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 №4**

по дисциплине: Теория информации

тема: «Исследование кода Гилберта-Мура»

Выполнил: ст. группы ПВ-233

Ситников Алексей Павлович

Проверил: Твердохлеб Виталий

Викторович

Белгород 2025 г.

**Задание 1–2.**

Изучить принцип построения кода, используя пример в закрепленном файле «Пример. ЛБ4.pdf».

Построить обработчик, выполняющий компрессию по алгоритму Гилберта-Мура.

**Код:**

#include <stdio.h>  
#include <math.h>  
#include <windows.h>  
  
int main() {  
 SetConsoleOutputCP(**CP\_UTF8**);  
  
 printf("Введите мощность алфавита\n");  
 int n;  
 scanf("%d", &n);  
 char A[n];  
 double p[n];  
 double d[n];  
 double q[n];  
 int log[n];  
 char code[n][30];  
 getchar();  
 for(int i = 0; i < n; i++){  
 printf("Введите символ и вероятность\n");  
 scanf("%c %lf", &A[i], &p[i]);  
 getchar();  
 }  
 d[0] = 0;  
 q[0] = p[0]/2;  
 for(int i = 0; i < n; i++){  
 if(i != 0){  
 double sum = 0;  
 for(int j = 0; j < i; j++){  
 sum+=p[j];  
 }  
 d[i] = sum;  
 q[i] = d[i] + p[i]/2;  
 }  
 log[i] = (int)(fabs(log2(p[i])) + 2);  
 double num = q[i];  
  
 char temp[log[i]+1];  
 char \*begin = temp;  
 int maxIterations = log[i];  
 while (num > 0 && maxIterations > 0) {  
 num \*= 2;  
 if (num >= 1) {  
 \*begin = '1';  
 num -= 1;  
 } else {  
 \*begin = '0';  
 }  
 maxIterations--;  
 begin++;  
 }  
 \*begin = '\0';  
 char \*b = code[i];  
 begin = temp;  
 while (\*begin!='\0'){  
 \*b = \*begin;  
 b++;  
 begin++;  
 }  
 while (num <= 0 && maxIterations > 0){  
 \*b = '0';  
 b++;  
 maxIterations--;  
 }  
 \*b = '\0';  
 }  
  
 printf("Введите строку для кодирования\n");  
 char buf[10000];  
 scanf("%s", buf);  
 char \*b = buf;  
 char ans[1000000];  
 char \*begin = ans;  
 while (\*b!='\0'){  
 for(int i = 0; i < n; i++){  
 if(\*b == A[i]){  
 char \*b\_code = code[i];  
 while (\*b\_code!='\0'){  
 \*begin = \*b\_code;  
 b\_code++;  
 begin++;  
 }  
 }  
 }  
 b++;  
 }  
 \*begin = '\0';  
  
 for(int i = 0; i < n; i++){  
 printf("%c %s\n", A[i], code[i]);  
 }  
  
 printf("%s", ans);  
 return 0;  
}

Пример программы:  
Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

**Задание 3.**

Создать генераторы данных, работающих как источники Хартли и Бернулли (в двоичном алфавите).

Хартли:

#include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
#include <windows.h>  
  
int main() {  
 SetConsoleOutputCP(**CP\_UTF8**);  
 int num\_bits;  
 printf("Введите количество символов\n");  
 scanf("%d", &num\_bits);  
 char ans[num\_bits + 1];  
 char \*begin = ans;  
 srand(time(**NULL**));  
 for (int i = 0; i < num\_bits; i++) {  
 \*begin = rand() % 2 == 1 ? '1' : '0';  
 begin++;  
 }  
 \*begin = '\0';  
 printf("%s", ans);  
 return 0;  
}

Пример вывода:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Бернулли:

#include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
#include <windows.h>  
  
  
int main() {  
 SetConsoleOutputCP(**CP\_UTF8**);  
 int num\_bits;  
 double p;  
 printf("Введите количество символов и вероятность символа \"1\"\n");  
 scanf("%d %lf", &num\_bits, &p);  
 srand(time(**NULL**));  
  
 char ans[num\_bits+1];  
 char \*begin = ans;  
  
 for (int i = 0; i < num\_bits; i++) {  
 double r = (double)rand() / **RAND\_MAX**; // Случайное число от 0 до 1  
 \*begin = (r < p) ? '1' : '0'; // Генерация бита с вероятностью p  
 begin++;  
 }  
 \*begin = '\0';  
 printf("%s", ans);  
 return 0;  
}

Пример вывода:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

**Задание 4.**

Сгенерировать 2 цепочки данных длиной 800 символов каждую (соответственно, порожденные по Бернуллевскому принципу и принципу Хартли).

Хартли:

10010011111010100001000011001110011110101011000101011010110011111011100000001011000011101010110010100101100101000000000001111001001010011000101000011100110100100011000011100111110110101111001101110110100100101000011010101101101010001010000101010010000000011011100010000101100100010011101100001011111000110000011110110100011011100001101111110000100000001011100101111001100001001111110100000000111101010001101100101101000101111110101110100011010010011001101010000011111110011101011111100100000011011011001110000100000100011100101011001000111111101010100011001110110111110000001010001011101110111001011000001010011100010101101101100111101100100001001010111010100000101110110011101101101110001011010100101010010011000000001100011111110101011011110001101110100101011100000101010111001101010100001101000100

Бернулли, вероятность «1» 0,7:

10111101001010001111010101110011010101100111111001111100100001000110110111011100010101111010110111111110111101101101011001010100101110101111111010111101101010011111111111111111111010010111111110111111100100001110011111011001011011011100101011111111111011111110011110110111101101110111101110101100011110111111111001100111111111111100100001110111101101111111011101111111111111111101011111101110101010110111011010011111111000011101110101100110011011111111110111110001001001110101111111010111110011111110111101111110101111001111111111001110111000111111110000111011010111011110001111101111110011101110011100111101111010111100100110101010111111100001110110011101110111101101101000111011011001110011101110011100011110011011110110111101111101111101010011111010111001111111110111101111011110011111111111101100

**Задание 5:**

Выбрать произвольную текстовую строку на русском языке длиной 100 символов, преобразовать в двоичный формат Unicode/ASCII (на выбор), пользуясь любым соответствующим онлайн-сервисом (на выходе имеем последовательность из 800 символов).

Солнце светит ярко и ветер шелестит листвой. Вдали слышен смех детей. Жизнь прекрасна и удивительна!

Представление в двоичном коде в соответствие с ASCII:  
11010000101000011101000010111110110100001011101111010000101111011101000110000110110100001011010100100000110100011000000111010000101100101101000010110101110100011000001011010000101110001101000110000010001000001101000110001111110100011000000011010000101110101101000010111110001000001101000010111000001000001101000010110010110100001011010111010001100000101101000010110101110100011000000000100000110100011000100011010000101101011101000010111011110100001011010111010001100000011101000110000010110100001011100011010001100000100010000011010000101110111101000010111000110100011000000111010001100000101101000010110010110100001011111011010000101110010010111000100000110100001001001011010000101101001101000010110000110100001011101111010000101110000010000011010001100000011101000010111011110100011000101111010001100010001101000010110101110100001011110100100000110100011000000111010000101111001101000010110101110100011000010100100000110100001011010011010000101101011101000110000010110100001011010111010000101110010010111000100000110100001001011011010000101110001101000010110111110100001011110111010001100011000010000011010000101111111101000110000000110100001011010111010000101110101101000110000000110100001011000011010001100000011101000010111101110100001011000000100000110100001011100000100000110100011000001111010000101101001101000010111000110100001011001011010000101110001101000110000010110100001011010111010000101110111101000110001100110100001011110111010000101100000010000100000000

**Задание 6–7:**

Бинарная последовательность:

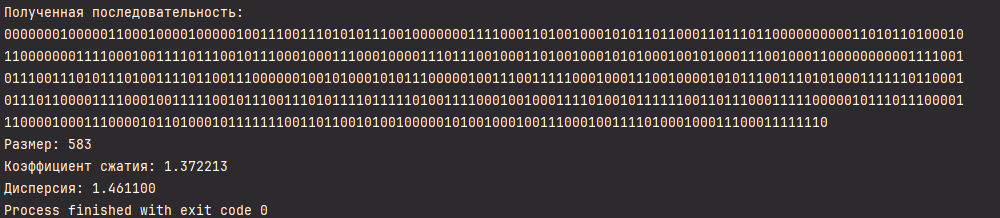
#include <stdio.h>  
#include <math.h>  
#include <windows.h>  
  
typedef struct node{  
 char letters[9];  
 char code[30];  
 double p;  
 double d;  
 double q;  
 int size;  
}node;  
int equalStr(char \*a, char \*b){  
 char \*ap = a;  
 char \*bp = b;  
 while (\*ap!='\0' && \*bp!='\0'){  
 if(\*ap!=\*bp){  
 return 0;  
 }  
 ap++;  
 bp++;  
 }  
 return \*ap == \*bp;  
}  
int main() {  
 SetConsoleOutputCP(**CP\_UTF8**);  
 char str[] = "11010000101000011101000010111110110100001011101111010000101111011101000110000110110100001011010100100000110100011000000111010000101100101101000010110101110100011000001011010000101110001101000110000010001000001101000110001111110100011000000011010000101110101101000010111110001000001101000010111000001000001101000010110010110100001011010111010001100000101101000010110101110100011000000000100000110100011000100011010000101101011101000010111011110100001011010111010001100000011101000110000010110100001011100011010001100000100010000011010000101110111101000010111000110100011000000111010001100000101101000010110010110100001011111011010000101110010010111000100000110100001001001011010000101101001101000010110000110100001011101111010000101110000010000011010001100000011101000010111011110100011000101111010001100010001101000010110101110100001011110100100000110100011000000111010000101111001101000010110101110100011000010100100000110100001011010011010000101101011101000110000010110100001011010111010000101110010010111000100000110100001001011011010000101110001101000010110111110100001011110111010001100011000010000011010000101111111101000110000000110100001011010111010000101110101101000110000000110100001011000011010001100000011101000010111101110100001011000000100000110100001011100000100000110100011000001111010000101101001101000010111000110100001011001011010000101110001101000110000010110100001011010111010000101110111101000110001100110100001011110111010000101100000010000100000000";  
 char \*begin = str;  
 char temp\_str[9];  
 char \*temp\_pointer;  
 node data[800];  
 int count = 0;  
 int count\_el = 0;  
 while (\*begin!='\0'){  
 temp\_pointer = temp\_str;  
 for(int i = 0; i < 8; i++){  
 \*temp\_pointer = \*begin;  
 temp\_pointer++;  
 begin++;  
 }  
 \*temp\_pointer = '\0';  
 int flag\_ = 0;  
 for(int i = 0; i < count; i++){  
 if(equalStr(data[i].letters, temp\_str)){  
 data[i].p++;  
 count\_el++;  
 flag\_ = 1;  
 }  
 }  
 if(flag\_==0){  
 temp\_pointer = temp\_str;  
 char \*b = data[count].letters;  
 data[count].p++;  
 while (\*temp\_pointer!='\0'){  
 \*b = \*temp\_pointer;  
 b++;  
 temp\_pointer++;  
 }  
 count\_el++;  
 count++;  
 }  
 }  
  
 for(int i = 0; i < count; i++){  
 data[i].p /= count\_el;  
 }  
 data[0].d = 0;  
 data[0].q = data[0].p / 2;  
 data[0].size = (int)(fabs(log2(data[0].p)) + 2);  
  
  
 for(int i = 0; i < count; i++){\  
 if(i!=0) {  
 data[i].d = data[i - 1].d + data[i - 1].p;  
 data[i].q = (data[i].p / 2) + data[i].d;  
 data[i].size = (int) (fabs(log2(data[i].p)) + 2);  
 }  
 double num = data[i].q;  
 begin = data[i].code;  
 int maxIterations = data[i].size;  
 while (num > 0 && maxIterations > 0) {  
 num \*= 2;  
 if (num >= 1) {  
 \*begin = '1';  
 num -= 1;  
 } else {  
 \*begin = '0';  
 }  
 maxIterations--;  
 begin++;  
 }  
 while (num <= 0 && maxIterations > 0){  
 \*begin = '0';  
 begin++;  
 maxIterations--;  
 }  
 \*begin = '\0';  
 }  
 for(int i = 0; i < count; i++){  
 printf("%s %s %d\n", data[i].letters, data[i].code, data[i].size);  
 }  
 begin = str;  
 int size\_ans = 0;  
 char ans[count\_el\*8 + 1];  
 char \*ansB = ans;  
 while (\*begin!='\0'){  
 char subStr[9];  
 for(int i = 0; i < 8; i++){  
 subStr[i] = \*begin;  
 begin++;  
 }  
 subStr[8] = '\0';  
 int flag = 0;  
 for(int i = 0; i < count; i++){  
 if(equalStr(data[i].letters, subStr)){  
 char \*b = data[i].code;  
 while (\*b!='\0'){  
 \*ansB = \*b;  
 ansB++;  
 b++;  
 size\_ans++;  
 }  
 flag = 1;  
 }  
 }  
 if(flag == 0) {  
 printf("Ошибка, символ не найден");  
 }  
 }  
 \*ansB = '\0';  
 printf("Полученная последовательность:\n%s\n", ans);  
 printf("Размер: %d\n", size\_ans);  
 printf("Коэффициент сжатия: %f\n", 8.\*count\_el/size\_ans);  
 double sr = 0;  
 for(int i = 0; i < count; i++){  
 sr+=data[i].p \* data[i].size;  
 }  
 double D = 0;  
 for(int i = 0; i < count; i++){  
 D+= data[i].p \* (data[i].size - sr) \* (data[i].size - sr);  
 }  
 printf("Дисперсия: %f", D);  
 return 0;  
}

Вывод программы:

Изображение выглядит как снимок экрана, текст

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как текст, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Последовательность символов:  
  


|  |
| --- |
| **Входные данные:**  11010000101000011101000010111110110100001011101111010000101111011101000110000110110100001011010100100000110100011000000111010000101100101101000010110101110100011000001011010000101110001101000110000010001000001101000110001111110100011000000011010000101110101101000010111110001000001101000010111000001000001101000010110010110100001011010111010001100000101101000010110101110100011000000000100000110100011000100011010000101101011101000010111011110100001011010111010001100000011101000110000010110100001011100011010001100000100010000011010000101110111101000010111000110100011000000111010001100000101101000010110010110100001011111011010000101110010010111000100000110100001001001011010000101101001101000010110000110100001011101111010000101110000010000011010001100000011101000010111011110100011000101111010001100010001101000010110101110100001011110100100000110100011000000111010000101111001101000010110101110100011000010100100000110100001011010011010000101101011101000110000010110100001011010111010000101110010010111000100000110100001001011011010000101110001101000010110111110100001011110111010001100011000010000011010000101111111101000110000000110100001011010111010000101110101101000110000000110100001011000011010001100000011101000010111101110100001011000000100000110100001011100000100000110100011000001111010000101101001101000010111000110100001011001011010000101110001101000110000010110100001011010111010000101110111101000110001100110100001011110111010000101100000010000100000000 |
| **Вывод:** Полученная последовательность:  001010011101001010100000101010100101011110111100010010001100101010001111011000011011100001100100111110000001110011011111  000010100011111010100001111101011001110110110010101000101000011100111010000110111000011001001111100000011001001111101011  101000111110111100011001000101010100110010011110110001111100000011100110111110000101000010101010011100110111101100011111  000000110111000010101000001111000011110010010100001111001100001111010000111101100010101010011100111010001111011000010101  010111111100010011111011110001100100010101111101000111101100001111100101001100100111111101000101000011110100001100100111  110000001100100011110000111100100101000011111010110011100110011111011010010101111011111111001101000011111101100111110101  100110010001110110110111110101100111101100111101100001010111100111101101010000111001110100011111111100100111101000011100  11001101110000111001101111100000011001000101010101111111100100101011110011110110111111011111111110  Размер: 938  Коэффициент сжатия: 1.569296  Дисперсия: 3.631734 |

|  |
| --- |
| **Входные данные:**  Солнце светит ярко и ветер шелестит листвой. Вдали слышен смех детей. Жизнь прекрасна и удивительна! |
| **Вывод:** Полученная последовательность:  000000010000011000100001000001001110011101010111001000000011110001101001000101011011000110111011000000000011010110100010  110000000111100010011110111001011100010001110001000011101110010001101001000101010001001010001110010001100000000001111001  011100111010111010011110110011100000010010100010101110000010011100111110001000111001000010101110011101010001111110110001  011101100001111000100111110010111001110101111011111010011110001001000111101001011111100110111000111110000010111011100001  1100001000111000010110100010111111100110110010100100000101001000100111000100111101000100011100011111110  Размер: 583  Коэффициент сжатия: 1.372213  Дисперсия: 1.461100 |

**Вывод:** по итогу лабораторной работы получена программа способная кодировать методом Гилберта-Мура.