**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Лабораторная работа №8

По дисциплине «Программирование»

 Биты. Байты. Машинные слова.

Группа: АВТ-041

Студент: Дударь М.А.

Преподаватель: Балакин В.В.

НОВОСИБИРСК 2021

1. **Постановка задачи:**

Необходимо реализовать кодирование и декодирование строки символов, содержащих цифры, в последовательность битов.  Десятичная цифра кодируется 4 битами - одной шестнадцатеричной цифрой. Цифра **F** обозначает, что за ней следует байт (2 цифры) с кодом символа, отличного от цифры. Разработать функции кодирования и декодирования с определением процента уплотнения.

1. **Алгоритм программы:**
2. Конвертирование исходной строки согласно условиям задачи.
3. Декодирование бинарный код обратно в строку.
4. Подсчет процента уплотнения.
5. **Принцип работы:**

Функция конвертирования строки в бинарный код. Причем цифра кодируется 4 битами, а символ 12 битами (четыре единицы и двоичное представление кода символа).

char \*converter(char \*str, int \*len) {  
 \*len = 0;  
 for (int i = 0; i < strlen(str); ++i) {  
 if (str[i] >= '0' && str[i] <= '9') { \*len += 4; }  
 else {  
 \*len += 12;  
 }  
 }  
 char \*cStr = (char \*) malloc(\*len \* sizeof(char));  
 int j = 0, num, k;  
 char \*bin = (char \*) malloc(8 \* sizeof(char));  
 for (int i = 0; i < strlen(str); ++i) {  
 num = str[i] - '0';  
 if (num < 0 || num > 9) {  
 for (int n = j; n <= j + 4; ++n) cStr[n] = '1';  
 j += 4;  
 num = (int) str[i];  
 add(cStr, bin, &j, &k, &num, 8);  
 } else {  
 add(cStr, bin, &j, &k, &num, 4);  
 }  
 }  
 cStr[\*len] = '\0';  
 return cStr;  
}

Функция перевода десятичного числа в двоичную систему счисления.

void add(char \*cStr, char \*bin, int \*j, int \*k, int \*num, int len) {  
 for (\*k = 0; \*num > 0; ++(\*k)) {  
 bin[\*k] = (char) ((\*num % 2) + '0');  
 \*num = \*num / 2;  
 }  
 for (int m = \*k; m < len; ++m) {  
 bin[m] = '0';  
 }  
 reverse(bin, len);  
 for (int n = \*j, m = 0; n <= \*j + len; ++n, ++m) cStr[n] = bin[m];  
 \*j += len;  
}

Функция «переворота строки».

void reverse (char \*str, int len) {  
 for (int i = 0; i < len / 2; i++) {  
 char buf = str[i];  
 str[i] = str [len - i - 1];  
 str [len - i - 1] = buf;  
 }  
}

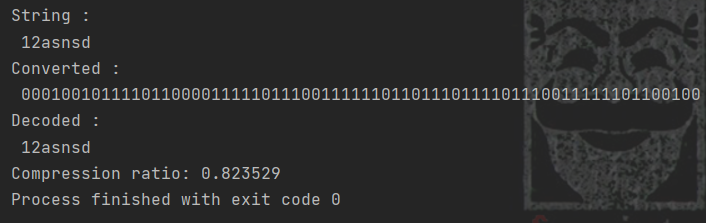
Функция декодирования бинарного кода в строку. Если встретились четыре подряд идущие единицы, значит следующий байт — это символ, иначе следующие 4 бита число.

char \*decoder(char \*str) {  
 int len, c = 0, flag, num, chCount = 0;  
 for (int i = 0; i < strlen(str); ++i) {  
 if (i % 4 == 0) {  
 flag = 1;  
 for (int j = i; j < i + 4; ++j) {  
 if (str[i] == '1' && flag) flag = 1;  
 else flag = 0;  
 }  
 if (flag) {  
 chCount++;  
 i += 11;  
 }  
 }  
 }  
 len = (int) strlen(str) / 4 - 2 \* chCount;  
 char \*dStr = (char \*) malloc(len \* sizeof(char));  
 char \*bin = (char \*) malloc(4 \* sizeof(char));  
 flag = 0;  
 for (int i = 0, j = 0; i <= strlen(str); ++i, ++j) {  
 num = 0;  
 if (flag == 1) {  
 if (j == 8) {  
 for (int k = 7; k >= 0; --k) num += (bin[k] - '0') \* (int) pow(2, 7 - k);  
 dStr[c++] = (char) (num);  
 j = 0;  
 flag = 0;  
 }  
 } else if (j == 4) {  
 for (int k = 3; k >= 0; --k) num += (bin[k] - '0') \* (int) pow(2, 3 - k);  
 if (num == 15) {  
 flag = 1;  
 j = 0;  
 bin[j] = str[i];  
 continue;  
 }  
 dStr[c++] = (char) (num + '0');  
 j = 0;  
 }  
 bin[j] = str[i];  
 }  
 dStr[len] = '\0';  
 return dStr;  
}

1. **Вывод:**

В ходе выполнения задания была произведена работа с кодированием информации в двоичные код и его декодированием на языке C. Подсчитан коэффициент уплотнения, и выяснено, что если в исходной строке больше цифр чем символов, то кодирование эффективно.

Приложение 1. Вывод программы.



Приложение 2. Текст программы.

#include <stdio.h>  
#include <string.h>  
#include <stdlib.h>  
#include <math.h>  
  
void reverse(char \*str, int len) {  
 for (int i = 0; i < len / 2; i++) {  
 char buf = str[i];  
 str[i] = str[len - i - 1];  
 str[len - i - 1] = buf;  
 }  
}  
  
void add(char \*cStr, char \*bin, int \*j, int \*k, int \*num, int len) {  
 for (\*k = 0; \*num > 0; ++(\*k)) {  
 bin[\*k] = (char) ((\*num % 2) + '0');  
 \*num = \*num / 2;  
 }  
 for (int m = \*k; m < len; ++m) {  
 bin[m] = '0';  
 }  
 reverse(bin, len);  
 for (int n = \*j, m = 0; n <= \*j + len; ++n, ++m) cStr[n] = bin[m];  
 \*j += len;  
}  
  
char \*converter(char \*str, int \*len) {  
 \*len = 0;  
 for (int i = 0; i < strlen(str); ++i) {  
 if (str[i] >= '0' && str[i] <= '9') { \*len += 4; }  
 else {  
 \*len += 12;  
 }  
 }  
 char \*cStr = (char \*) malloc(\*len \* sizeof(char));  
 int j = 0, num, k;  
 char \*bin = (char \*) malloc(8 \* sizeof(char));  
 for (int i = 0; i < strlen(str); ++i) {  
 num = str[i] - '0';  
 if (num < 0 || num > 9) {  
 for (int n = j; n <= j + 4; ++n) cStr[n] = '1';  
 j += 4;  
 num = (int) str[i];  
 add(cStr, bin, &j, &k, &num, 8);  
 } else {  
 add(cStr, bin, &j, &k, &num, 4);  
 }  
 }  
 cStr[\*len] = '\0';  
 return cStr;  
}  
  
char \*decoder(char \*str) {  
 int len, c = 0, flag, num, chCount = 0;  
 for (int i = 0; i < strlen(str); ++i) {  
 if (i % 4 == 0) {  
 flag = 1;  
 for (int j = i; j < i + 4; ++j) {  
 if (str[i] == '1' && flag) flag = 1;  
 else flag = 0;  
 }  
 if (flag) {  
 chCount++;  
 i += 11;  
 }  
 }  
 }  
 len = (int) strlen(str) / 4 - 2 \* chCount;  
 char \*dStr = (char \*) malloc(len \* sizeof(char));  
 char \*bin = (char \*) malloc(4 \* sizeof(char));  
 flag = 0;  
 for (int i = 0, j = 0; i <= strlen(str); ++i, ++j) {  
 num = 0;  
 if (flag == 1) {  
 if (j == 8) {  
 for (int k = 7; k >= 0; --k) num += (bin[k] - '0') \* (int) pow(2, 7 - k);  
 dStr[c++] = (char) (num);  
 j = 0;  
 flag = 0;  
 }  
 } else if (j == 4) {  
 for (int k = 3; k >= 0; --k) num += (bin[k] - '0') \* (int) pow(2, 3 - k);  
 if (num == 15) {  
 flag = 1;  
 j = 0;  
 bin[j] = str[i];  
 continue;  
 }  
 dStr[c++] = (char) (num + '0');  
 j = 0;  
 }  
 bin[j] = str[i];  
 }  
 dStr[len] = '\0';  
 return dStr;  
}  
  
int main() {  
 int len;  
 char \*str = {"1243654367545$%^&\*("};  
 printf("String :\n %s\n", str);  
 char \*con = converter(str, &len);  
 printf("Converted :\n %s\n", con);  
 char \*dec = decoder(con);  
 printf("Decoded :\n %s\n", dec);  
 printf("Compression ratio: %f", (float) (8 \* (float) strlen(str) / (float) (len)));  
  
 return 0;  
}