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

Факультет программной инженерии и компьютерной техники

Лабораторная работа №2

Синтез помехоустойчивого кода

Вариант №68

Выполнил: студент группы P3108 Васильев Никита

Проверил: Балакшин Павел Валерьевич, доцент факультета ПИиКТ, кандидат технических наук

Санкт-Петербург 2023

Содержание

[Задание 3](#_Toc147569162)

[Основные этапы вычисления 4](#_Toc147569163)

[Код Хэмминга (7; 4) 4](#_Toc147569164)

[Задание 1 – №50 4](#_Toc147569165)

[Задание 2 – №87 5](#_Toc147569166)

[Задание 3 – №12 5](#_Toc147569167)

[Задание 4 – №91 6](#_Toc147569168)

[Код Хэмминга (15; 11) 6](#_Toc147569169)

[Задание 5 – №68 6](#_Toc147569170)

[Задание 6 6](#_Toc147569171)

[Написание программы для определения ошибок в битах на языке программирования Python 7](#_Toc147569172)

[Заключение 8](#_Toc147569173)

[Список литературы 9](#_Toc147569174)

# Задание

Построить схему декодирования классического кода Хэмминга (7;4), которую представить в отчёте в виде изображения.

Показать, исходя из выбранных вариантов сообщений (по 4 у каждого – часть №1 в варианте), имеются ли в принятом сообщении ошибки, и если имеются, то какие. Подробно прокомментировать и записать правильное сообщение.

Построить схему декодирования классического кода Хэмминга (15;11), которую представить в отчёте в виде изображения.

Показать, исходя из выбранного варианта сообщений (по 1 у каждого – часть №2 в варианте), имеются ли в принятом сообщении ошибки, и если имеются, то какие. Подробно прокомментировать и записать правильное сообщение.

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.

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.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вариант | 1 | | | | 2 |
| 68 | 50 | 87 | 12 | 91 | 68 |

# Основные этапы вычисления

## Код Хэмминга (7; 4)

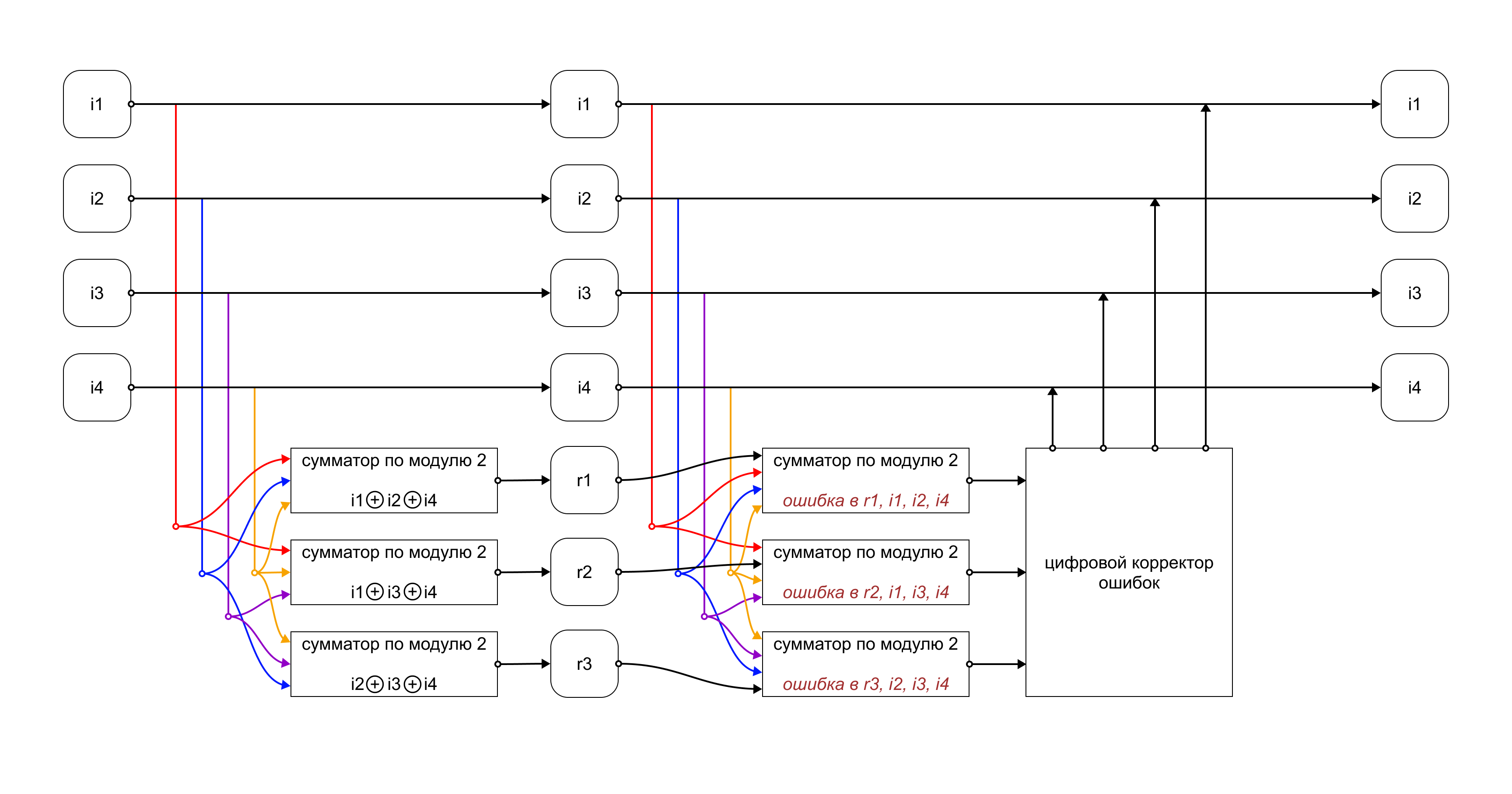


Рисунок 1 – схема классического кода Хэмминга (7, 4)

### Задание 1 – №50

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 |
| 50 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |

s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 1 ⊕ 0 ⊕ 0 ⊕ 1 = 0

s2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 0 ⊕ 0 ⊕ 1 ⊕ 1 = 0

s3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 1 ⊕ 0 ⊕ 1 ⊕ 1 = 1

s = 001 → Согласно Рисунку 2 ошибка в бите r3 → Правильное сообщение: 0011

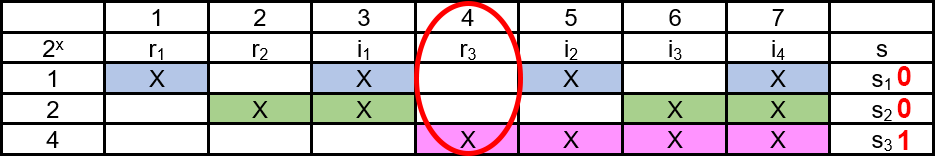


Рисунок 2 - Код Хэмминга с синдромом 001

### Задание 2 – №87

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 |
| 87 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |

s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 0 ⊕ 1 ⊕ 1 ⊕ 0 = 0

s2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 0 ⊕ 1 ⊕ 1 ⊕ 0 = 0

s3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 1 ⊕ 1 ⊕ 1 ⊕ 0 = 1

s = 001 → Согласно Рисунку 3 ошибка в бите r3 → Правильное сообщение: 1110

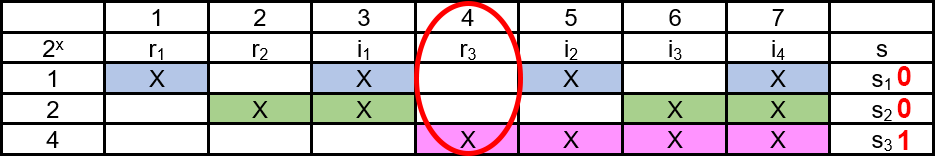


Рисунок 3 - Код Хэмминга с синдромом 001

### Задание 3 – №12

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 |
| 12 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 1 ⊕ 0 ⊕ 0 ⊕ 0 = 1

s2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 1 ⊕ 0 ⊕ 0 ⊕ 0 = 1

s3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 0 ⊕ 0 ⊕ 0 ⊕ 0 = 0

s = 110 → Согласно Рисунку 4 ошибка в бите i1 → Правильное сообщение: 1000

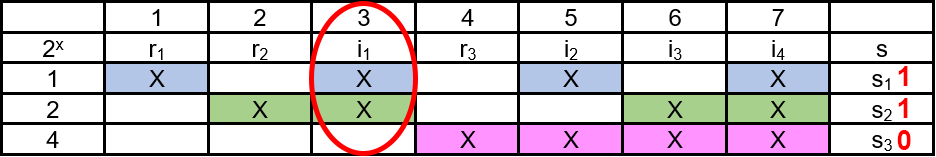


Рисунок 4 - Код Хэмминга с синдромом 110

### Задание 4 – №91

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 |
| 91 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |

s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 0 ⊕ 1 ⊕ 1 ⊕ 0 = 0

s2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 1 ⊕ 1 ⊕ 1 ⊕ 0 = 1

s3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 1 ⊕ 1 ⊕ 1 ⊕ 0 = 1

s = 011 → Согласно Рисунку 5 ошибка в бите i3 → Правильное сообщение: 1100

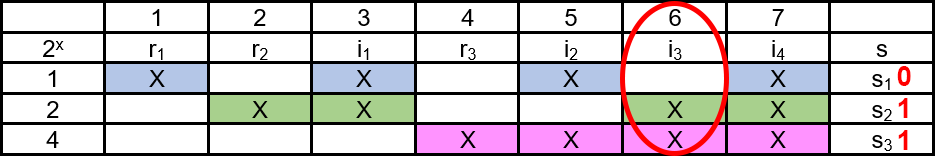


Рисунок 5 - Код Хэмминга с синдромом 011

## Код Хэмминга (15; 11)

### Задание 5 – №68

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 | r4 | i5 | i6 | i7 | i8 | i9 | i10 | i11 |
| 68 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |

s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 ⊕ i5 ⊕ i7 ⊕ i9 ⊕ i11 = 0 ⊕ 1 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 0 = 0

s2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 ⊕ i6 ⊕ i7 ⊕ i10 ⊕ i11 = 0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 0 = 1

s3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 ⊕ i8 ⊕ i9 ⊕ i10 ⊕ i11 = 1 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 = 0

s4 = r4 ⊕ i5 ⊕ i6 ⊕ i7 ⊕ i8 ⊕ i9 ⊕ i10 ⊕ i11 = 0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 = 1

s = 0101 → Согласно Рисунку 6 ошибка в бите i6 → Правильное сообщение: 11001100100

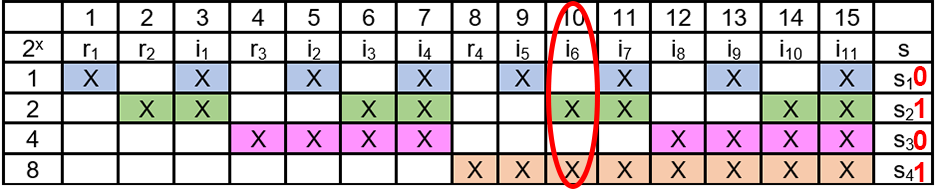


Рисунок 6 – Код Хэмминга с синдромом 0101

## Задание 6

Количество информационных разрядов *i* = (50 + 87+ 12 + 91 + 68) × 4 = 1232.

Определение минимального числа контрольных разрядов *r*: 2r ≥ r + i + 1.

Найдём *r*, для которого будет верным неравенство: 2r ≥ r + 1233.

Неравенство верно, начиная с *r* = 11: 211 ≥ 1244. Следовательно, необходимо 11 контрольных разрядов *r.*

Коэффициент избыточности = *r* ÷ *n* = 11 ÷ 1243 ≈ 0,00884956.

Ответ: *r* = 11; коэффициент избыточности ≈ 0,00884956.

## Написание программы для определения ошибок в битах на языке программирования Python

ent\_code = str(input("Введите код Хэмминга: "))  
  
#определяем, является введенный код кодом Хэмминга  
def code\_is\_code (ent\_code):  
 byte\_code = True  
 for i in ent\_code:  
 if i not in '01':  
 byte\_code = False  
 if len(ent\_code) != 7 or byte\_code == False:  
 return 'Введенный код не является кодом Хэмминга'  
 else:  
 return code\_Hemming(ent\_code)  
  
#создаём словарь для хранения значений битов  
bv = {}  
bv['r1'], bv['r2'], bv['i1'], bv['r3'], bv['i2'], bv['i3'], bv['i4'] = ent\_code[:7]  
  
#строим синдромы  
def code\_Hemming (ent\_code):  
 s1 = int(bv['r1']) ^ int(bv['i1']) ^ int(bv['i2']) ^ int(bv['i4'])  
 s2 = int(bv['r2']) ^ int(bv['i1']) ^ int(bv['i3']) ^ int(bv['i4'])  
 s3 = int(bv['r3']) ^ int(bv['i2']) ^ int(bv['i3']) ^ int(bv['i4'])  
 s = str(s1) + str(s2) + str(s3)  
 if s == '000':  
 return 'Ошибок нет'  
 else:  
 s = s[::-1]  
 pos = int(s, 2)  
 cnt = 0  
 for err\_elem, err\_result in bv.items():  
 cnt += 1  
 if cnt == pos:  
 return right\_code(err\_elem, pos)  
  
#выводим правильный код  
def right\_code(err\_elem, pos):  
 if err\_elem in 'i1' or err\_elem in 'i2' or err\_elem in 'i3' or err\_elem in 'i4':  
 err\_byte = bv[err\_elem]  
 global ent\_code  
 if int(err\_byte) == 0:  
 ent\_code = ent\_code[:pos - 1] + '1' + ent\_code[pos:]  
 else:  
 ent\_code = ent\_code[:pos - 1] + '0' + ent\_code[pos:]  
 print('Ошибка в бите', err\_elem)  
 print('Правильный код:', str(ent\_code[2] + ent\_code[4:]))  
  
  
code\_is\_code(ent\_code)

# Заключение

В ходе выполнения данной лабораторной работы я познакомился с помехоустойчивым кодированием, научился работать с кодом Хэмминга, определять ошибки, возникшие при передаче или хранении данных, вычислять необходимое количество контрольных разрядов и коэффициент избыточности.

# Список литературы

**Липницкий В. А. Чесалин Н. В.** Линейные коды и коды последовательности [Книга]. - Минск : БГУ, 2008. - стр. 41.

**Ромащенко А. Румянцев А., Шень А.** Заметки по теории кодирования [Книга]. - Москва : МЦНМО, 2017. - 2-е : стр. 88.