++){
        if(sum1 < sum2){
            part1[size1++] = array[i];
            sum1+=array[i]->count;
            putChar(array[i]->code, '1');
        }
        else{
            part2[size2++] = array[i];
            sum2+=array[i]->count;
            putChar(array[i]->code, '0');
        }
    }
    if(size1 > 1){
        fano(size1, part1);
    }
    if(size2 > 1){
        fano(size2, part2);
    }
}

int main(){
    clock_t start_time = clock();
    FILE *f = fopen("text.txt", "r");
    int size;
    scanf("%d", &size);
    char letters[size+1];
    char D[4120][size+1];
    int count = 0;
    while (fread(letters, 1, size, f) == size) { // Считываем по 2 символа
```

```
        letters[size] = '\0';
        for(int i = 0; i < size+1; i++){
            D[count][i] = letters[i];
        }
        count++;
    }
    fclose(f);
    char *begin;
    node *data[count];
    int size_data = 0;
    for(int i = 0; i < count; i++){
        int flag = 0;
        for(int j = 0; j < size_data; j++){
            int flag_isEq = 1;
            begin = D[i];
            char *b = data[j]->letter;
            while (*begin != '\0'){
                if(*begin != *b){
                    flag_isEq = 0;
                    break;
                }
                begin++;
                b++;
            }
            if(flag_isEq){
                flag = 1;
                data[j]->count++;
                break;
            }
        }
        if(flag == 0){
            data[size_data] = (node *)malloc(sizeof(node));
            char *b = data[size_data]->letter;
            begin = D[i];
            while (*begin!='\0'){
                *b = *begin;
                b++;
                begin++;
            }
            *b = '\0';
            data[size_data]->count = 1;
            data[size_data]->code[0] = '\0';
            size_data++;
        }
    }
    sort(size_data, data);
    fano(size_data, data);

    char answer[count * size * 10];
    int size_ans = 0;
    char *ansBg = answer;
    for(int i = 0; i < count; i++){
        for(int j = 0; j < size_data; j++){
            int flag = 1;
            char *b = data[j]->letter;
            begin = D[i];
            while (*begin != '\0'){
                if(*begin != *b){
                    flag = 0;
                }
                begin++;
                b++;
            }
            if(flag){
```

```
                begin = data[j]->code;
                while (*begin!='\0'){
                    *ansBg = *begin;
                    ansBg++;
                    size_ans++;
                    begin++;
                }
            }
        }
    }
    *ansBg = '\0';
    printf("%s", answer);
    printf("\nK = %.1f", 4210.0/size_ans);
    for(int i = 0; i < size_data; i++){
        free(data[i]);
    }
    clock_t end_time = clock();
    double time_taken = (double)(end_time - start_time) / CLOCKS_PER_SEC;
    printf("\nTime: %f\n", time_taken);
    return 0;
}
```

Выводы программы:

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
2
10110000011100011011000001101111101100110100000110110000011011111011001101000000000100001011001101000011101100000011000001
10110000011001001011001101000011101100110100000110110000011011111011000001100110000110000001000010110000011000011011000
01100100101100000110111010110000001101101101100000110000010110000011001100001100000010000101100000110010110110011010000000
10110011010010101011001101001111101100110100000110110000011011111011000001100110000100001011000001100100001000010110000
01100010101100110100001010110000001101100101100000110111110110000011001100001000010110000011010111011000001100000000010000
10110011010000001011000001100000101100000110111010110000001101011101100011010001101011000001101111000100001011000001100110
10110000011000001011000001101100101100110100011000011001000100001011000001010001101100000110100100010000101100000110110 1
10110000011000001011000001100110101100000110100110110000001100011101100000110101110110001101000110110000011010000000010000
10110011010000011011000001101001101100000110101110110000001101111101100000110101110110011010001011000001100101101100000
01100100101100000110100000010000101100000110001010110000011010011011000001100110101100000110100110110011010000111010000
01100101101100000110100110110000011010000001100000010000101100110100000110110000011010010001000010110000011000101011001 1
01000000101100110100001010110000011000111011001101000110101100000110100000010000101100000110001110110000011000010110011
01000011101100000110100110110000011000011011001101000110101100000110100000010000101100110100000010110000011000010110011
01000011101100000110010110110000011000001011001101000001101100000110100110110000011010000001000010110011010000010110000
01100000101100110100001110110000011010101011000001101111101100000110000110110000011000010110000011001100001000010110000
01101001101100000110101100010000101100110100001110110000011000011011000001101001101100110100100010000101100000110001 1
10110000011010011011000001101111101100000110000110110011010000101011001101001000100010000101100000110101010110000011111
10110011010000111011000001101111101100000110101111011000001100101101100110100010000101100000110100110101000100001011100101100000
01101001101100000110101111011000001100000101100110100000010110000011001000010000101100110100011101011001101000010110011 1
01000001101100110100100000010000101100000110111010110000001100000101100000110100010110000011011111011001101000001101100000
01101011101100000110100100010000101100000110100010110000001100100101100000110111101100110100010100001000010110000011001101100000
01100100000100001011001101000011101100000110010110110000001100001101100000110100110110000011011101011001101001000000010000
10110000011010011011000001100010101100110100000010110000011010011011000001101000101100000110101110110011010001101010110000
01101111000100001011000001100011101100000110111110110000001100110101100110100011010110000011011110001000010110011010011111
10110000011001101011000001101001101100000110101010110011010010001011001101001010000100001011001101000011101100000011101011
10110000011011111011000001100010101100000110000000011000000100001011000001101001101100000110001110110000011001001011000000
01100110101100110100100010110000011010110110000011010010001000010110011010000111011001101001010000100001011000000110101010110000
01100000101100000110000110110011010001001011000001100100101100000110111110110011010000111011001101001010000100001011000000
01101011101100000110000000010000101100110100000011011001101000000101100000110100110110011010000011011001101000010101100000
01100000101100110100000010110011010001100001100000001000010110000011010111011000001100000000100001011001101000010101100000
01100110101100000110010010110011010011011001101000010000110000001000010110000011010111011000001100000000100001011001101100011
01001011101100000110010110110000011001001011000001101010101100000110000010110000011011011000001100100000110000010000000000
10110000011001101011000001101001101100110100010010110000011000001011000001101100101100000110111101100000110011100010000000
10110000011001000010000101100000110101010110011010000001011000001101001101100110101001111101100000110100110110000001101101
10110000011001001011001101000111100011001
K = 1.0
Time: 2.909000
```

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
4
0001000011111000010011001010000010001101110000100110010100000100011010011101000000010001100100000100110111000010011001100
00010001100100000001000110111000010011001010000100110110111001011111010000001001101110000100110011000010011010110001001101
111000100110100000010011011011100101111101000000100110011110000001000110100000001000110001000000100011001010000001000110111000
010011001010000100110110111010000001001100110111010000001001101010000001000110101000010011001111100010011001010000100110110
11101000000100110000000010011010011101000000010001101000000100110100000010011010110001001100000000100011011000010011001010111
010000010011011000010011010000010011001111100001000110110111001110111101000000100011110111000010011001110111010000001001101
111000100110100000010011011000010011001110000100110010000001001100000000100011011000010011001011111010000000100011011100001
00110011100001001100000000100110010100001001100000000010001100101100010011001111000010011001100001001100101111101000001001
10101000010011001110000100110110000100110011100000010001100100000010011001111000010011001110000100110010111110010111110100
0000100011011100001001100111011101000000100110101000000100011010000000100011010100001001100100000001000110110000100110010111
110100000010011001000000100110100000001000110010000001001100111000010011011100000010001101100001001100101111101000000010001101
00000100110100000001000110010000001001100111100001001101000000010001101110000100110011100001001100101111101000000001000110100
00010011010000000100011001000000100110001000010011001010000100110111000010011010000001001101101110100000010011001110000010011
0000111010000000100011001000000100110111000010011001110000010001100111011101000000100110010000001001100111000010011001010000
100110111000000100011010100000100011001110111010000001001100010000100110010100000100011001000001001100101000010011000000001
00110011110000001000110101011100111011101000000100011110011111000100110011100001001100000000100110100000001000110100000010011
001101110100000001000110101100001000110101000001000110111000001000110010111110100000010011010110001001101000000100110010110
0010011001010000001000110111000010011000000001001100111011101000000100110010110001001100110000100110101000010011010000001001
1011000010011001101110100000001000110010000010011001111000010011011100001001100111000010011010110000100011001011111010000
010011001110000100110101000000100011010000001001100111000010011001011000100110000000010001101100001001100101011101000000100
110010000001001100101000010011011000000100011011000010011001010111010000000100011001010000100110110000100110011100001001100
010000010001100101100001000110001011101000000010001100100000010011000000001001100101000010011010100001001101001110010111110
100000100110011100001001100100000010011001100001001101100000010001100101100010011000000001001100111011101000000100011001000
00010001101100001001100010000010011010000001001101110000001000110011000010011001100001001100101000000100011001000000010001100
0101110100000010011000000001001101001110100000001000110111000000100011010000001001100111000000100011011100000010001101010000100
11010000000100011010000000100011011011100101111101000000100110000000010011010011101000000010001101010000100110110000100110011
0000010001101111000010001101010111001011111010000001001100000000100110100111010000000100011000000001001100111100001001100110
0001001100010000010011010000001001101111000100110011011100101111101000000100110110000100110011100000010001100110000100110100
0001001100111110001001100101000010011000111110100000010011001101110100000010011000100000010001101000000100110011100000100011
0010100000100110011100001001101111000100110011000000100011001010111001110
K = 1.2
Time: 1.168000

Process finished with exit code 0
```

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
8
0011111100010100100101110001010010001110011001001011000000101100011010100101100100101110001010000011000111101100000101100
011010001011110001010110011110000110001111011000011101010001110010011011110001111110010111000101000001100110000110101100
000100101001010100001101100010100000110011000001000001111011001000111000011110001011110000100010001010000101001100000110
000111100001101101000101001011011100001110110011100110000101011001111000011000011100000110100001000100010100001111101100
100101110001110000010000010100000100010001011100111010001101000111110110000010010001110000011000011100100101100001110100
011100001111100111101100100101110001110011000001001010001110010010101000011010100010100001111101100000110100011110100101
100001110000010110100010100001111101100100011100001111010010110000111010001111010010111000111000011111011001000111000011
11010010110000110110001010000010110001111000011001100001110000001000110010010110000001011000111001000100110110000011010001
110000101000000101101001010101000100110110000110110001010010010110000101000000100000111010100101010101101110000001110100111
000001000001111010001110000110101100100111111110010101010010111010001011111000010111100011110001111100010100100101110000
1000001110011000011111000110100001001000111100001100001101011001001011000011101000010110001110000101111010001011111100001
1100000100101000111000011100001111100001000100010100001010011000001101000101000000110010001010000101001100100011111000110
0001110000110110100010111100110111110010010110000010000010100000100100011110011110110000111000000110100011010000110010001
0111000100000111001100100101100100010100001101100011110000101101000101010011010001010010010110010011011111100000100000111
1011001001011101000111000011100100101110100101010001111010001110010001010001111011000000100000111101100100101010000110000
1101010001101111001010100111101100000100000111101100100101111100111010001101000110110001111000101011001101001111011000000
1100001110010001010100111100001101100010100000100111111000011010110000110110100011100001110010001111100111000010101100110
10100011111101101
K = 2.0
Time: 1.699000

Process finished with exit code 0
```

Метод Хаффмана:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

typedef struct node{
    int count;
    char letter[15];
    char code[30];
    struct node* left;
    struct node* right;
}node;
void sort(int n, node **array){
    for(int i = 0; i < n; i++){
        int is_sort = 1;
        for(int j = n - 1; j > i; j--){
            if(array[j]->count > array[j-1]->count){
                node *temp = array[j];
                array[j] = array[j-1];
                array[j-1] = temp;
                is_sort = 0;
            }
        }
        if(is_sort){
            break;
        }
    }
}
void haffman(node *let, char* code, char *endCode){
    if(let->letter[0] == '\0'){
        *endCode = '0';
        *(endCode+1) = '\0';
        haffman(let->left, code, endCode+1);
        *endCode = '1';
        *(endCode+1) = '\0';
        haffman(let->right, code, endCode+1);
    }
    else{
        char *begin = let->code;
        char *begin_code = code;
        while (*begin_code!='\0'){
            *begin = *begin_code;
            begin++;
            begin_code++;
        }
        *begin = '\0';
    }
}
void huffmanTreeToArray(node **data, int* count, node *root){
    if(root->letter[0] != '\0'){
        data[*count] = root;
        (*count)+=1;
    }
    else{
        huffmanTreeToArray(data, count, root->left);
        huffmanTreeToArray(data, count, root->right);
    }
}
void free_data(node *root){
    if(root->letter[0] == '\0'){
        free_data(root->left);
        free_data(root->right);
    }
```

```
    free(root);
}


int main(){
    clock_t start_time = clock();
    FILE *f = fopen("text.txt", "r");
    int size;
    scanf("%d", &size);
    char letters[size+1];
    char D[4120][size+1];
    int count = 0;
    while (fread(letters, 1, size, f) == size) { // Считываем по 2 символа
        letters[size] = '\0';
        for(int i = 0; i < size+1; i++){
            D[count][i] = letters[i];
        }
        count++;
    }

    fclose(f);
    char *begin;



    node *data[count];
    int size_data = 0;
    for(int i = 0; i < count; i++){
        int flag = 0;
        for(int j = 0; j < size_data; j++){
            int flag_isEq = 1;
            begin = D[i];
            char *b = data[j]->letter;
            while (*begin != '\0'){
                if(*begin != *b){
                    flag_isEq = 0;
                    break;
                }
                begin++;
                b++;
            }
            if(flag_isEq){
                flag = 1;
                data[j]->count++;
                break;
            }
        }
        if(flag == 0){
            data[size_data] = (node *)malloc(sizeof(node));
            char *b = data[size_data]->letter;
            begin = D[i];
            while (*begin!='\0'){
                *b = *begin;
                b++;
                begin++;
            }
            *b = '\0';
            data[size_data]->count = 1;
            data[size_data]->code[0] = '\0';
            size_data++;
        }
    }
    sort(size_data, data);
    while (size_data != 1){
```

```
        node *temp = malloc(sizeof(node));
        temp->count = data[size_data-1]->count + data[size_data-2]->count;
        temp->letter[0] = '\0';
        temp->code[0] = '\0';
        temp->left = data[size_data-1];
        temp->right = data[size_data-2];
        data[size_data-2] = temp;
        size_data--;
        sort(size_data, data);
    }
    node *root = data[0];
    char code[300];
    haffman(root, code, code);
    size_data = 0;
    huffmanTreeToArray(data, &size_data, root);


    char answer[count * size * 10];
    int size_ans = 0;
    char *ansBg = answer;
    for(int i = 0; i < count; i++){
        for(int j = 0; j < size_data; j++){
            int flag = 1;
            char *b = data[j]->letter;
            begin = D[i];
            while (*begin != '\0'){
                if(*begin != *b){
                    flag = 0;
                }
                begin++;
                b++;
            }
            if(flag){
                begin = data[j]->code;
                while (*begin!='\0'){
                    *ansBg = *begin;
                    ansBg++;
                    size_ans++;
                    begin++;
                }
            }
        }
    }

    *ansBg = '\0';
    printf("%s", answer);

    printf("\nK = %.1f", 4210.0/size_ans);
    free_data(root);
    clock_t end_time = clock();
    double time_taken = (double)(end_time - start_time) / CLOCKS_PER_SEC;

    printf("\ntime: %f\n", time_taken);
    return 0;
}
```

Выводы программы:

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
2
1001111100011100100111110010010110011101001111001001111100100101100111010011111111001111100111010011101100111110010110011111
1001111100100011100111010011110110011101001111001001111100100101100111110010001011001011100111110011111001011001001111
0010001110011111001001101001111100100100100111110010111110011111001000101100101111001111100111110010000010011101001111
1001110100111010100111010011010110011101001111001001111100100101100111110010001011001111100111110010001111001111001111
0010111010011101001111101001111100100111001111100100101100111110010001011001111100111110010100110011110010111111001111
1001110100111111100111110010111110011111001001101001111100101001100111010011001010011111001001011100111110011111100100010
1001111100101111100111110010011110011101001100101100100011001111100111110000110010011111001010001100111110011111100100100
1001111100101111100111110010001010011111001010001001111100101101100111110010100110011101001100101001111100101111001111001
1001110100111001001111100101000100111110010100110011111001001011001111100101001100111010011100111011100111110000010011111
0010001110011111001010111100111110011111001011101001111100101000100111110010001010011110010100101001111100101000010011101
0010000100111110010100011001111100101101100111010011001010011111001001011001111100101001100111010011100111110010111101001101
0011111100111010011111010011111001010110011101001100101001111100101011110011111001111001011011001111100101111001111001101
0011110110011111001010001001111100101100100111010011001010011111001010111001111100111010011111110011111001011111001111001101
0011110110011111001000001001111100101111001110100111100100111100101000100111110010101110011111001110100111111100111100101
0010111110011101001111011001111100101010100111110010010110011111001010011110011111001011111001111001110100111111010011111
0010100010011111001010011100111110011101001111011001111001001011001111100101000100111010011100011001111100111100101101
1001111100101000100111110010010110011111001011001001110100111101001110100111000110011111001111001010100111110010010101
1001110100111011001111100100101100111110010100110011101000001001110100111101000100110011110011110000011110011111
0010100010011111001010011100111110010111100111010011110001011001001111100101000100111010011100111010011110100111101001
0010001110011111001110100111001111100101111100111010011110001011001001111100101000100111010011100111010011110100111101001
1001111100101000100111110010111010011101001111111001111001010001001111100101011100111110010100110011101001100101001111
0010010111001111001111100101101100111110010010110011111001000101001110100110010100111110010010111001111100111010001101
1001111100100010100111110010100010011111001010101001110100111011001110100110101100111110010100110011101100111100101001
1001111100100101100111110010110100111110010111110010111001111100111110010100010011111001011011001111100110100111011001
0010001010011101001110110011111001010011001111100101000110011111001110100111011001110100110010100111110010101010011111
0010111110011111001011001001110100110011100111110010001110011111001110100111011001110100110010100111110010111001111001
0010100110011111001011111100111100111010011110010011101001111111001111100101000100111010011100111001001110100111101001
0010111110011101001111111001110100110010110011101111001111100111100101001100111110010111111001111001110100111101001111
0010001010011111001000111001110100110100100111010011111011001011100111110010100110011111001011111001110100111101001101
0011100110011111001000001001111100100011100111110010101010011111001011110011111001001001001111100100011110010111001111
1001111100100010100111110010100010011101001100111001111001011111001111100100111001111001001011001111100100001110011111
1001111100100011100111110011111001010101001110100111111100111100101000100111010011010110011111001010001001111100100100
1001111100100011100111010011010111001000
K = 1.0
time: 0.776000
```

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
4
1001001011001100100111100100100001101110010011110010010000110101110001100001101000100111111001001111110110000110100010000
0110111001001111001001001111111110000111111000110011111100100111111011001111001110011100111001110100100111101001111111
0000111111000110011110011100100001101011000011010010111000011010010010000110111001001111001001001111111110001100111110
1110001100111100101010000110100101010011110010110110011110010010011111111100011001111010011110111000110000110101100111
0110011110011101110011111010000110111110011110010011000110011111111001111011001111001010110110000110111111000011011000110
1001110011001001111001101100011001111001110101001111011001111111100111001101001111000100111101000011011110011111001111
1100011000011011100100111100110100111110100111100100100111101000011010011111001110011100100111111010011100111001111110001
1001111001010100111001101001111111100111001101000011010001001111001110010011110011010011100111111000011111100011000001
1011100100111001101100011001111001010100001101011000011010010100111100010000110111100111001111100011001111000100110011
1101100001101000100111100110100111110010000110111110011110010100111100011000011010110011111001110010100010011110011001001
1110110000110111001001111001101001111001111110001100001101011001111011000011010001001111001011100111001100100100111110010
0111101100111111111000110011100110101001111101100011000011010001001111110010011110011010000110100110111000110011110010011
1100110100111100100100111111001000011010010101000011010011011000110011110011010111001111001001000011010001001111001001001
1110100111001110010000110100101010000011011000110010011100001011011001111001101001111101001111011000011010110011111011
1000110000110100111011100001101001010100001101110010000110100111110001100111100111011001111011001111001111001111100100
1000011011100100111110100111001101100011001111001111100111111011001111001010100111101100111111110011111011100011000011
0100010011110011100100111111001001111001101001111001110111000011010011111100011001111001010011110010101000011010110011
1001101001111001111001111101000011011110011110010011000110011110001001111001001000111111110000110111100111100100110001
1000011010010010011111111001111001101001111001011110000110100111110000110100101111100011000011010001001111101001111001
1001111001010100111011100001111110001100111100110100111100010011111101100111111100001101001111001111010011110011010000
0011000011010001000011011111001111001011110011110110011111100100001101101100111111010011100011001111000100001101000100
1011111000110011111010011110111000110000110111001000011010110011110011010000110111001000011010010100111101100001101011
0000110111111000011111100011001111010011101110001100001101001010100111111100111111011000011010011101010000110101010101
1000011111100011001111010011101110001100001101101001111001110010011111101100111100101111001111011001111001110101001111
1101110000111111000110011111111001111001101000011011101100111011001111001011011001111100100111001011000110011111001111
1011100011001111001011110000110101100111100110100001101001001001111001101001111001110101001111110110000110100100110000
0
K = 1.3
time: 1.060000

Process finished with exit code 0
```

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
8
1100101100111010110000000111010110000001101110001000110111101101011100010001000000011101011101010011000101111011111011010
1111011111011011001111000111010100110001011110101001110000001100001010110001100101000000011101011101010101111010111011101
0010100011101110010100111010111010101011110110111000110111000000111000111011111011011001010011110101101111010101011101
1100101001001010010010111101111001011011110011101111011001111000111010101100111100100111011011001010011110100010111000000
0011001111011011110101111011011000111111110100111101011111010001011110100101100111101010110011100010001101001111001111101
0000110001011100000001100111011110100101000000110001110111001001100101001111010001011110010011100011000100011001111011111
0100101001111010001011100000011100011000100011010011110001100000001100111110100010111000000111000110001000110010001110100
1110111101100011101010101111001111011011100010001101111011001110000111000111011111001001110011111010111011111010001110110
0011100011011110010001110101100010001110101110110110010011100011101001011110111001011011001111011011100011000000011101
0111011100011100101100011101100000001000111111101111011111010001111010001101011000000110110111001110111011000111011101000101011
1101001011000111010101101011011100010001101001111011110110011101111010001111110111001111010010100000011100111110100
0110110110010100111101011011110010011110101110101010010010011110101101110001100101101010110011110010001000111111100001010
1101110001000110110111101011101001011000101100010111100111100100111010111101010100011111111011011100111011100010001001010
0011100100011000111011111010001110000110101111101011000100010000101011101110110111000110111000000010000001110011100000001
0001110111000110000001100101001011000101111011011100011011100011101110101011010111000111001001000111010110001011111011011
1000110111000111001111101001111010111100100011000111011001111010110110001011110101011001110001110000110001110010100111101
0111011100110101111010111011110010001000001110011100011001011001111011001111010111000110010001011
K = 2.1
time: 1.370000

Process finished with exit code 0
```

По результатам видно, что алгоритм Хаффмана справился с задачей сжатия лучше, таже скорость больше у алгоритма Хаффмана, но алгоритм Шеннона-Фано проще в реализации.

Задание 6.

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

Код, кодирование Шеннона-Фано:

```c
#include <stdio.h>
#include <stdlib.h>


typedef struct node{
    int count;
    char letter[15];
    char code[30];
}node;
void sort(int n, node *array[n]){
    for(int i = 0; i < n; i++){
        int is_sort = 1;
        for(int j = n - 1; j > i; j--){
            if(array[j]->count > array[j-1]->count){
                node *temp = array[j];
                array[j] = array[j-1];
                array[j-1] = temp;
                is_sort = 0;
            }
        }
        if(is_sort){
            break;
        }
    }
}
void putChar(char *arr, char c){
    char *begin = arr;
    while (*begin!='\0'){
        begin++;
    }
    *begin = c;
    *(++begin) = '\0';
}
void fano(int size, node *array[size]){
    int sum1 = 0, sum2 = 0;
    node *part1[size];
    node *part2[size];
    int size1 = 0;
    int size2 = 0;
    for(int i = 0; i < size; i++){
        if(sum1 < sum2){
            part1[size1++] = array[i];
            sum1+=array[i]->count;
            putChar(array[i]->code, '1');
        }
        else{
            part2[size2++] = array[i];
            sum2+=array[i]->count;
            putChar(array[i]->code, '0');
        }
    }
    if(size1 > 1){
        fano(size1, part1);
    }
    if(size2 > 1){
        fano(size2, part2);
```

```
    }
}



int main(){
    FILE *f = fopen("text.txt", "r");
    int size;
    scanf("%d", &size);
    char letters[size+1];
    char D[4120][size+1];
    int count = 0;
    while (fread(letters, 1, size, f) == size) { // Считываем по 2 символа
        letters[size] = '\0';
        for(int i = 0; i < size+1; i++){
            D[count][i] = letters[i];
        }
        count++;
    }
    fclose(f);
    char *begin;
    node *data[count];
    int size_data = 0;
    for(int i = 0; i < count; i++){
        int flag = 0;
        for(int j = 0; j < size_data; j++){
            int flag_isEq = 1;
            begin = D[i];
            char *b = data[j]->letter;
            while (*begin != '\0'){
                if(*begin != *b){
                    flag_isEq = 0;
                    break;
                }
                begin++;
                b++;
            }
            if(flag_isEq){
                flag = 1;
                data[j]->count++;
                break;
            }
        }
        if(flag == 0){
            data[size_data] = (node *)malloc(sizeof(node));
            char *b = data[size_data]->letter;
            begin = D[i];
            while (*begin!='\0'){
                *b = *begin;
                b++;
                begin++;
            }
            *b = '\0';
            data[size_data]->count = 1;
            data[size_data]->code[0] = '\0';
            size_data++;
        }
    }
    sort(size_data, data);
    fano(size_data, data);

    char answer[count * size * 10];
    int size_ans = 0;
    char *ansBg = answer;
```

```
    for(int i = 0; i < count; i++){
        for(int j = 0; j < size_data; j++){
            int flag = 1;
            char *b = data[j]->letter;
            begin = D[i];
            while (*begin != '\0'){
                if(*begin != *b){
                    flag = 0;
                }
                begin++;
                b++;
            }
            if(flag){
                begin = data[j]->code;
                while (*begin!='\0'){
                    *ansBg = *begin;
                    ansBg++;
                    size_ans++;
                    begin++;
                }
            }
        }
    }
    *ansBg = '\0';

    char string[4121];
    begin = string;
    ansBg = answer;
    int shift = 0;
    while (*ansBg!='\0'){
        int flag = 0;
        for(int i = 0; i < size_data; i++){
            int flag_isEq = 1;
            for(int j = 0; j < shift + 1; j++){
                if(ansBg[j] != data[i]->code[j]){
                    flag_isEq = 0;
                    break;
                }
                if(j == shift && data[i]->code[j+1] != '\0'){
                    flag_isEq = 0;
                }
            }
            if(flag_isEq){
                ansBg+=shift+1;
                flag = 1;
                char *b = data[i]->letter;
                while (*b != '\0'){
                    *begin = *b;
                    b++;
                    begin++;
                }
            }
        }
        if(flag){
            shift = 0;
        }
        else{
            shift++;
        }
    }
    *begin = '\0';
    printf("%s\n",string);
    for(int i = 0; i < size_data; i++){
        free(data[i]);
```

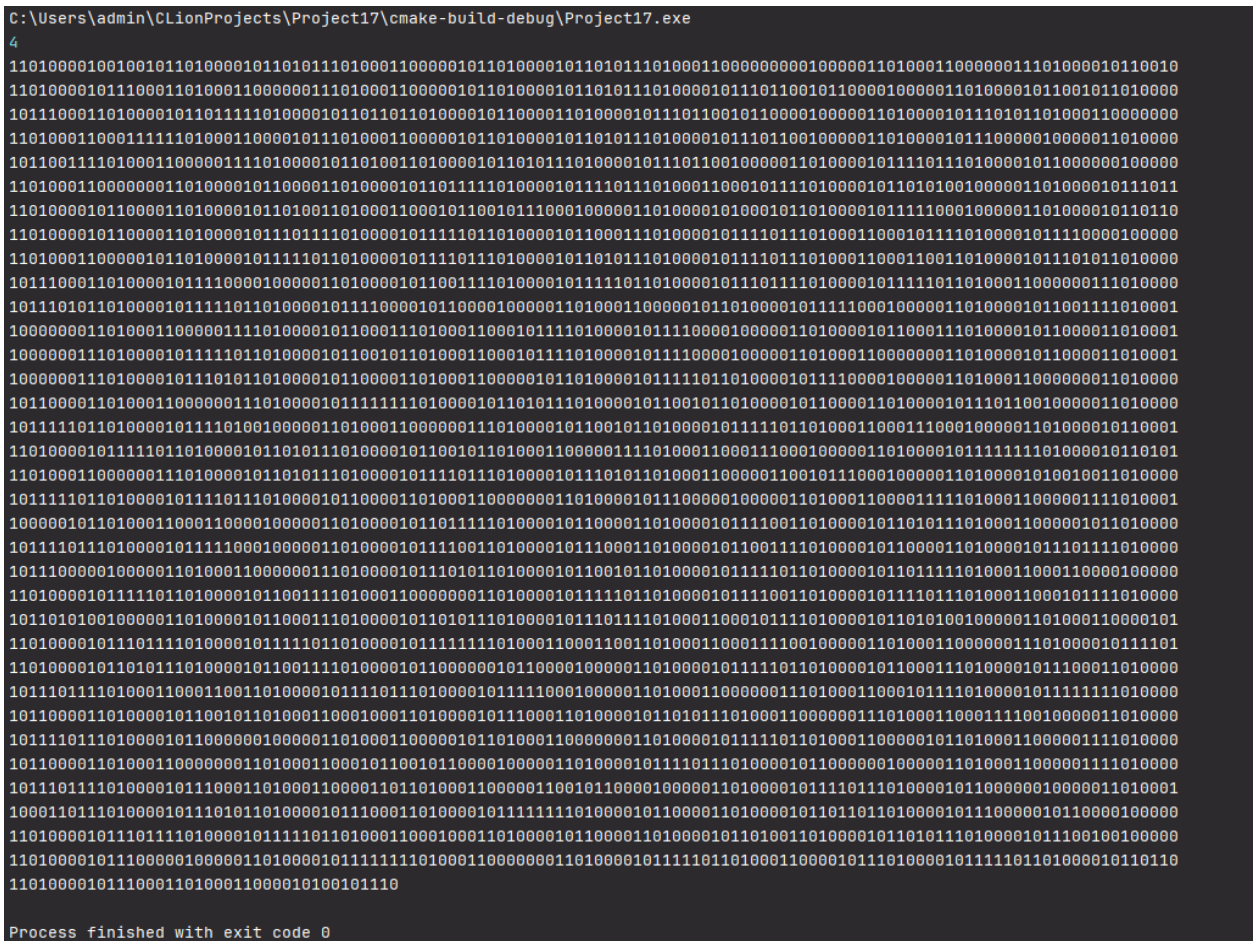
```
    }
    return 0;
}
```

Выводы программы:

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
2
11010000100100101101000010110101110100011000001011010000101101011101000110000000001000001101000110000001110100001011001
11010000101110001101000110000001110100011000001011010000101101011101000010111011001011000010000011010000101100101100000
10111000110100001011011111010000101101101100100001011000011010000101110110010110000100000110100001011101011010001100000000
11010001100011111101000110000101110100011000001011010000101101011101000010111011001000000110100001011100000100000110100000
10110011110100011000001111010000101101001101000010110101110100001011101110100001011000000010000011010000101110000010000000
11010001100000001101000010110000110100001011011111010000101111011101000110001011110000100001101010010000011010000101110011
11010000101100001101000010110100110100011000101100101110001000001101000010100010110100001011111000100000110100001011011011
11010000101100001101000010111011101000010111110110100001011000111010000101111011101000110001011110100001011110000100000100000
11010001100000101101000010111110110100001011110111010000101101011101000010111011101000110001100110100001011101011010000
10111000110100001011110000100000110100001011001111010000101111101101000010111011101000010111110101000011000000111010000
10111010110100001011111011010000101111000010110000100000110100011000001011010000101111100100000110100001011001111010001
10000000110100011000001111010000101100011101000110001011101000010111100001000011010000101100011101000010110000110100001
10000001110100001011111011010000101100101101000110001011101000010111100001000011010001100000001101000010110000110100001
10000001110100001011101011010000101100001101000110000010110100001011111011010000101111000100000110100011000000011010000
10110000110100011000000111010000101111111101000010110101110100001011001011010000101100001101000010111011001000001101000
10111110110100001011110100100000110100011000000111010000101100101101000011010000101111011010001100011100010000011010000101100010
11010000101111101101000010110101110100001011001011010001100000111101000110001110001000001101000010111111101000010110101
11010001100000011101000010110101110100001011110111010000101110101101000110000011001011100010000011010000101001001100000
10111110110100001011110111010000101100001101000110000001101000010111000001000011010001100001111010001100000111010001
10000010110100011000110000100000110100001011011111010000101100001101000010111001101000010110101110100011000001011010000
10111101110100001011111000100000110100001011110011010000101110001101000010110011110100001011000011010000101110111010000
10111000001000001101000110000001110100001011101011010000101100101101000010111011101000010110111110100011000110000100000
11010000101111101101000010110011110100011000000011010000101111101101000010111001101000010111101110100011000101111010000
10110101001000001101000010110001110100001011010111010000101110111101000110001011110100001011010100100000110100011000010
11010000101110111101000010111110110100001011110111111010001100011101000010111011101000110000101110100011000001110100001
11010000101110101101000010110011110100001011000000101100001000001101000010111110110100001011000111010000101110001101000
10111011110100011000110011010000101111011101000010111110001000001101000110000011101000110001011110100001011111111010000
10110000110100001011001011010001100010001101000010111000110100001011010111010001100000011101000110001111001000001101000
10111101110100001011000000100000110100011000001011010001100000001101000010111110110100011000001011010000101111010000100
10111011110100001011100011010001100001101101000110000011010000101111011101000110000101110100001011000001000001101000110
10001101110100001011101011010000101110001101000010111111110100001011000011010000101101101101000010111000001011000010000
11010000101110111101000010111110110100011000100011010000101100001101000010110100110100001011010111010000101110010010000
11010000101110000010000011010000101111111101000110000000110100001011110110100011000010111010000101111101101000010110110
11010000101110001101000110000101001011110
```

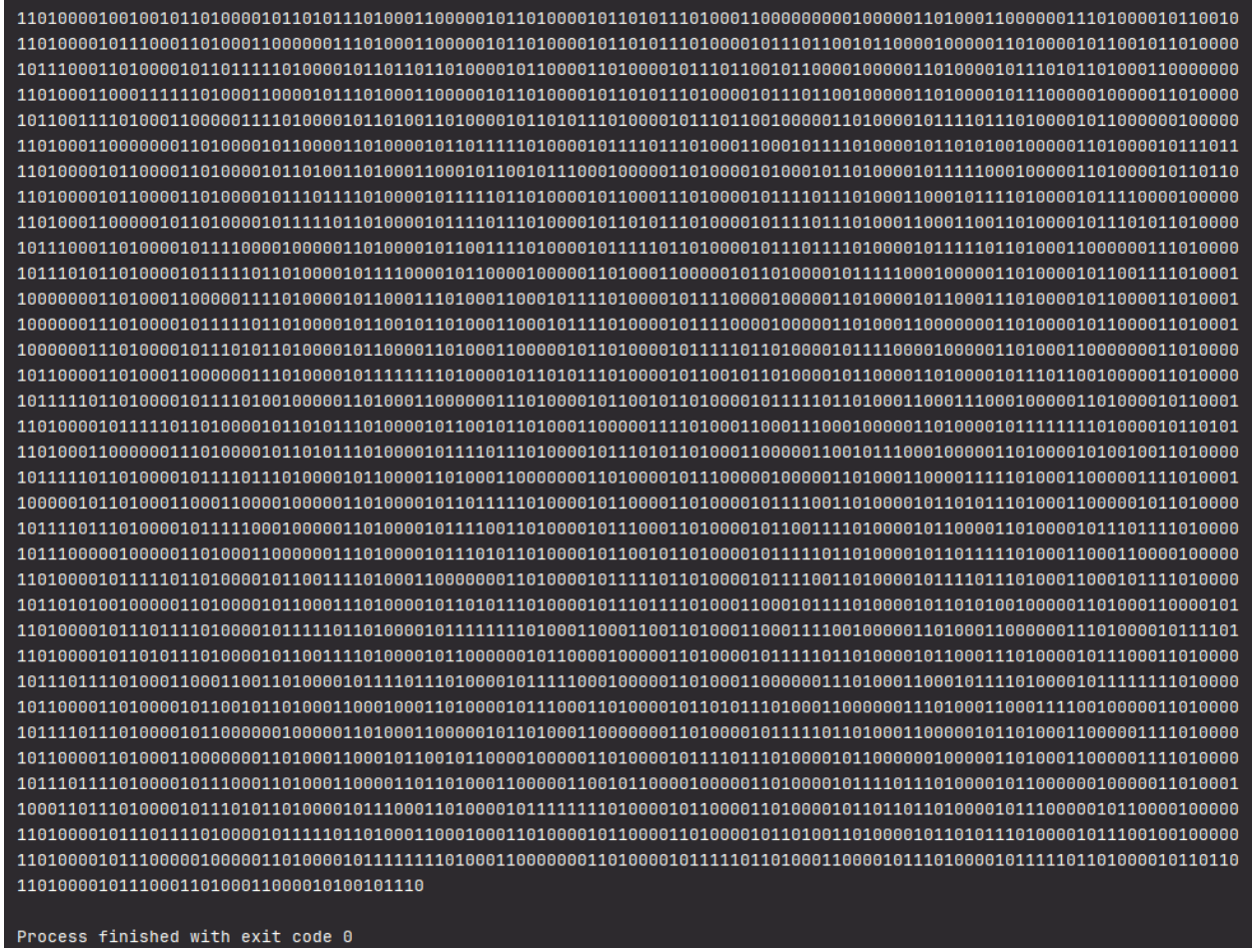
Process finished with exit code 0

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
4
```



```
Process finished with exit code 0
```

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
8
```



```
Process finished with exit code 0
```

Код, кодирование Хаффмана:

```
#include <stdio.h>
#include <stdlib.h>

typedef struct node{
    int count;
    char letter[15];
    char code[30];
    struct node* left;
    struct node* right;
}node;
void sort(int n, node **array){
    for(int i = 0; i < n; i++){
        int is_sort = 1;
        for(int j = n - 1; j > i; j--){
            if(array[j]->count > array[j-1]->count){
                node *temp = array[j];
                array[j] = array[j-1];
                array[j-1] = temp;
                is_sort = 0;
            }
        }
        if(is_sort){
            break;
        }
    }
}
void haffman(node *let, char* code, char *endCode){
    if(let->letter[0] == '\0'){
        *endCode = '0';
        *(endCode+1) = '\0';
        haffman(let->left, code, endCode+1);
        *endCode = '1';
        *(endCode+1) = '\0';
        haffman(let->right, code, endCode+1);
    }
    else{
        char *begin = let->code;
        char *begin_code = code;
        while (*begin_code!='\0'){
            *begin = *begin_code;
            begin++;
            begin_code++;
        }
        *begin = '\0';
    }
}
void huffmanTreeToArray(node **data, int* count, node *root){
    if(root->letter[0] != '\0'){
        data[*count] = root;
        (*count)+=1;
    }
    else{
        huffmanTreeToArray(data, count, root->left);
        huffmanTreeToArray(data, count, root->right);
    }
}
void free_data(node *root){
    if(root->letter[0] == '\0'){
        free_data(root->left);
        free_data(root->right);
    }
    free(root);
```

```
}

int main(){
    FILE *f = fopen("text.txt", "r");
    int size;
    scanf("%d", &size);
    char letters[size+1];
    char D[4120][size+1];
    int count = 0;
    while (fread(letters, 1, size, f) == size) { // Считываем по 2 символа
        letters[size] = '\0';
        for(int i = 0; i < size+1; i++){
            D[count][i] = letters[i];
        }
        count++;
    }
    fclose(f);
    char *begin;
    node *data[count];
    int size_data = 0;
    for(int i = 0; i < count; i++){
        int flag = 0;
        for(int j = 0; j < size_data; j++){
            int flag_isEq = 1;
            begin = D[i];
            char *b = data[j]->letter;
            while (*begin != '\0'){
                if(*begin != *b){
                    flag_isEq = 0;
                    break;
                }
                begin++;
                b++;
            }
            if(flag_isEq){
                flag = 1;
                data[j]->count++;
                break;
            }
        }
        if(flag == 0){
            data[size_data] = (node *)malloc(sizeof(node));
            char *b = data[size_data]->letter;
            begin = D[i];
            while (*begin!='\0'){
                *b = *begin;
                b++;
                begin++;
            }
            *b = '\0';
            data[size_data]->count = 1;
            data[size_data]->code[0] = '\0';
            size_data++;
        }
    }
    sort(size_data, data);
    while (size_data != 1){
        node *temp = malloc(sizeof(node));
        temp->count = data[size_data-1]->count + data[size_data-2]->count;
        temp->letter[0] = '\0';
        temp->code[0] = '\0';
        temp->left = data[size_data-1];
        temp->right = data[size_data-2];
        data[size_data-2] = temp;
```

```
        size_data--;
        sort(size_data, data);
    }
    node *root = data[0];
    char code[300];
    haffman(root, code, code);
    size_data = 0;
    huffmanTreeToArray(data, &size_data, root);
    char answer[count * size * 10];
    int size_ans = 0;
    char *ansBg = answer;
    for(int i = 0; i < count; i++){
        for(int j = 0; j < size_data; j++){
            int flag = 1;
            char *b = data[j]->letter;
            begin = D[i];
            while (*begin != '\0'){
                if(*begin != *b){
                    flag = 0;
                }
                begin++;
                b++;
            }
            if(flag){
                begin = data[j]->code;
                while (*begin!='\0'){
                    *ansBg = *begin;
                    ansBg++;
                    size_ans++;
                    begin++;
                }
            }
        }
    }

    *ansBg = '\0';
    char string[4121];
    begin = string;
    ansBg = answer;
    int shift = 0;
    while (*ansBg!='\0'){
        int flag = 0;
        for(int i = 0; i < size_data; i++){
            int flag_isEq = 1;
            for(int j = 0; j < shift + 1; j++){
                if(ansBg[j] != data[i]->code[j]){
                    flag_isEq = 0;
                    break;
                }
                if(j == shift && data[i]->code[j+1] != '\0'){
                    flag_isEq = 0;
                }
            }
            if(flag_isEq){
                ansBg+=shift+1;
                flag = 1;
                char *b = data[i]->letter;
                while (*b != '\0'){
                    *begin = *b;
                    b++;
                    begin++;
                }
            }
        }
```

```
        if(flag){
            shift = 0;
        }
        else{
            shift++;
        }
    }
    *begin = '\0';
    printf("%s\n",string);


    free_data(root);

    return 0;
}
```

Выводы программы:

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
2
11010000100100101101000010110101110100011000001011010000101101011101000110000000001000001101000110000001110100001011001
11010000101110001101000110000001110100011000001011010000101101011101000010111011001011000010000011010000101100101101000
10111000110100001011011111010000101101101101000010110000110100001011101100101100001000001101000010111010110100011000000
11010001100011111101000110000101110100011000001011010000101101011101000010111011001000001101000010111000010000011010000
11010011110100011000001111010000101101001101000010110101110100001011101100100000110100001011110111010000101100000100000
11010001100000001101000010110000110100001011011111010000101111011101000110001011101000010111010100100000110100001011101
11010000101100001101000010110100110100011000101100101110001000000110100001011101110100001011101010010000011010000101110
11010000101100001101000010111011110100001011111011010000101100011101000010111011101000110001011110100001011110000100000
11010001100000101101000010111110110100001011110111010000101101011101000010111011101000110001100110101000010111010110000
10111000110100001011110000100000110100001011001111010000101111011101000010111011101000110001011110100001011110000111010
11010000101100001101000010111110110100001011110111010000101101011101000010111011101000110001100110101000010111010110000
10111010110100001011111011010000101111000010110000100000110100011000001011010000101111100010000011010000101100111101001
10000000110100011000001111010000101100011101000110001011110100001011110000100000110100001011000111010000101100001101001
10000001110100001011111011010000101100101101000110001011110100001011110000100000110100011000000011010000101100011010000
10000001110100001011101011010000101100001101000110000010110100001011111011010000101111000100000110100011000000011010000
10110000110100011000000111010000101111111101000010110101110100001011001011010000101100011010000101110110010000011010000
10111101101000010111101001000001101000110000001110100001011001011010000101111000100000110100001011001110100000111010001
11010000101111101101000010110101110100001011001011010001100000111010001100011100010000011010000101111111010000101110000
10111010100000011010000101100011110100001011010111010000101100011010000101111011010000101110001110100001011101111010001
11010000101110111010000101111101101000010111111111010001100011001101000110001111001000001101000110000001110100001011101
11010000101101011101000010110011110100001011000000101100001000001101000010111110110100001011000111010000101110001101000
10111011110100011000110011010000101111011101000010111110001000001101000011000001110100011000101111010000101111111010000
10110000110100001011001011010001100010001101000010111000110100001011010111010001100000011101000110001111001000001101000
10111101110100001011000000100000110100011000001011010001100000011010000101111011010001100000101101000110000001111010000
10110000110100011000000011010001100010110010110000100000110100001011101110100001011000001000001101000110000001111010000
10111011110100001011100011010001100001101101000110000011001011000010000011010000101111011101000010110000001000001101001
10001101110100001011101011010000101110001101000010111111110100001011000011010000101101101010000101110000010110000100000
11010000101110111101000010111110110100011000100010110000101100001101000010111010011010000101101011101000010111001001000
11010000101110000010000011010000101111111101000110000000110100001011111011010001100001011101000101111011010000010000000
11010000101110001101000110000101001011110
```

Process finished with exit code 0

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
4
1101000010010010110100001011010111101000110000001011010000101101011110100011000000000100000110100011000000111010000101100 10
1101000010111000110100011000000111010001100000101101000010110101111010000101110110010110000100000110100001011001011010000
10111000110100001011011111010000101101101101000010110000110100001011101100101100001000001101000010111010110100011000000 0
110100011000111111010001100001011101000110000010110100001011010111101000010111011001000001101000010111000000100000110 10000
10110011110100011000001111010000101101001101000010110101111010000101110110010000011010000101111011110100001011000000 0100000
11010001100000001101000010110000110100001011011111010000101111011101000110001011110100001011010100100000110100001011101 1
11010000101100001101000010110100110100011000101100101110001000001101000010100010110100001011111000100000110100001011 0110
11010000101100001101000010111011110100001011111011010000101100011101000010111101110100011000101111010000101111000010 0000
11010001100000101101000010111110110100001011110111010000101101011101000010111101110100011000110011010000101110101101 0000
10111000110100001011110000100000110100001011001111010000101111101101000010111011110100001011111011010001100000011101 0000
101110101101000010111110110100001011110000101100001000001101000110000010110100001011111000100000110100001011001111010001
100000001101000110000011110100001011000111010001100010111101000010111100001000001101000010110001110100001011000011010001
100000011101000010111110110100001011001011010001100010111101000010111100001000001101000110000000110100001011000011010001
100000011101000010111010110100001011000011010001100000101101000010111110110100001011110000100000110100011000000011010000
101100001101000110000001110100001011111111010000101101011101000010110010110100001011000011010000101110110010000011010000
101111101101000010111101001000001101000110000001110100001011001011010000101111101101000110001110001000001101000010110001
110100001011111011010000101101011101000010110010110100011000001111010001100011100010000011010000101111111101000010110101
110100011000000111010000101101011101000010111101110100001011101011010001100000110010111000100000110100001010010011010000
101111101101000010111101110100001011000011010001100000001101000010111000000100000110100011000011111010001100000111101000 1
100000101101000110001100001000001101000010110111110100001011000011010000101111001101000010110101110100011000001011010000
101111011101000010111110001000001101000010111100110100001011100011010000101100111101000010110000110100001011101111010000
101110000001000001101000110000001110100001011101011010000101100101101000010111110110100001011011111010001100011000010000 0
110100001011111011010000101100111101000110000000110100001011111011010000101111001101000010111101110100011000101111010000
101101010010000011010000101100011101000010110101110100001011101111010001100010111101000010110101001000001101000110000101
110100001011101111010000101111101101000010111111110100011000110011010001100011110010000011010001100000011101000010111101
110100001011010111010000101100111101000010110000001011000010000011010000101111101101000010110001110100001011100011010000
101110111101000110001100110100001011110111010000101111100010000011010001100000011101000110001011110100001011111111010000
101100001101000010110010110100011000100011010000101110001101000010110101110100011000000111010001100011110010000011010000
101111011101000010110000000100000110100011000001011010001100000001101000010111110110100011000001011010001100000111101000 0
101100001101000110000000110100011000101100101100001000001101000010111101110100001011000000010000011010001100000111101000 0
101110111101000010111000110100011000011011010001100000110010110000100000110100001011110111010000101100000001000001101000 1
100011011101000010111010110100001011100011010000101111111101000010110000110100001011011011010000101110000001011000010000 0
110100001011101111010000101111101101000110001000110100001011000011010000101101001101000010110101110100001011100100100000
110100001011100000010000011010000101111111101000110000000110100001011111011010001100001011101000010111110110100001011 0110
11010000101110001101000110000101001011 10

Process finished with exit code 0
```

```
C:\Users\admin\CLionProjects\Project17\cmake-build-debug\Project17.exe
8
1101000010010010110100001011010111101000110000001011010000101101011110100011000000000100000110100011000000111010000101100 10
1101000010111000110100011000000111010001100000101101000010110101111010000101110110010110000100000110100001011001011010000
10111000110100001011011111010000101101101101000010110000110100001011101100101100001000001101000010111010110100011000000 0
110100011000111111010001100001011101000110000010110100001011010111101000010111011001000001101000010111000000100000110 10000
10110011110100011000001111010000101101001101000010110101111010000101110110010000011010000101111011110100001011000000 0100000
11010001100000001101000010110000110100001011011111010000101111011101000110001011110100001011010100100000110100001011101 1
11010000101100001101000010110100110100011000101100101110001000001101000010100010110100001011111000100000110100001011 0110
11010000101100001101000010111011110100001011111011010000101100011101000010111101110100011000101111010000101111000010 0000
11010001100000101101000010111110110100001011110111010000101101011101000010111101110100011000110011010000101110101101 0000
10111000110100001011110000100000110100001011001111010000101111101101000010111011110100001011111011010001100000011101 0000
101110101101000010111110110100001011110000101100001000001101000110000010110100001011111000100000110100001011001111010001
100000001101000110000011110100001011000111010001100010111101000010111100001000001101000010110001110100001011000011010001
100000011101000010111110110100001011001011010001100010111101000010111100001000001101000110000000110100001011000011010001
100000011101000010111010110100001011000011010001100000101101000010111110110100001011110000100000110100011000000011010000
101100001101000110000001110100001011111111010000101101011101000010110010110100001011000011010000101110110010000011010000
101111101101000010111101001000001101000110000001110100001011001011010000101111101101000110001110001000001101000010110001
110100001011111011010000101101011101000010110010110100011000001111010001100011100010000011010000101111111101000010110101
110100011000000111010000101101011101000010111101110100001011101011010001100000110010111000100000110100001010010011010000
101111101101000010111101110100001011000011010001100000001101000010111000000100000110100011000011111010001100000111101000 1
100000101101000110001100001000001101000010110111110100001011000011010000101111001101000010110101110100011000001011010000
101111011101000010111110001000001101000010111100110100001011100011010000101100111101000010110000110100001011101111010000
101110000001000001101000110000001110100001011101011010000101100101101000010111110110100001011011111010001100011000010000 0
110100001011111011010000101100111101000110000000110100001011111011010000101111001101000010111101110100011000101111010000
101101010010000011010000101100011101000010110101110100001011101111010001100010111101000010110101001000001101000110000101
110100001011101111010000101111101101000010111111110100011000110011010001100011110010000011010001100000011101000010111101
110100001011010111010000101100111101000010110000001011000010000011010000101111101101000010110001110100001011100011010000
101110111101000110001100110100001011110111010000101111100010000011010001100000011101000110001011110100001011111111010000
101100001101000010110010110100011000100011010000101110001101000010110101110100011000000111010001100011110010000011010000
101111011101000010110000000100000110100011000001011010001100000001101000010111110110100011000001011010001100000111101000 0
101100001101000110000000110100011000101100101100001000001101000010111101110100001011000000010000011010001100000111101000 0
101110111101000010111000110100011000011011010001100000110010110000100000110100001011110111010000101100000001000001101000 1
100011011101000010111010110100001011100011010000101111111101000010110000110100001011011011010000101110000001011000010000 0
110100001011101111010000101111101101000110001000110100001011000011010000101101001101000010110101110100001011100100100000
110100001011100000010000011010000101111111101000110000000110100001011111011010001100001011101000010111110110100001011 0110
11010000101110001101000110000101001011 10

Process finished with exit code 0
```

Задание 7.

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

Полученное сообщение:

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.

Вывод: в ходе проделанной работы я научился кодировать и декодировать двоичную последовательность.