Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 «Программная инженерия»

Системное и прикладное программное обеспечение

**Отчёт**

**По лабораторной работе №1**

**Помехоустойчивые коды**

**Вариант: 34**

Работу выполнил:

Поленов Кирилл Александрович

Группа Р3113

Работу принял:

Рыбаков Степан Дмитриевич

Оглавление

[Задание 3](#_Toc147777126)

[Основные этапы вычисления 6](#_Toc147777127)

[Заключение 19](#_Toc147777128)

[Список литературы 20](#_Toc147777129)

# Задание

1. Определить свой вариант задания с помощью номера в ISU (он же номер

студенческого билета). Вариантом является комбинация 3-й и 5-й цифр.

Т.е. если номер в ISU = 123456, то вариант = 35.

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

сообщений в виде последовательности 7-символьного кода.

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

которую представить в отчёте в виде изображения.

4. 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.

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

виде последовательности 11-символьного кода.

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

которую представить в отчёте в виде изображения.

7. 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.

8. 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.

9. 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

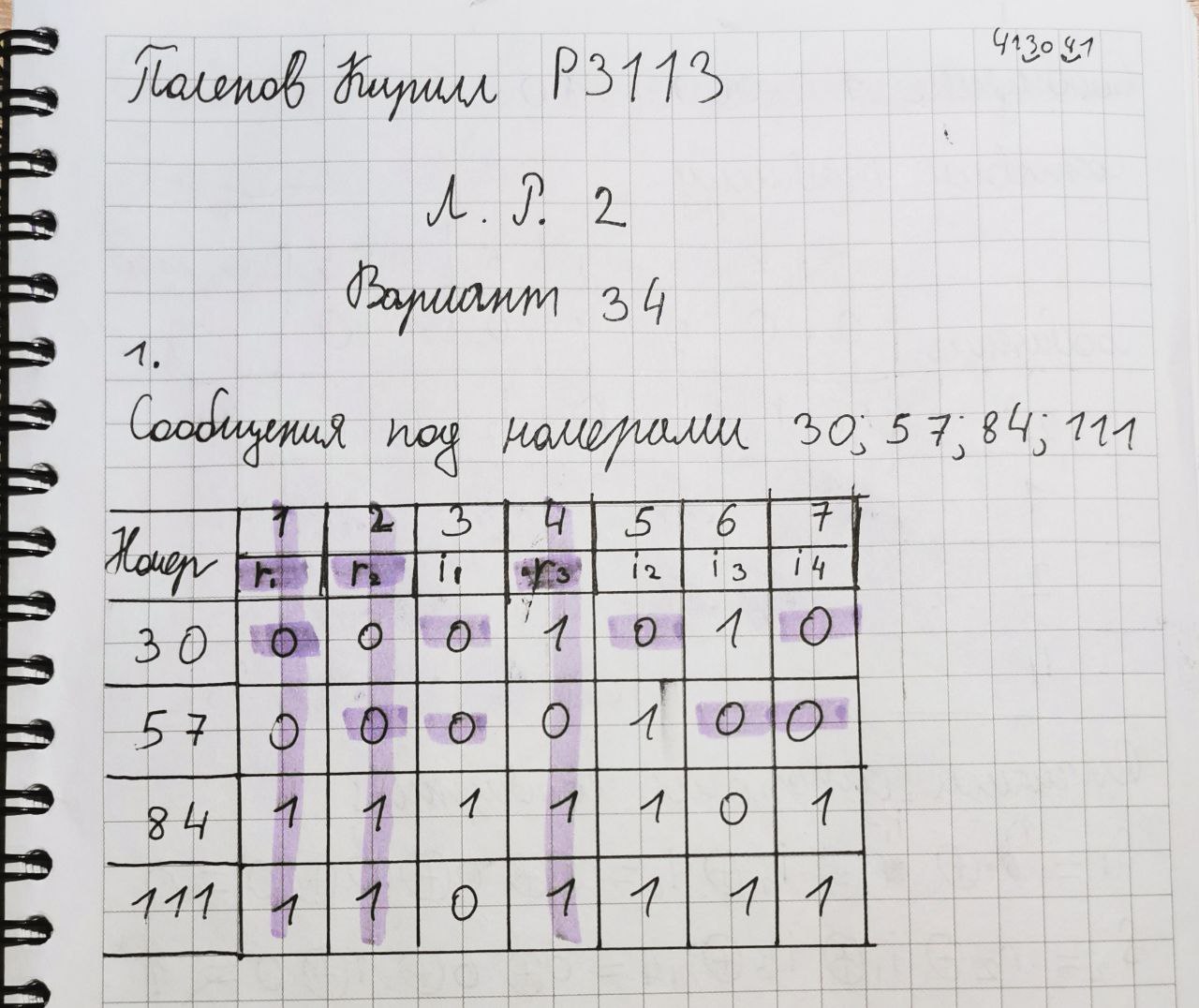
на основе классического кода Хэмминга (7,4), а затем выдает правильное

сообщение (только информационные биты) и указывает бит с ошибкой при

его наличии.

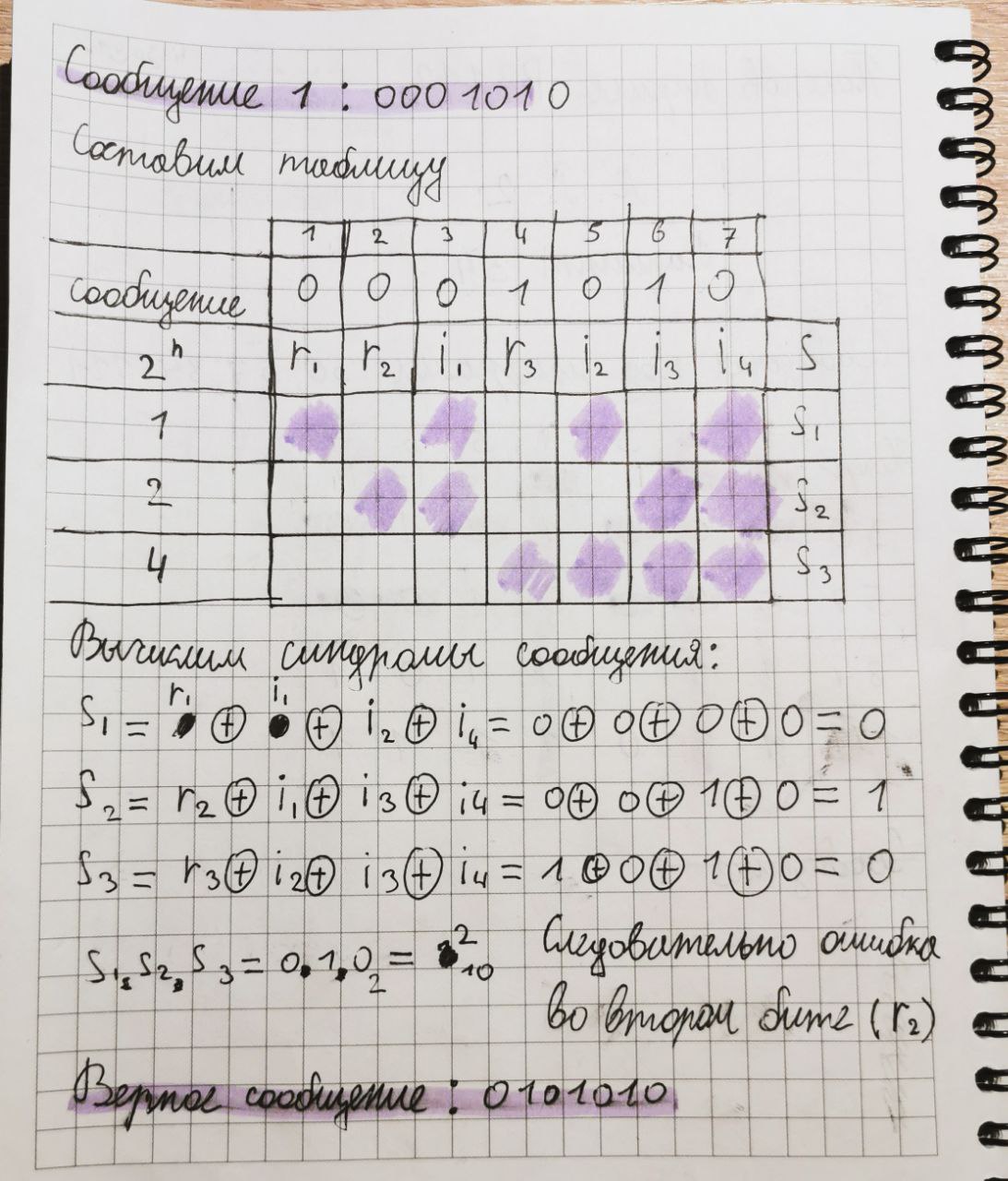
# Основные этапы вычисления

\*\*\* некоторые синдромы не соответствуют тем битам, в которых присутствует ошибка. В коде программы это исправлено, однако в тестах и фотографиях тетради эти ошибки всё ещё присутствуют.



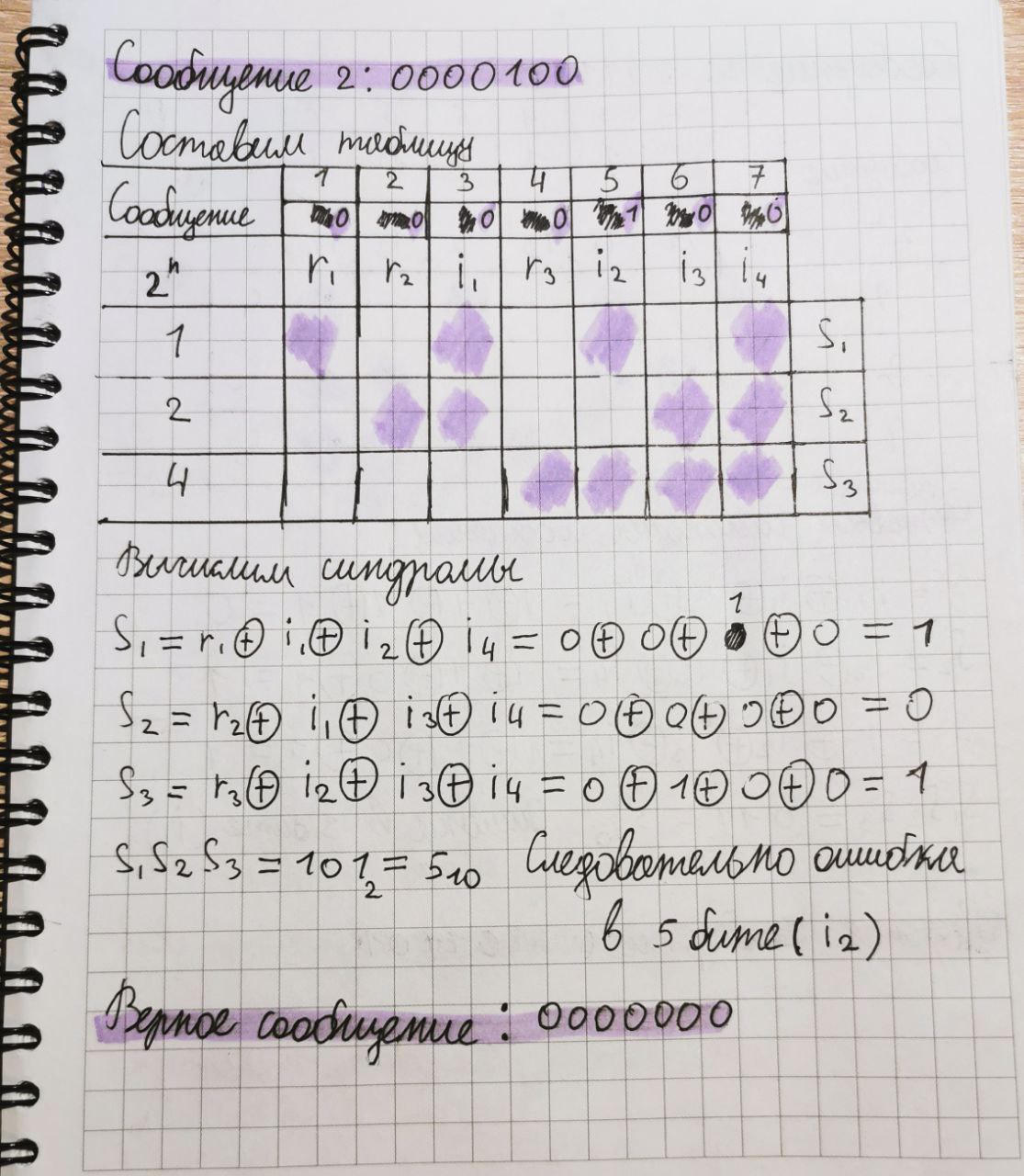
Задание 1

Сообщение 1:



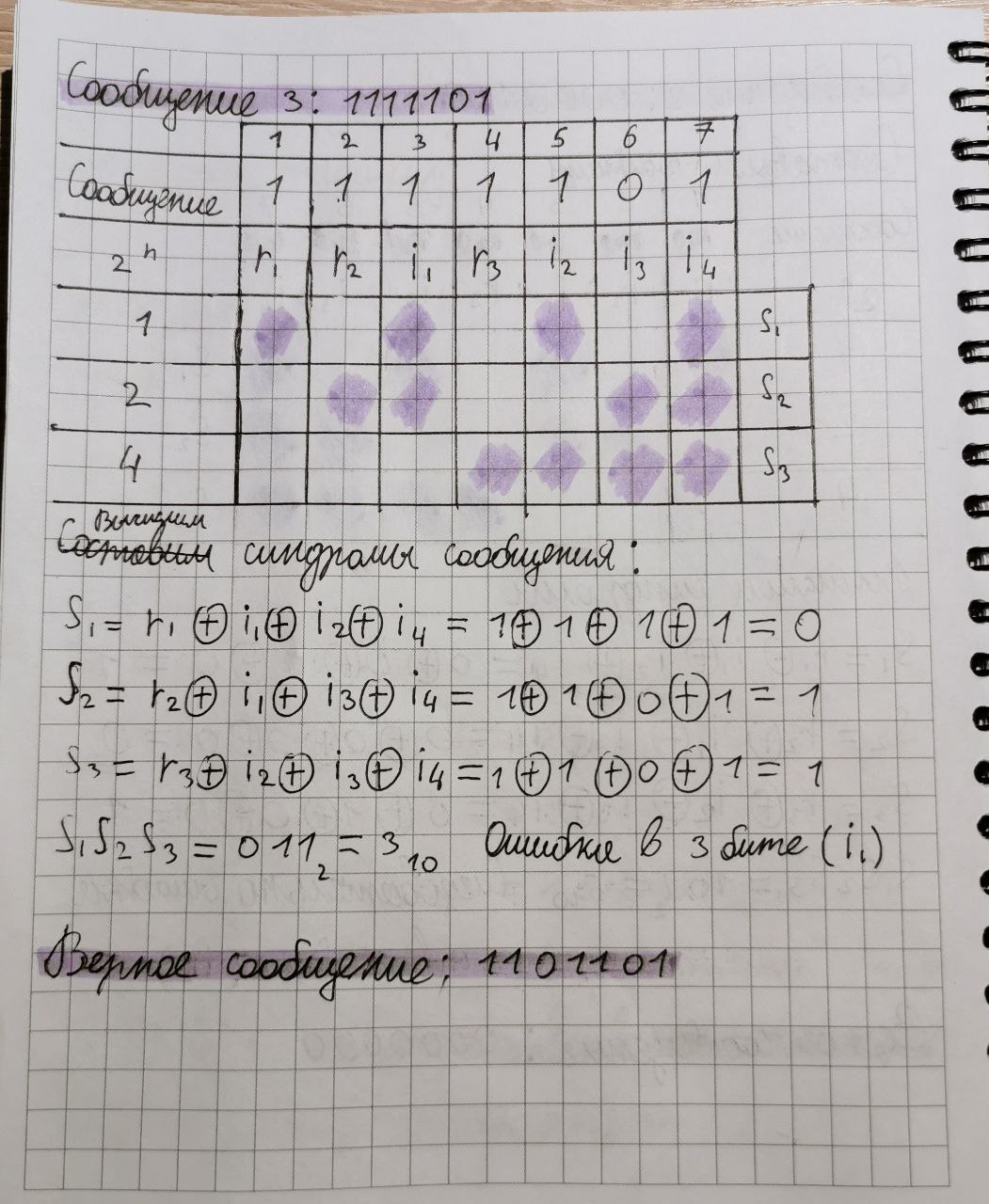
*Рисунок 1*

Сообщение 2:

**

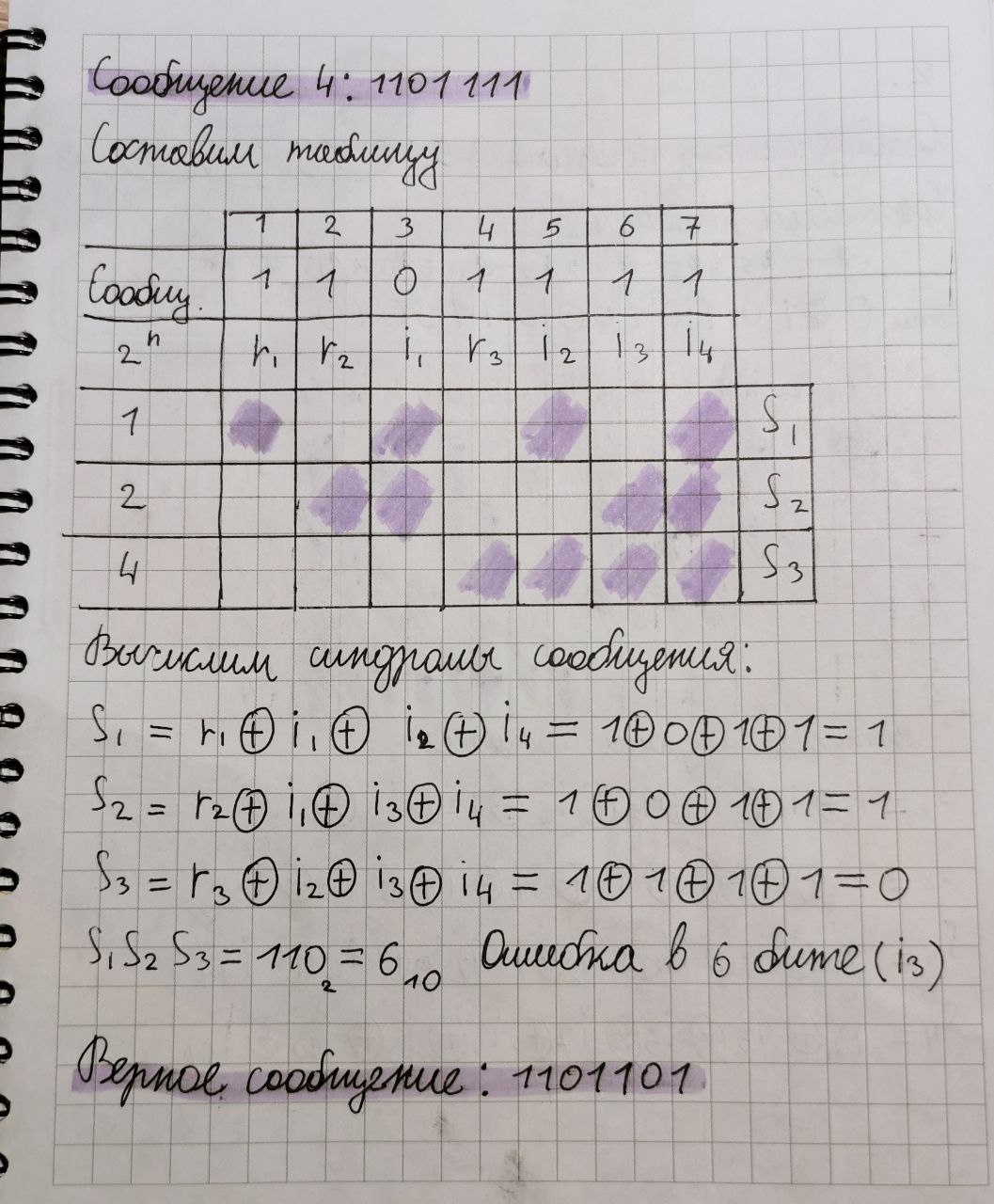
*Рисунок 2*

Сообщение 3:



*Рисунок 3*

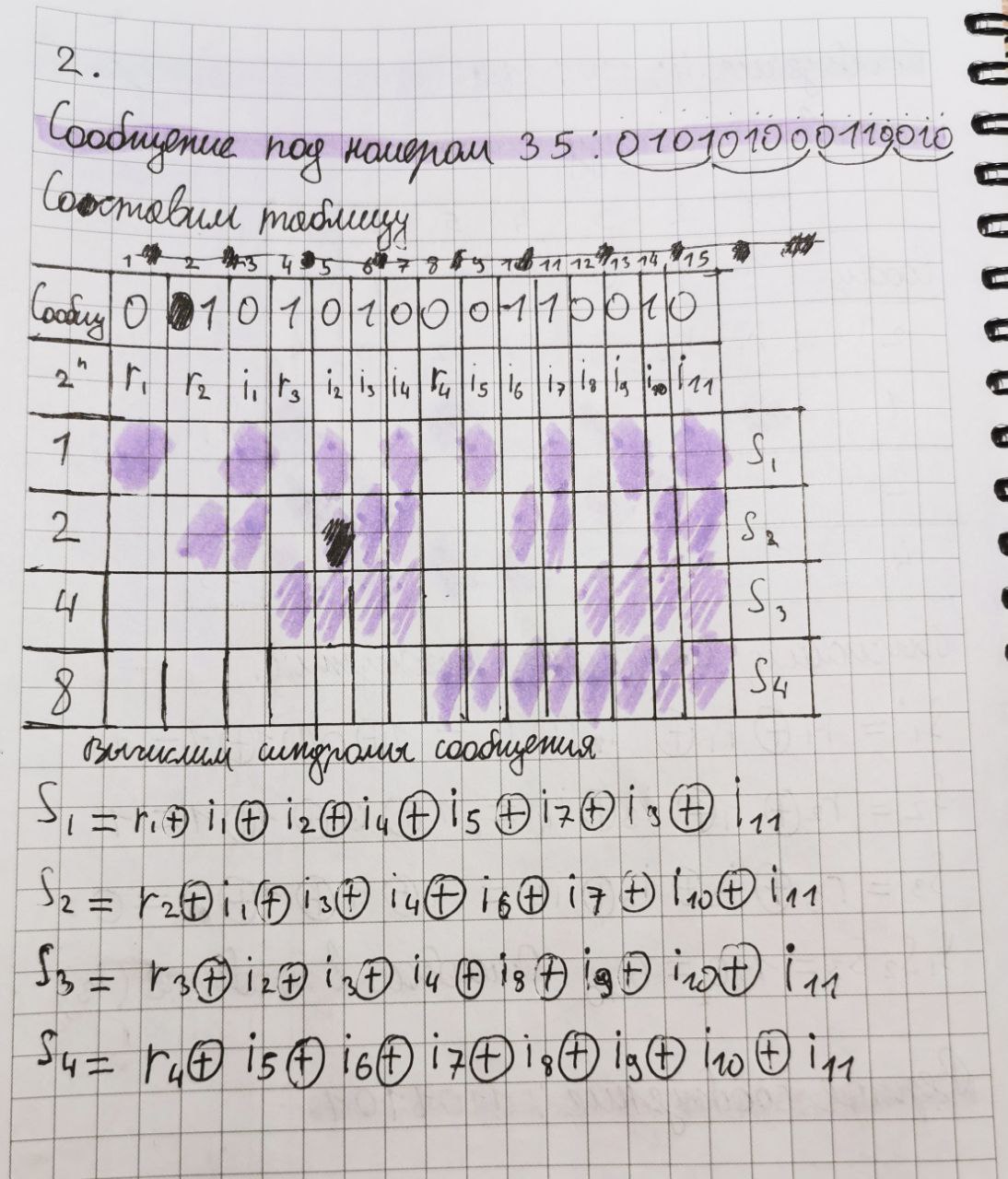
Сообщение 4:



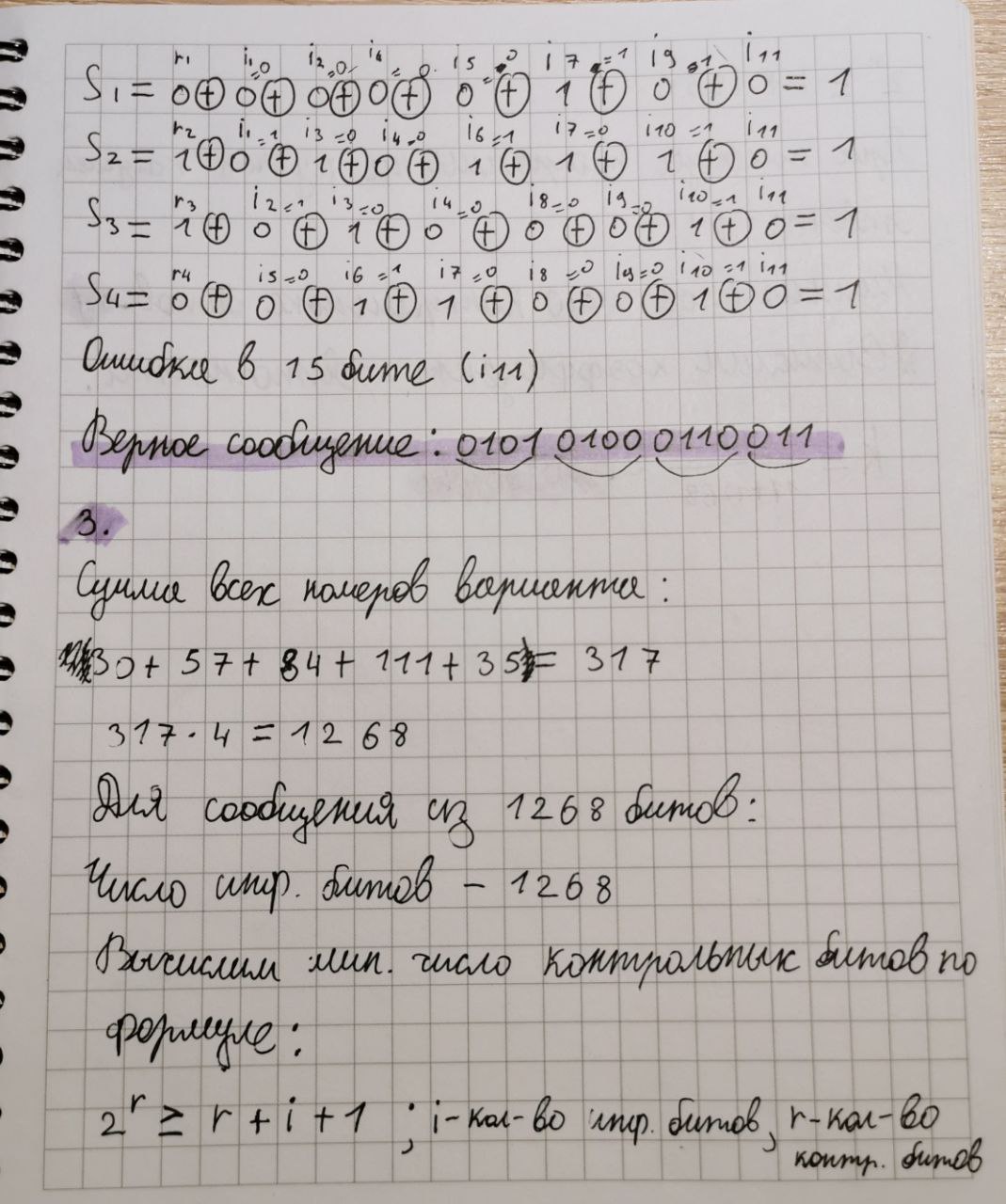
*Рисунок 4*

Задание 2

Сообщение 1:

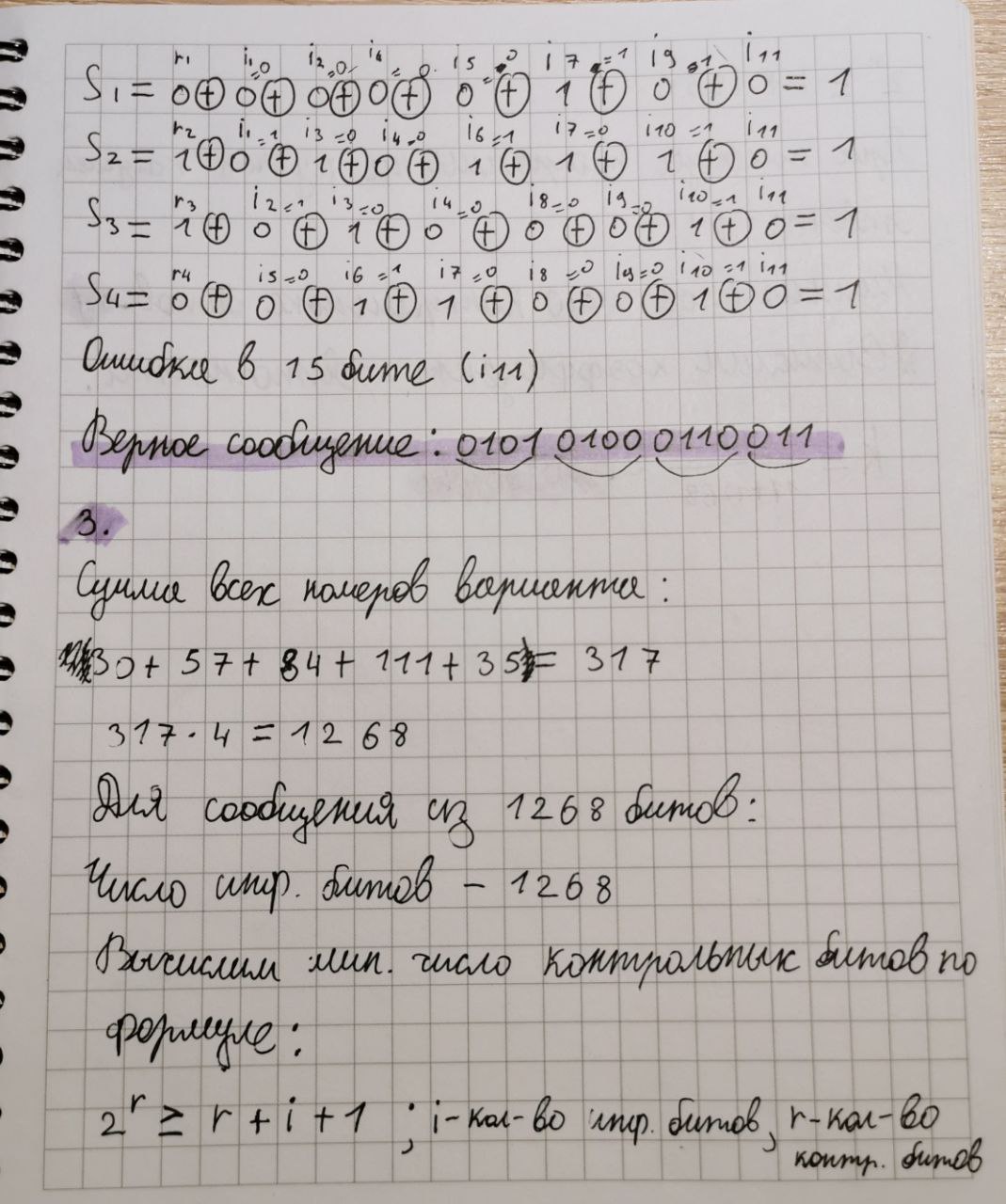


*Рисунок 5*

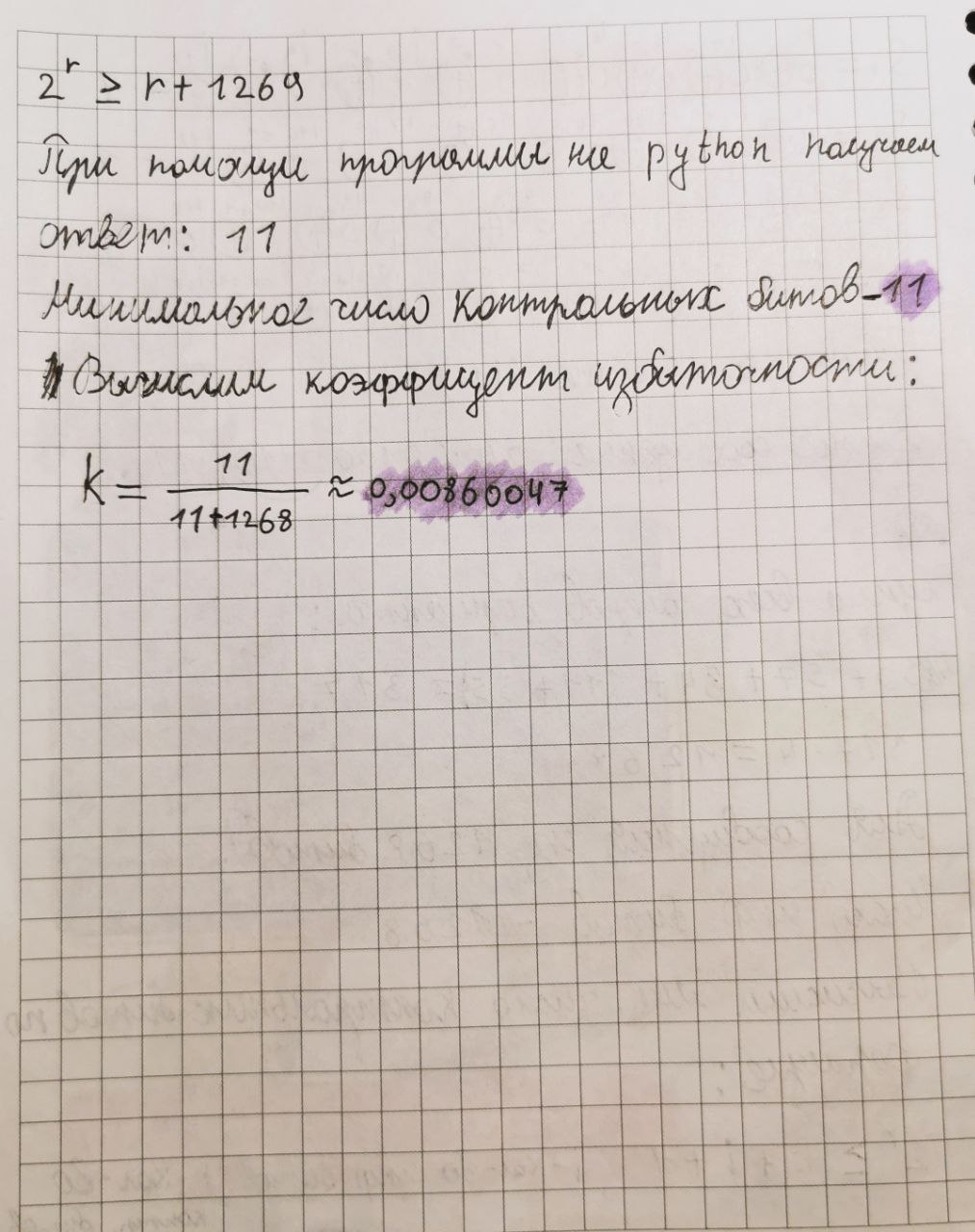


*Рисунок 6*

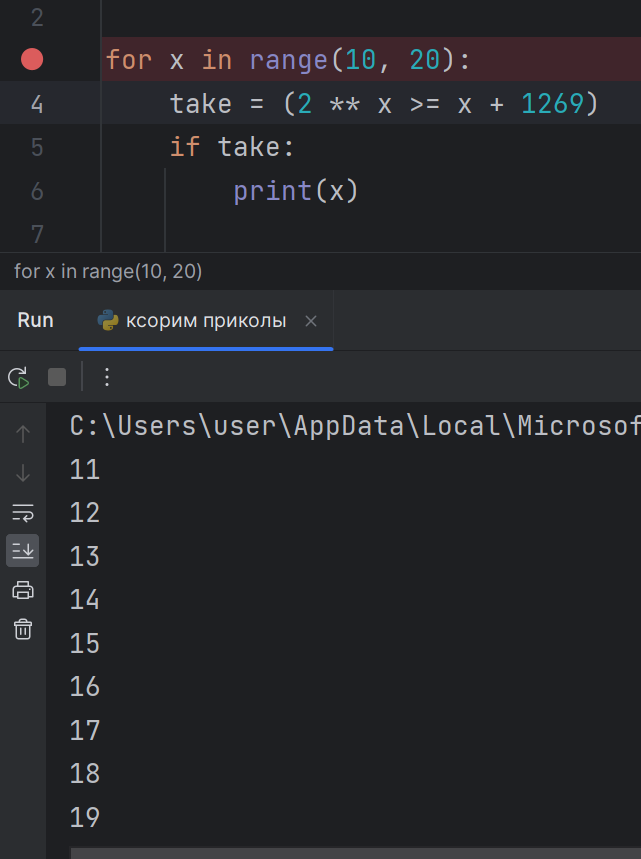
Задание 3



*Рисунок 7*

**

*Рисунок 8*

**

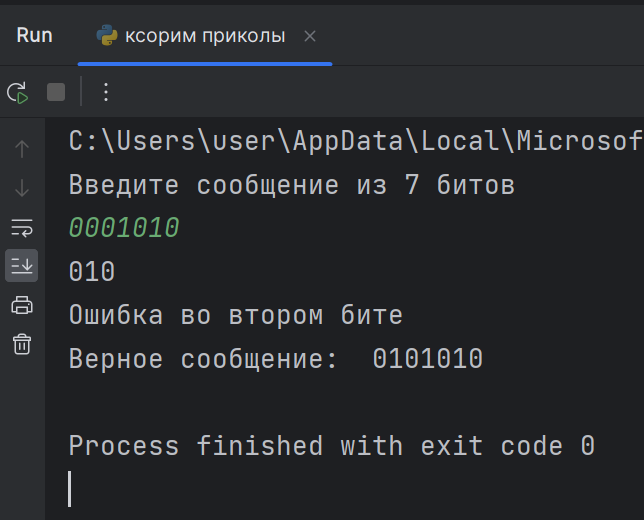
*Рисунок 9*

Дополнительное задание

Код программы на Python:

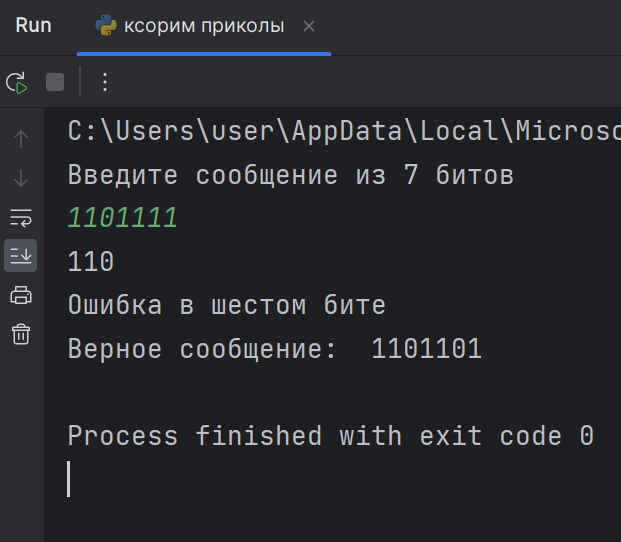
print('Введите сообщение из 7 битов')  
  
message = input()  
  
if len(message) != 7:  
 print('Броу, давай без приколов')  
 exit(1)  
  
s1 = int(message[0]) ^ int(message[2]) ^ int(message[4]) ^ int(message[6])  
s2 = int(message[1]) ^ int(message[2]) ^ int(message[5]) ^ int(message[6])  
s3 = int(message[3]) ^ int(message[4]) ^ int(message[5]) ^ int(message[6])  
  
syndrome = str(s1) + str(s2) + str(s3)  
print(syndrome)  
  
match syndrome:  
 case '000':  
 print('Ошибок нет. Всё кул ^\_^')  
  
 case '001':  
 print('Ошибка в четвертом бите')  
 if message[3] == '0':  
 message = message[:3] + '1' + message[4:]  
 else:  
 message = message[:3] + '0' + message[4:]  
  
 print('Верное сообщение: ', message)  
  
 case '010':  
 print('Ошибка во втором бите')  
 if message[1] == '0':  
 message = message[1] + '1' + message[2:]  
 else:  
 message = message[1] + '0' + message[2:]  
  
 print('Верное сообщение: ', message)  
  
 case '011':  
 print('Ошибка в шестом бите')  
 if message[5] == '0':  
 message = message[:5] + '1' + message[6:]  
 else:  
 message = message[:5] + '0' + message[6:]  
  
 print('Верное сообщение: ', message)  
  
 case '100':  
 print('Ошибка в первом бите')  
 if message[0] == '0':  
 message = '1' + message[2:]  
 else:  
 message = '0' + message[2:]  
  
 print('Верное сообщение: ', message)  
  
 case '101':  
 print('Ошибка в пятом бите')  
 if message[4] == '0':  
 message = message[:4] + '1' + message[5:]  
 else:  
 message = message[:4] + '0' + message[5:]  
  
 print('Верное сообщение: ', message)  
  
 case '110':  
 print('Ошибка в третьем бите')  
 if message[2] == '0':  
 message = message[:2] + '1' + message[3:]  
 else:  
 message = message[:2] + '0' + message[3:]  
  
 print('Верное сообщение: ', message)  
  
 case '111':  
 print('Ошибка в седьмом бите')  
 if message[1] == '0':  
 message = message[:7] + '1'  
 else:  
 message = message[:7] + '0'  
  
 print('Верное сообщение: ', message)

Тест №1



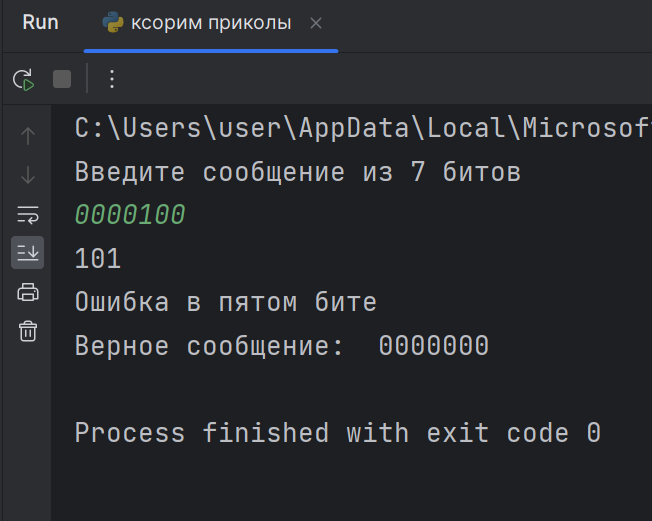
*Рисунок 10*

Тест №2



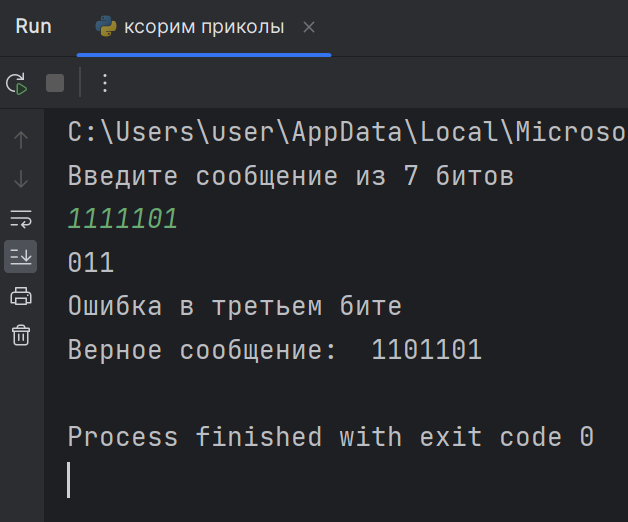
*Рисунок 11*

Тест №3



*Рисунок 12*

Тест №4



*Рисунок 13*

# Заключение

В ходе выполнения данной лабораторной работы я познакомился с кодом Хемминга. Узнал принцип его работы, научился составлять таблицы Хемминга для проверки сообщения на ошибки, научился вычислять синдромы сообщения и при помощи них находить ошибки в полученном сообщении, а также закрепил навыки программирования на языке Python.

# Список литературы

**AGalilov (название YouTube канала),** Код Хэмминга. Самоконтролирующийся и самокорректирующийся код. – URL: <https://youtu.be/QsBYshN5idw?si=iddXwSZEYuyY0KgW> (Дата обращения: 07.10.2023)

**П.В. Балакшин, В.В. Соснин, И.В. Калинин, Т.А. Малышева, С.В. Раков, Н.Г. Рущенко, А.М. Дергачев** Информатика: лабораторные работы и тесты [Электронный ресурс] – https://t.me/balakshin\_students (Дата обращения: 09.10.2023)