Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

*Факультет программной инженерии и компьютерной техники*

*Направление подготовки: 09.03.04 – Программная инженерия*

*Дисциплина «Информатика»*

**Отчет**

**По лабораторной работе №2**

**"Синтез помехоустойчивого кода»**

**Вариант №28**

Выполнил:

Белогубов Григорий Дмитриевич

Группа: Р3118

Преподаватель:

Рыбаков Степан Дмитриевич

г. Санкт-Петербург

2023

**ОГЛАВЛЕНИЕ**

Оглавление

[ЗАДАНИЕ 3](#_Toc147890056)

[ОСНОВНЫЕ ЭТАПЫ ВЫЧИСЛЕНИЯ 5](#_Toc147890057)

[**ЗАДАНИЕ №1** 5](#_Toc147890058)

[Пример №1. 5](#_Toc147890059)

[Пример №2. 6](#_Toc147890060)

[Пример №3. 6](#_Toc147890061)

[Пример №4. 7](#_Toc147890062)

[Пример №5. 8](#_Toc147890063)

[**ЗАДАНИЕ №2** 10](#_Toc147890064)

[**ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ №1** 11](#_Toc147890065)

[ЗАКЛЮЧЕНИЕ 15](#_Toc147890066)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 16](#_Toc147890067)

# **ЗАДАНИЕ**

1. Построить схему декодирования классического кода Хэмминга (для 1–4 примера (7;4), для 5 (15;11)). Показать, имеются ли в принятых сообщениях ошибки, и если имеются, то какие. Записать правильное сообщение.

|  |  |  |
| --- | --- | --- |
| № пример | Код Хэмминга | Полученное сообщение |
| 1 | (7;4) | 1010001 |
| 2 | (7;4) | 1010011 |
| 3 | (7;4) | 1000101 |
| 4 | (7;4) | 0111111 |
| 5 | (15;11) | 011000111010001 |

1. Сложить номера всех 5 вариантов заданий. Умножить полученное число на 4. Принять данное число как число информационных разрядов в передаваемом сообщении. Вычислить для данного числа минимальное число проверочных разрядов и коэффициент избыточности.
2. Дополнительное задание. Написать программу на любом языке программирования, которая на вход получает набор из 7 цифр «0» и «1», записанных подряд, анализирует это сообщение на основе классического кода Хэмминга (7,4), а затем выдает правильное сообщение (только информационные биты) и указывает бит с ошибкой при его наличии.

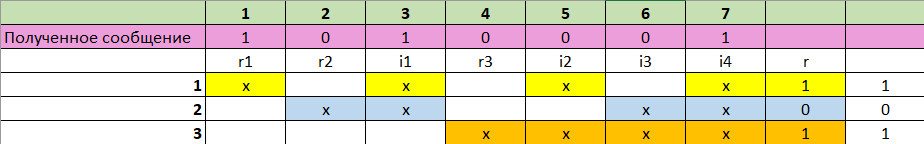
# ОСНОВНЫЕ ЭТАПЫ ВЫЧИСЛЕНИЯ

## 

## **ЗАДАНИЕ №1**

### Пример №1.

**Полученное сообщение: 1010001**



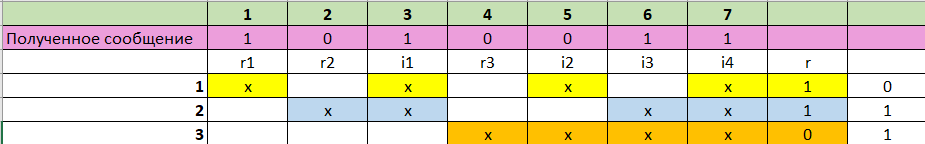
*Рисунок 1*

После получения сумм, видим, что 2-й проверочный бит не совпадает с их текущим значением, значит ошибка в информационном бите i2, который надо инвертировать, чтобы исправить ошибку.

**Верное сообщение: 1010101**

### Пример №2.

**Полученное сообщение: 1010011**

****

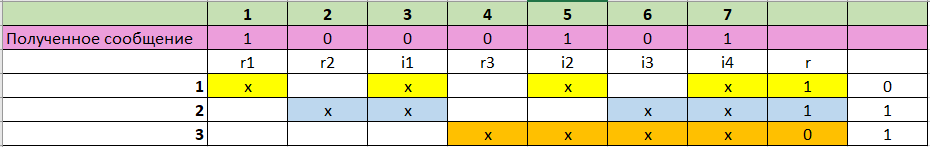
*Рисунок 2*

После получения сумм, видим, что 3-й проверочный бит не совпадает с их текущим значением, значит ошибка в информационном бите i1, который надо инвертировать, чтобы исправить ошибку.

**Верное сообщение: 1000**

### Пример №3.

**Полученное сообщение:** 1000101



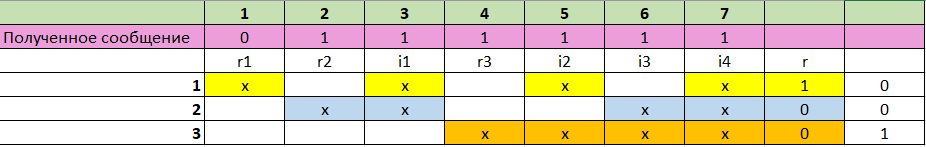
*Рисунок 3*

После получения сумм, видим, что 3-й проверочный бит не совпадает с их текущим значением, значит ошибка в информационном бите i1, который надо инвертировать, чтобы исправить ошибку.

**Верное сообщение:** 1010101

### Пример №4.

**Полученное сообщение: 0111111**



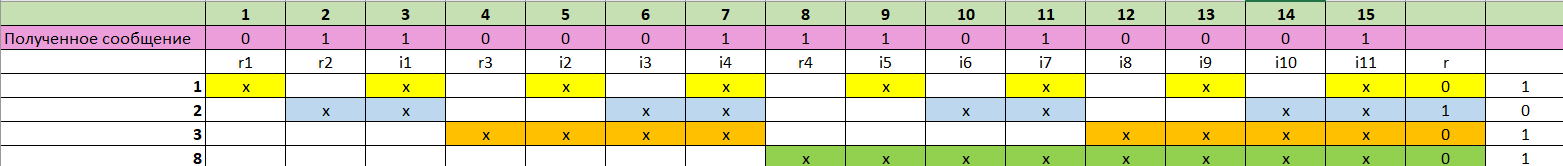
*Рисунок 4*

После получения сумм, видим, что 2-й и 3-й проверочные биты не совпадают с их текущими значениями, значит ошибка в информационном бите r1, который надо инвертировать, чтобы исправить ошибку.

**Верное сообщение: 1111111**

### Пример №5.

**Полученное сообщение:** 011000111010001

*Рисунок 5*

После получения сумм, видим, что 1-й, 3-й и 4-й проверочные биты не совпадает с их текущим значением, значит ошибка в информационном бите i7, который надо инвертировать, чтобы исправить ошибку.

**Верное сообщение:** 011000111000001

## **ЗАДАНИЕ №2**

Сумма всех 5 вариантов заданий: 24 + 51 + 78 + 105 + 29 = 287

Эта сумма, умноженная на 4: 287 \* 4 = 1148

Соответственно, i (число информационных разрядов) = 1148

Тогда, высчитаем минимальное число проверочных разрядов для i по формуле 2r >= r + i + 1, где r – число проверочных разрядов:

2r  >= r + 1148 + 1

2r  >= r + 1148

Методом подбора, определим минимальное значение r:

|  |  |
| --- | --- |
| **Значение r** | **2r  >= r + 1149** |
| 1 | 2 >= 1150 (ложь) |
| … | … |
| 9 | 512 >= 1157 (ложь) |
| 10 | 1024 >= 1158 (ложь) |
| *11* | *2048 >= 1159(истина)* |
| … | … |

Таблица 9

Соответственно, минимальное число проверочных разрядов (r) равно 11.

Вычислим коэффициент избыточности по формуле: КИ = r / (i + r)

Соответственно, КИ = 11 / (11 +1148) ≈ 0,0094

**ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ № 1**

import java.util.Scanner;

public class Main {

public static void main(String[] yolo) {

Scanner readCode = new Scanner(System.in);

int stop = 0;

System.out.println("Введите сообщение 7-ми значное сообщение:");

String code = readCode.nextLine();

readCode.close();

if (code.length() != 7) System.out.println("Введены некорректные данные.");

else {

String[] mass = code.split("");

int[] codeChars = new int[7];

for (int i = 0; i < 7; i++) {

if (mass[i].equals("1") || mass[i].equals("0")) codeChars[i] = Integer.parseInt(mass[i]);

else {

stop++;

if (stop == 1) System.out.println("Введены некорректные данные.");

}

}

if (stop == 0) {

int s1 = (codeChars[0] + codeChars[2] + codeChars[4] + codeChars[6]) % 2;

int s2 = (codeChars[1] + codeChars[2] + codeChars[5] + codeChars[6]) % 2;

int s3 = (codeChars[3] + codeChars[4] + codeChars[5] + codeChars[6]) % 2;

String str = Integer.toString(s1) + (s2) + (s3);

switch (str) {

case "000":

System.out.print("Вы ввели правильное сообщение и бита с ошибкой нет: " + code + "." + "\n");

break;

case "001":

codeChars[3] = (codeChars[3] == 0) ? 1 : 0;

System.out.print("Правильное сообщение: ");

for (int i = 0; i < 7; i++) System.out.print(codeChars[i]);

System.out.print("." + "\n");

System.out.print("Бит с ошибкой: побочный бит r3. Позиция: 4." + "\n");

break;

case "010":

codeChars[1] = (codeChars[1] == 0)? 1: 0;

System.out.print("Правильное сообщение: ");

for (int i = 0; i < 7; i++) System.out.print(codeChars[i]);

System.out.print("." + "\n");

System.out.print("Бит с ошибкой: побочный бит r2. Позиция: 2." + "\n");

break;

case "011":

codeChars[5] = (codeChars[5] == 0)? 1: 0;

System.out.print("Правильное сообщение: ");

for (int i = 0; i < 7; i++) System.out.print(codeChars[i]);

System.out.print("." + "\n");

System.out.print("Бит с ошибкой: информационный бит i3. Позиция: 6." + "\n");

break;

case "100":

codeChars[0] = (codeChars[0] == 0)? 1: 0;

System.out.print("Правильное сообщение: ");

for (int i = 0; i < 7; i++) System.out.print(codeChars[i]);

System.out.print("." + "\n");

System.out.print("Бит с ошибкой: побочный бит r1. Позиция: 1." + "\n");

break;

case "101":

codeChars[4] = (codeChars[4] == 0)? 1: 0;

System.out.print("Правильное сообщение: ");

for (int i = 0; i < 7; i++) System.out.print(codeChars[i]);

System.out.print("." + "\n");

System.out.print("Бит с ошибкой: информационный бит i2. Позиция: 5." + "\n");

break;

case "110":

codeChars[2] = (codeChars[2] == 0)? 1: 0;

System.out.print("Правильное сообщение: ");

for (int i = 0; i < 7; i++) System.out.print(codeChars[i]);

System.out.print("." + "\n");

System.out.print("Бит с ошибкой: информационный бит i1. Позиция: 3." + "\n");

break;

case "111":

codeChars[6] = (codeChars[6] == 0)? 1: 0;

System.out.print("Правильное сообщение: ");

for (int i = 0; i < 7; i++) System.out.print(codeChars[i]);

System.out.print("." + "\n");

System.out.print("Бит с ошибкой: информационный бит i4. Позиция: 7." + "\n");

break;

}

}

}

}

}

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения данной лабораторной работы я изучил код Хэмминга: научился строить таблицы кода Хэмминга, вычислять число проверочных разрядов из числа информационных разрядов, а затем и вычислять коэффициент избыточности; научился искать ошибки в сообщениях в виде кода Хемминга и исправлять их. Кроме того, мне удалось написать код на Java, который выводит ошибочный бит, инвертирует его и выводит правильное сообщение

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

* **Балакшин П.В. Соснин В.В., Машина Е.А.** – СПб: Университет ИТМО, 2020 // Информатика. Методическое пособие "Информатика". Раздел 2 "Системы счисления". <https://vk.com/doc-31201840_566998093>
* Код Хэмминга. Пример работы алгоритма. // Хабр

URL: https://habr.com/ru/articles/140611/ (дата обращения: 01.10.23).