**Кодирование**

**1. Основные понятия**

*Закодировать* текст – значит сопоставить ему другой текст. *Кодирование* применяется при передаче данных – для того, чтобы зашифровать текст от посторонних, чтобы сделать передачу данных более надежной, потому что канал передачи данных может передавать только ограниченный набор символов (например, - только два символа, 0 и 1) и по другим причинам.

При кодировании заранее определяют алфавит, в котором записаны исходные тексты (*исходный* алфавит) и алфавит, в котором записаны закодированные тексты (*коды*), этот алфавит называется *кодовым* алфавитом. В качестве кодового алфавита часто используют *двоичный* алфавит, состоящий из двух символов (*битов*) 0 и 1. Слова в двоичном алфавите иногда называют битовыми последовательностями.

**2. Побуквенное кодирование**

Наиболее простой способ кодирования – побуквенный. При побуквенном кодировании каждому символу из исходного алфавита сопоставляется *кодовое слово –* слово в кодовом алфавите. Иногда вместо «кодовое слово буквы» говорят просто «*код буквы*». **При побуквенном кодировании текста коды всех символов записываются подряд, без разделителей.**

***Пример 1.*** Исходный алфавит – алфавит русских букв, строчные и прописные буквы не различаются. Размер алфавита – 33 символа.

Кодовый алфавит – алфавит десятичных цифр. Размер алфавита  - 10 символов.

Применяется побуквенное кодирование по следующему правилу: буква кодируется ее номером в алфавите: код буквы А – 1; буквы Я – 33 и т.д.

Тогда код слова АББА – это 1221.

***Внимание:*** Последовательность 1221 может означать не только АББА, но и КУ (К – 12-я буква в алфавите, а У – 21-я буква). Про такой код говорят, что он НЕ допускает *однозначного декодирования*

***Пример 2.***Исходный и кодовый алфавиты – те же, что в примере 1. Каждая буква также кодируется своим номером в алфавите, НО номер всегда записывается *двумя* цифрами: к записи однозначных чисел слева добавляется 0. Например, код А – 01, код Б – 02 и т.д.

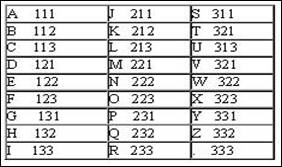
В этом случае кодом текста АББА будет 01020201. И расшифровать этот код можно только одним способом. Для расшифровки достаточно разбить кодовый текст 01020201 на двойки:  01 02 02 01 и для каждой двойки определить соответствующую ей букву.

Такой способ кодирования называется ***равномерным.*** Равномерное кодирование всегда допускает однозначное декодирование.

Код называется **равномерным** (или кодом **постоянной длины**), если все его кодовые слова содержат одинаковое число букв (одинаковую длину слов). Соответственно, кодирование называется **равномерным**, если соответствующий ему код имеет постоянную длину.

К равномерным кодам относится телеграфный код Бодо (Baudot code). Его можно считать двоичным равномерным алфавитным кодом. Первоначальный вариант этого кода разработал Эмиль Бодо в 1870 году для своего телеграфа. Код вводился прямо клавиатурой, состоящей из пяти клавиш, нажатие или ненажатие клавиши соответствовало передаче или непередаче одного бита в пятибитном коде. Например, буква **А** передавалась как **- - + - -** , что соответствовало нажатию средней клавиши. В двоичном коде это можно записать как **00100**. Таким образом, каждая буква записывалась пятью битами. Следовательно, кодом Бодо можно было передать **25=32** различных символа.

Другим интересным примером равномерного кода является код Трисиме, в котором знакам латинского алфавита ставятся в соответствие кодовые слова длины 3 над алфавитом из 3-х символов: {1, 2, 3}. Этот код представлен в следующей таблице :



Понятно, что код Трисиме не может кодировать более чем **33=27** символов.

Число букв в алфавите кода называется **основанием** кода, а длина кодовых слов равномерного кода называется **порядком** кода. Коды с основанием 2, как уже говорилось, называются **двоичными**, а с основанием 3 – **троичными**, и так далее. Так код Бодо имеет основание 2, а порядок 5, а у кода Трисиме и основание, и порядок равны 3.

Пусть кодовый алфавит содержит к букв, а длина элементарного кода n букв. Сколько символов (m) при равномерном кодировании можно закодировать?

Элементарные коды ─ это кортежи из n букв, взятых из данных к, поэтому они являются размещениями с повторениями. По известной формуле их число равно

**m = кn** (1)

Поставим обратную задачу. Имеется m букв исходного алфавита. Сколько букв должны содержать элементарные коды при равномерном кодировании?

Из формулы (1), логарифмируя, получи

**n = log к m**(2)

Если же алфавит В={0,1} – двоичный, то закодировать n – битным кодом можно в силу формулы (1) имеем букв

**m = 2 n** (3)

Пусть имеется алфавит А, состоящий из m символов. Сколько бит должен содержать каждый элементарный код при равномерном кодировании?

Из формулы (2) получим, если m – степень 2,

**n = log 2m**(4)

Если m – не степень 2, то по формуле:

**n=| log 2m | + 1** (5)

где | **log 2m** | - целая часть **log 2m**..

Формулы (4) - (5) называются **формулами Хартли.**

Например.

Если в качестве алфавита А взять русский алфавит, который содержит 33 символа и пробел, то для 34 символов: по формуле Хартли

n =| **log 234** | +1=5+1= 6 бит. При этом мы можем 6-битным кодом закодировать еще 30 символов. Такое кодирование называется **избыточным**. В компьютерах применяется 1-байтовое кодирование (256 символов) и оно также является избыточным.

# Постановка задачи.

# Закодировать текст равномерным двоичным кодом.

# Пояснение.

Выделить алфавит строки **(**т. е. множество всех различных символов исходной строки); каждому символу алфавита присвоить порядковый номер, начиная с 0; порядковый номер перевести в двоичную систему; довести нулями коды до одинаковой длины.

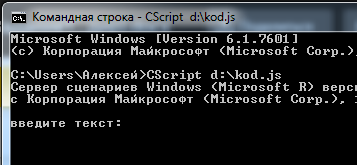
Вход программы: некоторый текст (ввод с консоли).

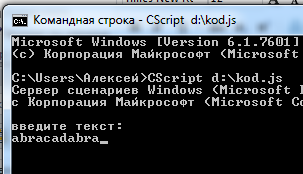
Результаты работы программы:

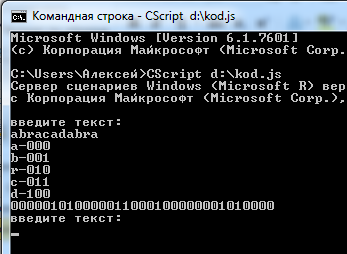
1) код каждого символа.

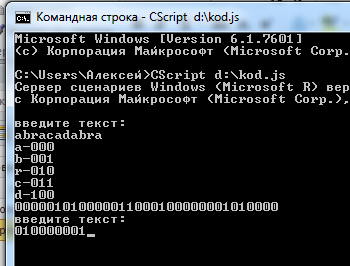
2) закодированный текст.

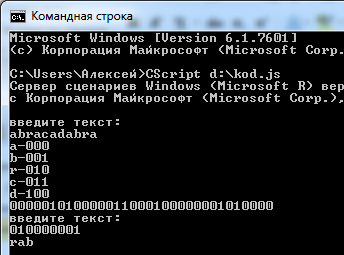
3) раскодировать последовательность кодов, которая вводится с консоли.











Напоминание.

CScript

str=WScript.StdIn.ReadLine() - чтение с консоли

Node.js

Ввод с консоли

let readline = require('readline');

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout

});

rl.question('Ввод', (answer) => {

rl.close();

let output = answer[1]; действия с введенным выражением

console.log(output);

2-й ввод с консоли:

rl.question('Ввод', (answer) => {

rl.close();

let output1 = answer[2];

console.log(output1);

});

});