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

Факультет программной инженерии и компьютерной техники

**Отчет**

По лабораторной работа № 2 по дисциплине “Информатика”

Синтез помехоустойчивого кода

Вариант № 94

Студент:

Мирзаитов Тимур

Группа P3112

Преподаватель:

Рыбаков Степан Дмитриевич

Санкт-Петербург 2023

Оглавление

[Задание 2](#_Toc147096190)

[Основные этапы вычисления 3](#_Toc147096191)

[Заключение 8](#_Toc147096192)

[Список использованных источников 9](#_Toc147096193)

# Задание

1. На основании номера варианта задания выбрать набор из 4 полученных

сообщений в виде последовательности 7-символьного кода.

2. Построить схему декодирования классического кода Хэмминга (7;4),

которую представить в отчёте в виде изображения.

3. 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.

4. На основании номера варианта задания выбрать 1 полученное сообщение в

виде последовательности 11-символьного кода.

5. Построить схему декодирования классического кода Хэмминга (15;11),

которую представить в отчёте в виде изображения.

6. 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.

7. 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.

8. Дополнительное задание №1 (позволяет набрать от 86 до 100 процентов от

максимального числа баллов БаРС за данную лабораторную). 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.



Рисунок 1 Варианты заданий

# Основные этапы вычисления

Часть 1

Набор из 4 полученных сообщений

1) №105 - «0111111»

2) №26 - «1100001»

3) №64 - «1000100»

4) №10 - «1010000»

1)

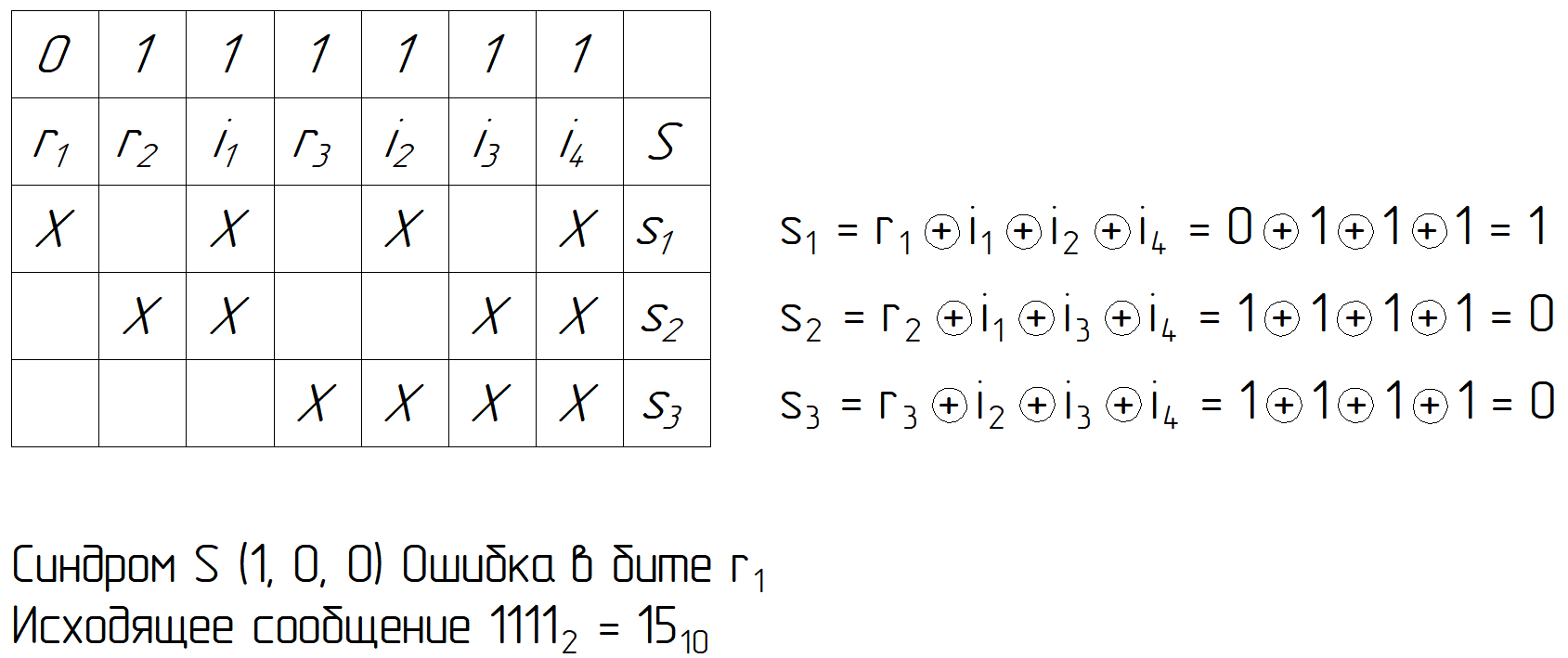


Рисунок №1 - Декодирование кода Хэмминга для примера №1

В этом сообщении есть ошибка в первом бите, нашел её с помощью декодирования кода Хэмминга (рис. 1), т.к. синдром S равен 100.

Правильное значение: 1111111

2)

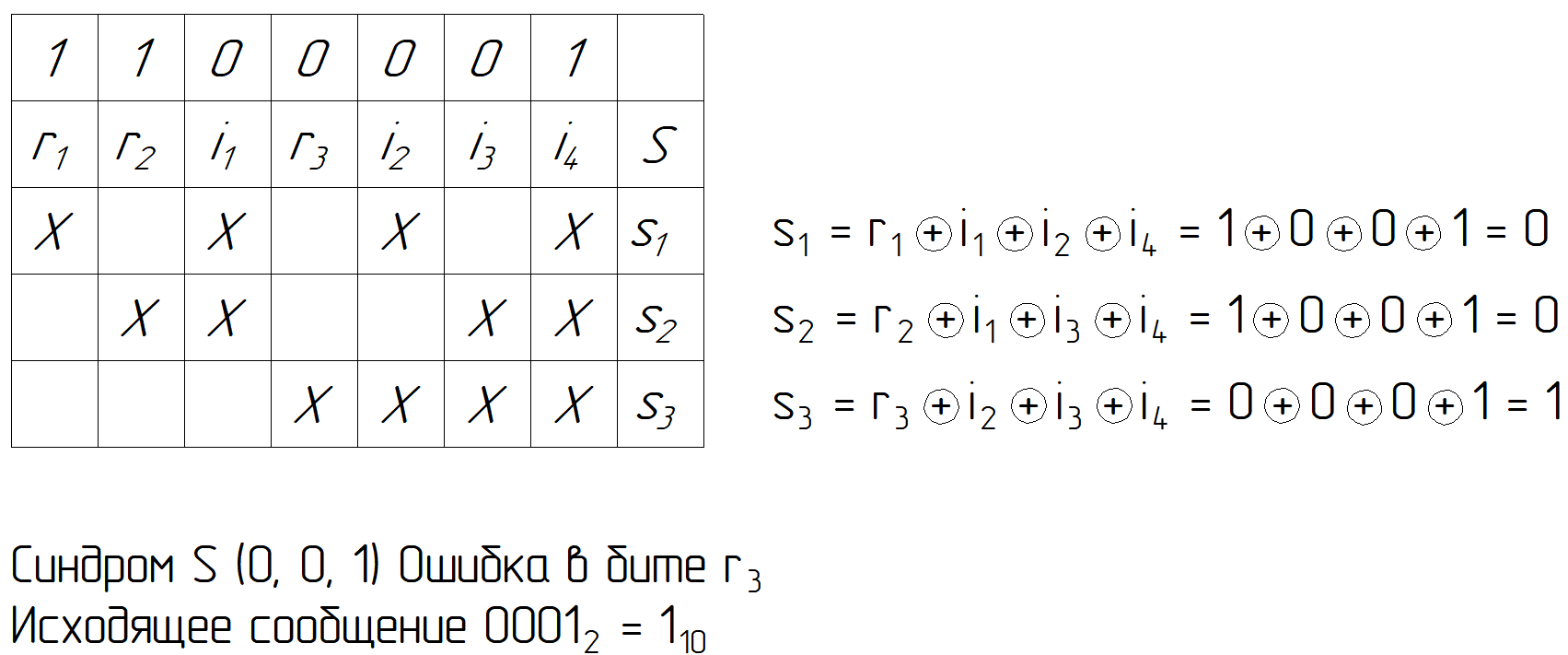


Рисунок №2 - Декодирование кода Хэмминга для примера №2

В этом сообщении есть ошибка в четвертом бите, нашел её с помощью декодирования кода Хэмминга (рис. 2), т.к. синдром S равен 001.

Правильное значение: 1101001

3)

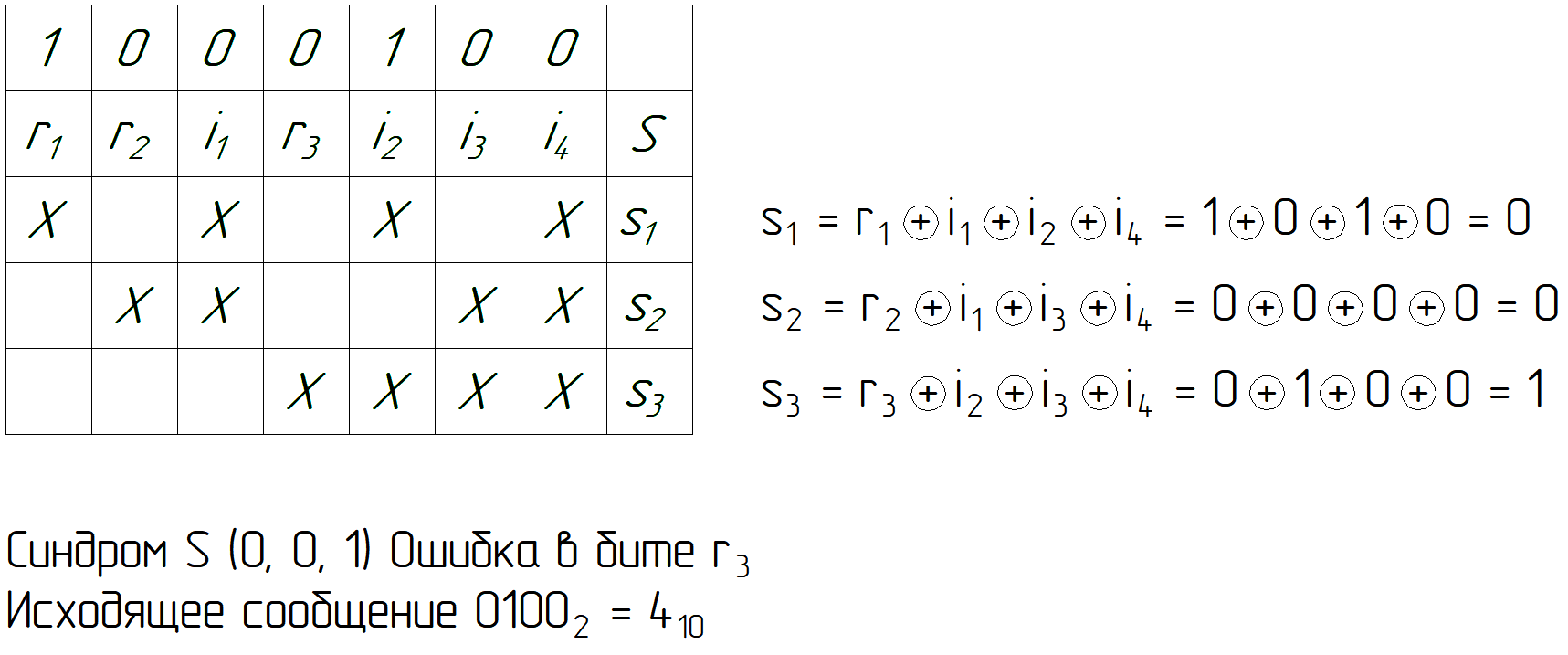


Рисунок №3 - Декодирование кода Хэмминга для примера №3

В этом сообщении есть ошибка в четвёртом бите, нашел её с помощью декодирования кода Хэмминга (рис. 3), т.к. синдром S равен 001.

Правильное значение: 1001100

4)

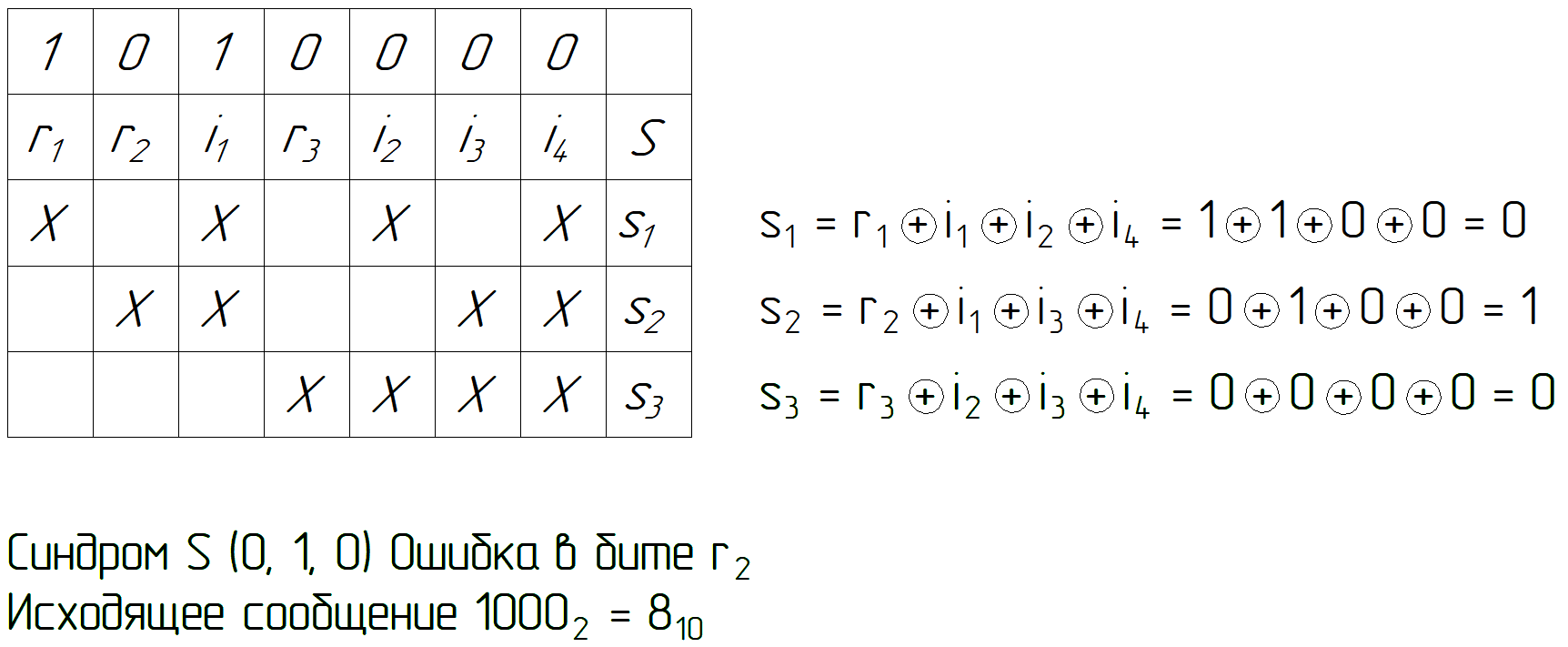


Рисунок №4 - Декодирование кода Хэмминга для примера №4

В этом сообщении есть ошибка во втором бите, нашел её с помощью декодирования кода Хэмминга (рис. 4), т.к. синдром S равен 010.

Правильное значение: 1110000

Часть 2

Полученное сообщение

№84 -­ «001010101000101»

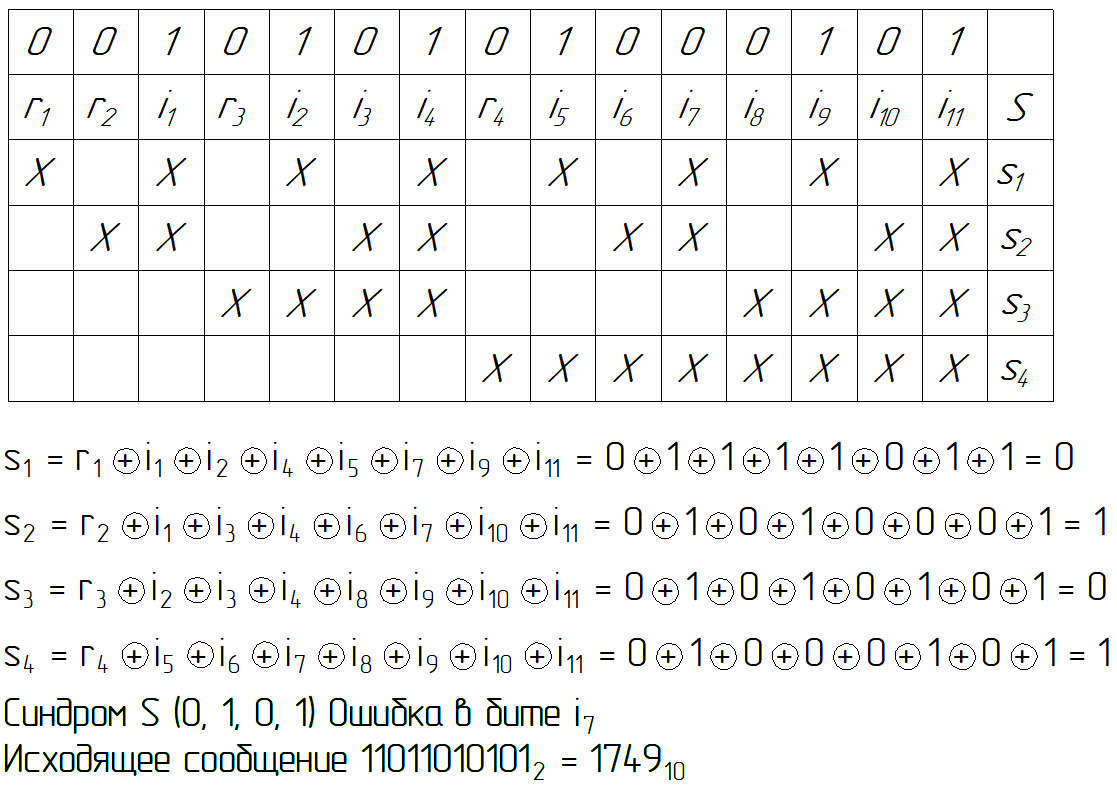


Рисунок №5 - Декодирование кода Хэмминга для примера №1 Части 2

В этом сообщении есть ошибка в одиннадцатом бите, нашел её с помощью декодирования кода Хэмминга (рис. 5), т.к. синдром S равен 0101.

Правильное значение: 001010101010101

Часть 3

105 + 26 + 64 + 10 + 84 = 289

289 \* 4 = 1156

Определение минимального числа контрольных разрядов:

2r ≥ r + i + 1

Где i = 1156

При r = 11, условие выполняется

2048 ≥ 11 + 1156 +1 = 1168

При r = 10:

1024 ≥ 10 + 1056 + 1 = 1067 (ложно)

Значит r = 11 минимально

Посчитаем коэффициент

Дополнительное задание

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90 | **public** **class** **Main** {  **public** **static** **void** **main**(String[] args) {  String input = "0111111";  //String input = args[0];  **boolean** r1, r2, r3, i1, i2, i3, i4;  r1 = input.toCharArray()[**0**] - '0' == **1**;  r2 = input.toCharArray()[**1**] - '0' == **1**;  i1 = input.toCharArray()[**2**] - '0' == **1**;  r3 = input.toCharArray()[**3**] - '0' == **1**;  i2 = input.toCharArray()[**4**] - '0' == **1**;  i3 = input.toCharArray()[**5**] - '0' == **1**;  i4 = input.toCharArray()[**6**] - '0' == **1**;  **boolean** s1, s2, s3;  s1 = r1 ^ i1 ^ i2 ^ i4;  s2 = r2 ^ i1 ^ i3 ^ i4;  s3 = r3 ^ i2 ^ i3 ^ i4;  **if**(s1 && s2 && s3)  {  System.out.print(input.toCharArray()[**2**]);  System.out.print(input.toCharArray()[**4**]);  System.out.print(input.toCharArray()[**5**]);  System.out.println(input.toCharArray()[**6**] == '0' ? '1' : '0');  System.out.println("Errors in i4 or 7th byte");  }  **else** **if**(s1 && s2)  {  System.out.print(input.toCharArray()[**2**] == '0' ? '1' : '0');  System.out.print(input.toCharArray()[**4**]);  System.out.print(input.toCharArray()[**5**]);  System.out.println(input.toCharArray()[**6**]);  System.out.println("Errors in i1 or 3th byte");  }  **else** **if**(s2 && s3)  {  System.out.print(input.toCharArray()[**2**]);  System.out.print(input.toCharArray()[**4**]);  System.out.print(input.toCharArray()[**5**] == '0' ? '1' : '0');  System.out.println(input.toCharArray()[**6**]);  System.out.println("Errors in i3 or 6th byte");  }  **else** **if**(s1 && s3)  {  System.out.print(input.toCharArray()[**2**]);  System.out.print(input.toCharArray()[**4**] == '0' ? '1' : '0');  System.out.print(input.toCharArray()[**5**]);  System.out.println(input.toCharArray()[**6**]);  System.out.println("Errors in i2 or 5th byte");  }  **else** **if**(s1)  {  System.out.print(input.toCharArray()[**2**]);  System.out.print(input.toCharArray()[**4**]);  System.out.print(input.toCharArray()[**5**]);  System.out.println(input.toCharArray()[**6**]);  System.out.println("Errors in r1 or 1st byte");  }  **else** **if**(s2)  {  System.out.print(input.toCharArray()[**2**]);  System.out.print(input.toCharArray()[**4**]);  System.out.print(input.toCharArray()[**5**]);  System.out.println(input.toCharArray()[**6**]);  System.out.println("Errors in r2 or 2nd byte");  }  **else** **if**(s3)  {  System.out.print(input.toCharArray()[**2**]);  System.out.print(input.toCharArray()[**4**]);  System.out.print(input.toCharArray()[**5**]);  System.out.println(input.toCharArray()[**6**]);  System.out.println("Errors in r3 or 4th byte");  }  **else**  {  System.out.print(input.toCharArray()[**2**]);  System.out.print(input.toCharArray()[**4**]);  System.out.print(input.toCharArray()[**5**]);  System.out.println(input.toCharArray()[**6**]);  System.out.println("Errors not found");  }  }  } |

# Заключение

В ходе данной лабораторной работы, я познакомился с кодом Хэмминга, научился декодировать его и понимать в каком конкретно месте произошла ошибка при передаче. Также я научился определять минимальное число контрольных разрядов и высчитывать коэффициент избыточности. А еще я научился находить ошибки используя язык программирования Java

# Список использованных источников

1. Методические пособия по информатике Балакшин П.В., Соснин В.В., Машина Е.А. Информатика. – СПб: Университет ИТМО, 2020. – 122 с.

<https://isu.ifmo.ru/pls/apex/f?p=2143:0:107301084256427:DWNLD_F:NO::FILE:AFCE39920F4F4287E79C1554D056FE9B>

1. The Art of Doing Science and Engineering Richard W.Hamming U.S. Naval Postgraduate School Monterey, California 2005

<http://worrydream.com/refs/Hamming-TheArtOfDoingScienceAndEngineering.pdf>