ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

*Московский институт электроники и математики им. А.Н.Тихонова*

**ОТЧЕТ**

**К КУРСОВОЙ РАБОТЕ**

**“Программный модуль для реализации различных схемы модуляции и кодирования в ns-3”**

Проверил:

Ст. преп. Ролич А.Ю.

Выполнили:

Студенты группы БИВ231

Суслова Анастасия Алексеевна

Швыркина Алина Юрьевна

Москва 2024 г

Оглавление

[Обоснование темы 3](#_Toc168575070)

[Техническое задание 4](#_Toc168575071)

[Описание продукта 4](#_Toc168575072)

[Требования к интерфейсу 5](#_Toc168575073)

[Архитектура системы 5](#_Toc168575074)

[Требования к документации 7](#_Toc168575075)

[Документация пользователя 8](#_Toc168575076)

[Назначение программы 8](#_Toc168575077)

[Инструкция по использованию 8](#_Toc168575078)

[Системные требования 11](#_Toc168575079)

[Документация разработчика 12](#_Toc168575080)

[Подробное описание функциональности 12](#_Toc168575081)

[Системные требования 14](#_Toc168575082)

[Описание алгоритма решения задачи 16](#_Toc168575083)

[Список использованных источников 22](#_Toc168575084)

[*Приложение 1* 23](#_Toc168575085)

[*Приложение 2* 28](#_Toc168575086)

# Обоснование темы

Программа, которую мы разработали, предназначена для расчета границы чувствительности приемника. Граница чувствительности определяет минимальный уровень сигнала, который приемник может обнаружить и правильно интерпретировать. Для этого программа учитывает параметры канала связи, такие как количество подканалов связи, количество ресурсных блоков в каждом подканале связи, используемая схема модуляции кодирования, нумерология, уровень шума и др.

Мы используем специальные алгоритмы и моделирование, чтобы точно определить величину чувствительности приемника при разных конфигурациях канала связи. После введения необходимых параметров программа проводит расчет и выводит достоверную границу чувствительности приемника.

Эта программа будет являться важным инструментом для инженеров и разработчиков в области радиосвязи, радиолокации, беспроводных сетей и других областей, где важно определить возможности приемника в заданных условиях. Благодаря точному расчету границы чувствительности приемника, можно эффективно планировать и оптимизировать работу радиосистем.

# Техническое задание

## Описание продукта

Функциональные требования:

* Основной функцией программного обеспечения является процедура расчета чувствительности приемника на базе конфигурационного файла, содержащего необходимую для расчета информацию о каналах связи.
* Программное обеспечение должно обеспечивать расчет метрики и формирование выходных файлов, в которых указываются значения рассчитанной метрики и значения параметров канала связи.
* Программное обеспечение должно обеспечивать визуализацию зависимостей чувствительности приемника от параметров канала связи.

Описание функции:

* Получение входных данных от пользователя и их проверка
* Чтение расчетных таблиц для введенного пользователем типа канала
* Расчет границы чувствительности приемника по формуле
* Вывод результата на консоль и в файл

Входные данные:

Входных данных будет 2 типа. Первый — это числовые данные, которые пользователь введет через консоль. Границы входных данных будут написаны непосредственно перед вводом, т.к. для каждого канала они разные.  А второй – файл, в котором будут записаны числовые значения необходимые для вычисления границы чувствительности источника

Выходные данные:

Выходные данные будут представлять собой список конфигураций каналов и чувствительностей их приемников, записанные в файл, и график зависимости нужной пользователю величины от чувствительности приемника. Числовые данные будут записаны в файл и, по желанию пользователя, выведены на экран. График можно будет увидеть, открыв и запустив скрипт из пайтона.

Нефункциональные требования:

* Время отклика на запрос пользователя не должно превышать 6 секунд.
* Программное обеспечение должно иметь алгоритм обработки ошибок и некорректных данных
* Требования к безопасности не предъявляются

## Требования к интерфейсу

Графический интерфейс отсутствует. Для получения результата пользователю необходимо ввести входные данные через консоль.

## Архитектура системы

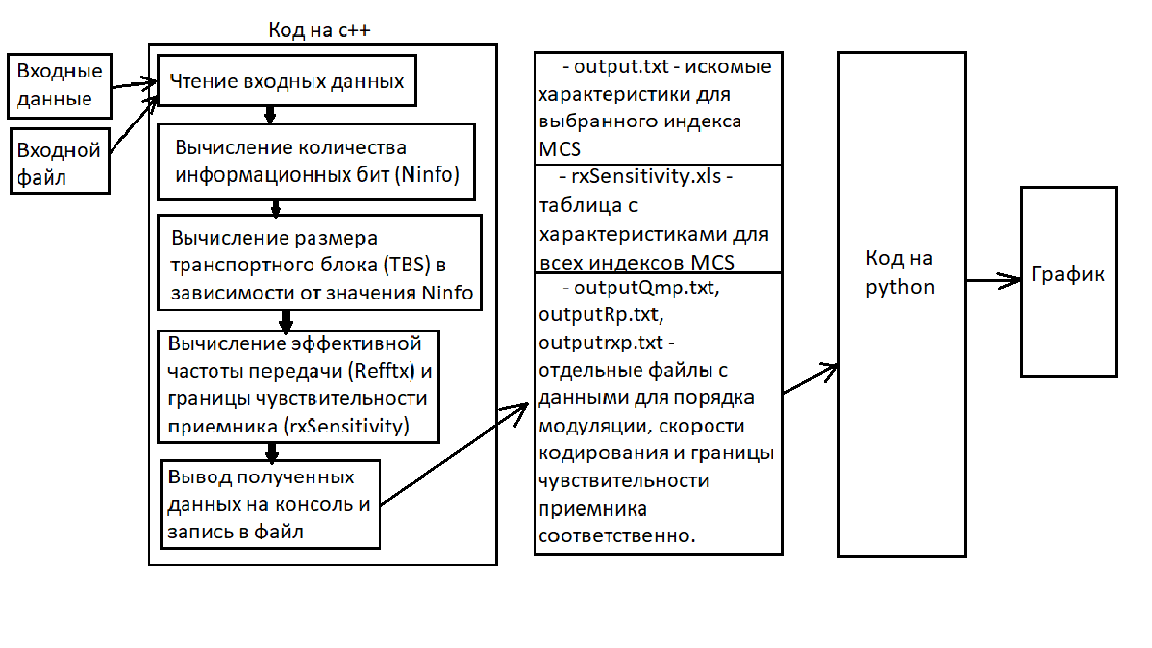


Рис. 1. Архитектура системыТестирование

1. Типы тестирования:

* Модульное тестирование: Проверка отдельных модулей (функций, классов) на корректность их работы.
* Интеграционное тестирование: Проверка взаимодействия между разными модулями.
* Системное тестирование: Проверка всей системы в её целостности.
* Приемочное тестирование: Тестирование системы с точки зрения конечного пользователя.

1. Автоматизированное тестирование: не требуется.
2. Тестирование с использованием данных: Тестирование формы ввода данных с использованием различных комбинаций значений.
3. Тестирование производительности: не требуется
4. Тестирование совместимости:не требуется.
5. Тестирование безопасности: не требуется.
6. Тестирование восстановления:не требуется.
7. Тестирование масштабирования: не требуется.
8. Тестирование обновлений и изменений:тестирование совместимости изменений в коде.

## Требования к документации

Требования к документированию кода:

* Каждая функция, метод и сложный участок кода должен содержать подробные комментарии, поясняющие его функциональность, входные и выходные данные.
* Необходимо использовать комментарии для описания основных этапов алгоритмов и логики, особенно если они не очевидны.
* Каждая функция должна иметь описания параметров, возвращаемого значения и возможных исключений.
* Нужно разделять код на логические блоки с использованием комментариев или меток.
* Файлы кода должны быть структурированы и названы ясно, чтобы быстро определить их содержимое.
* Использование четкого и понятного стиля написания на русском языке

Требования к пользовательской документации:

* Необходимо предоставить подробные инструкции по установке и настройке программного обеспечения.
* Нужно создать подробное руководство, описывающее основные функции программы.

# Документация пользователя

Назначение программы

Данная программа предназначена для расчета границы чувствительности приемника в сети 5G NR. Программа позволяет выбрать таблицу кодирования, ввести необходимые параметры и получить следующие выходные данные:

- Размер транспортного блока (TBS)

- Эффективная частота передачи

- Граница чувствительности приемника

## Инструкция по использованию

1. Зайдите в проводник и создайте в нем папку для загрузки программного модуля для вычисления границы чувствительности приемника. Затем откройте командную строку из этой папки (это можно сделать с помощью написания команды *cmd* в адресной строке) и с помощью команды *git clone https://github.com/Anastasija666/kursovaya.git* скачайте папку с программным модулем из гитхаба. В результате работы команды появится папка kyrsovaya.
2. Запустите программу *main.cbp* из папки *kyrsovaya*, предварительно убедившись в наличии файлов «*Table 5.1.3.1-1 MCS index table 1 for PDSCH.txt*», «*Table 5.1.3.1-2 MCS index table 2 for PDSCH.txt*», «*Table 5.1.3.1-3 MCS index table 3 for PDSCH.txt*» и «*TBS\_for\_Ninfo\_less\_then\_3824.txt*» в данной папке.

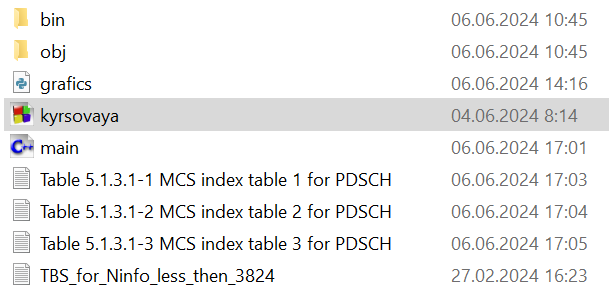
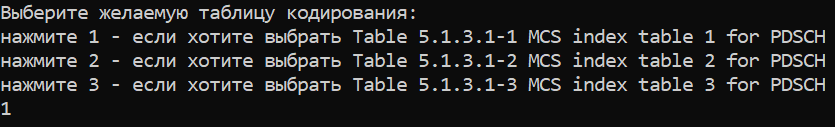


Рис.2. Папка kyrsovaya

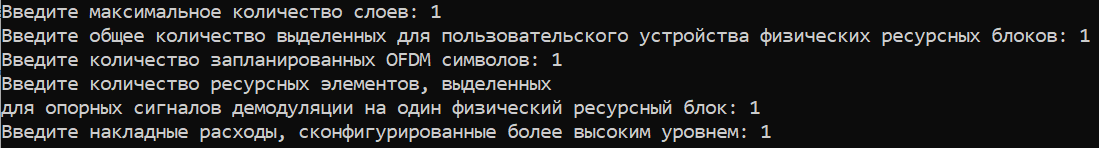
1. Затем выберите желаемую таблицу кодирования введя соответствующий таблице номер. Таблицы (*Table 5.1.3.1-1, Table 5.1.3.1-2* и *Table 5.1.3.1-3*) были взяты с сайта RFwireless-world.com[[1]](#footnote-2).

Рис.3. Выбор желаемой таблицы кодирования



1. После введите положительные значения максимального количества слоев, количества выделенных физических ресурсных блоков, количества запланированных OFDM символов, количества ресурсных элементов, выделенных для опорных сигналов демодуляции, и накладных расходов.

Рис. 4. Демонстрация ввода данных



1. Программа выведет в консоль характеристики для каждого индекса MCS: порядок модуляции, кодовую скорость, размер транспортных блоков, эффективную частоту передачи, и, главное, границу чувствительности приемника.

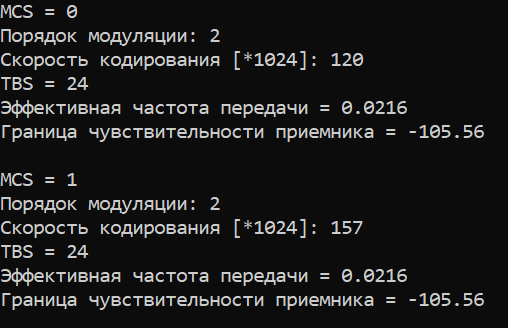


Рис. 5.1. Пример вывода данных в консоль

Также программа создаст файл *rxSensitivity.xls* в папке *kyrsovaya* в котором эти данные будут записаны в формате таблицы и, в той же папке, файл *output.txt* в котором данные будут записаны в том же формате что и в консоли.

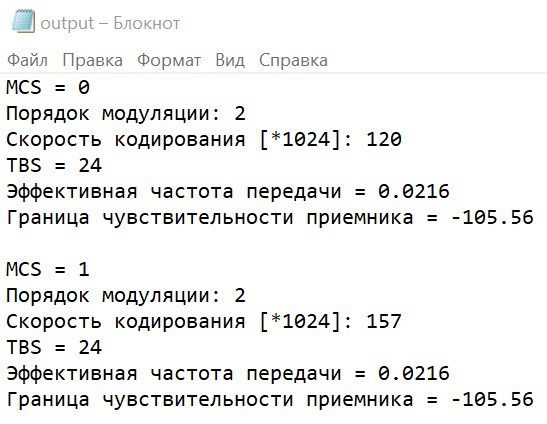
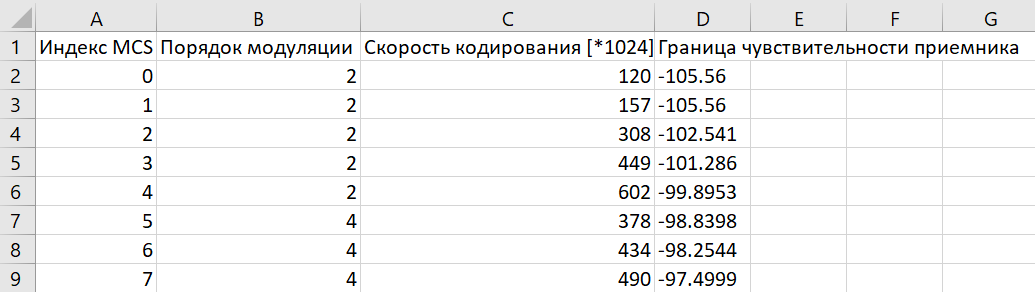


Рис. 5.2. Пример вывода данных в файлы

Для того, чтобы зайти в файл *rxSensitivity.xls* нажмите «Да» на всплывающем окне (рис 5.3).

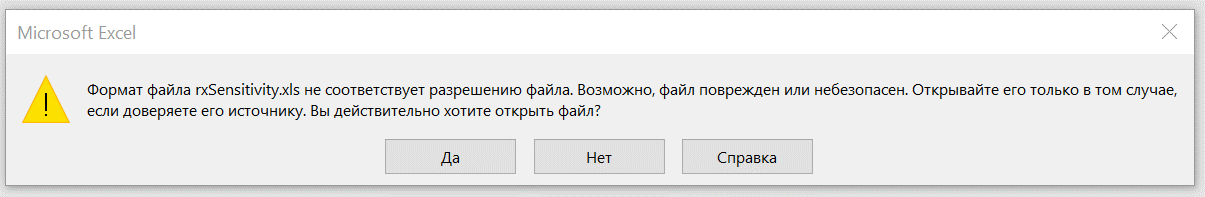


Рис. 5.3. Окно появляющееся при открытии файла *rxSensitivity.xls*

1. Затем запустите программу *grafics.py* из папки *kyrsovaya*. Убедитесь, что у вас установлены необходимые библиотеки: pandas, matplotlib.

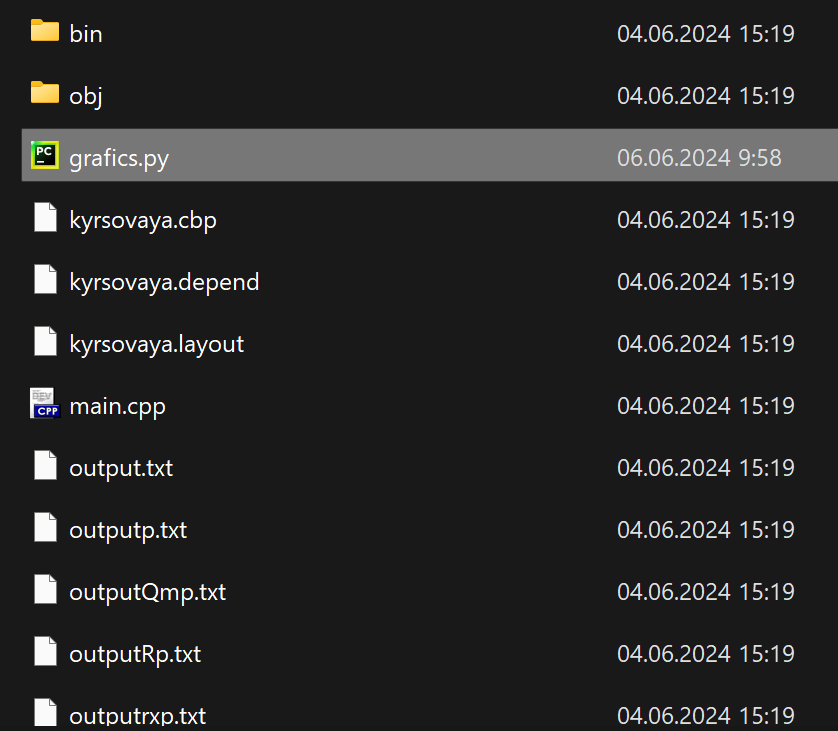


Рис. 6. Программа grafics.py в папке kyrsovaya

1. Запустите скрипт.
2. Программа выведет список доступных графиков:

* Зависимость чувствительности приёмника от скорости
* Зависимость чувствительности приёмника от значения модуляции

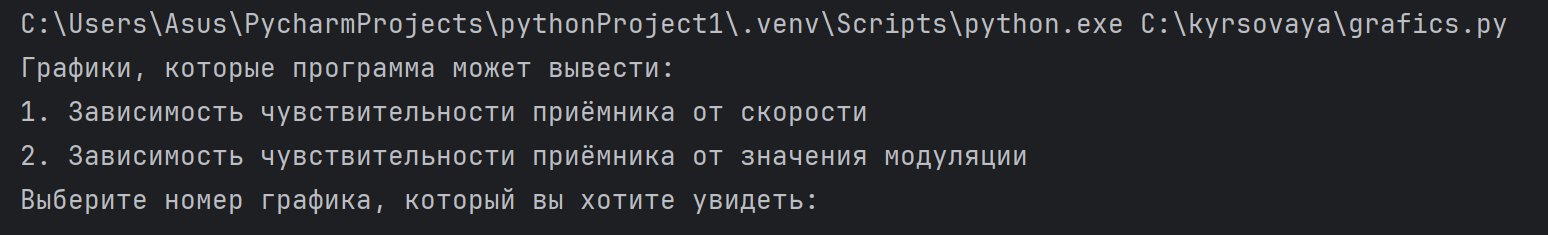


Рис. 7. Выбор графика

1. Введите номер графика, который вы хотите увидеть (1 или 2).
2. Программа построит и сохранит выбранный график в текущей директории под именем *graf1.png* или *graf2.png*.

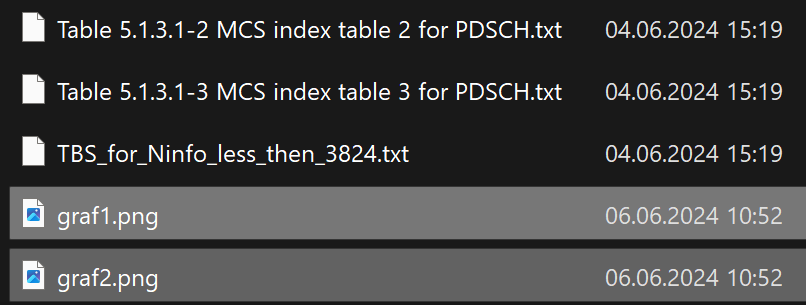


Рис. 8. Сохранение графиков в папке kyrsovaya

## Системные требования

* Компилятор C++ (например, GCC, Clang или Visual C++)
* Наличие текстовых файлов с таблицами кодирования («*Table 5.1.3.1-1 MCS index table 1 for PDSCH.txt*», «*Table 5.1.3.1-2 MCS index table 2 for PDSCH.txt*», «*Table 5.1.3.1-3 MCS index table 3 for PDSCH.txt», «TBS\_for\_Ninfo\_less\_then\_3824.txt*») в одной папке с исполняемым файлом программы.
* Python: Версия Python 3.6 или выше. Код использует современные библиотеки Python, которые могут не работать с более ранними версиями.
* Библиотеки Python:
* pandas: Версия 1.0.0 или выше. Эта библиотека используется для чтения данных из CSV-файлов.
* matplotlib: Версия 3.2.0 или выше. Эта библиотека используется для построения и сохранения графиков.

**Примечания:**

* Для каждого нового запуска программы будут созданы новые файлы вывода.

# Документация разработчика

## Подробное описание функциональности

1. Выбор таблицы кодирования

* Пользователю предлагается выбрать одну из трех таблиц кодирования (*Table 5.1.3.1-1*, *Table 5.1.3.1-2* или *Table 5.1.3.1-3*) для дальнейших расчетов.
* Пользовательский ввод сохраняется в переменной table.
* В зависимости от выбора пользователя, программа будет считывать данные из соответствующего файла.

1. Ввод параметров

Программа запрашивает у пользователя ввод следующих параметров для дальнейшего использования в расчетах одновременно проверяя являются ли введенные числа положительными:

* Максимальное количество слоев (v)
* Общее количество выделенных для пользовательского устройства физических ресурсных блоков (n\_PRB)
* Количество запланированных OFDM символов (N\_sh\_symb)
* Количество ресурсных элементов, выделенных для опорных сигналов демодуляции на один физический ресурсный блок (Ndmrs)
* Накладные расходы, сконфигурированные более высоким уровнем (Noh)

1. Расчет характеристик

Программа перебирает все индексы MCS от 0 до 31 и для каждого индекса MCS:

* Считывает из соответствующего файла порядок модуляции (Qm) и скорость кодирования (R).
* Вычисляет количество информационных бит (Ninfo) с учетом количества ресурсных элементов (Nre), скорости кодирования и порядка модуляции.
* Вычисляет размер транспортного блока (TBS) в зависимости от значения Ninfo.
* Вычисляет эффективную частоту передачи (Refftx) и границу чувствительности приемника (rxSensitivity).

1. Вывод результатов

* Программа выводит на консоль порядок модуляции, кодовую скорость, размер транспортных блоков, эффективную частоту передачи, и границу чувствительности приемника для каждого индекса MCS.
* Также, результаты сохраняются в следующих файлах:
  + output.txt - основные характеристики для каждого индекса MCS, в том же формате что и в консоли
  + rxSensitivity.xls - таблица с данными границы чувствительности приемника, скоростью кодирования и порядком модуляции для всех индексов MCS
  + outputQmp.txt, outputRp.txt, outputrxp.txt - отдельные файлы с данными порядка модуляции, скорости кодирования и границы чувствительности приемника соответственно необходимые для дальнейшего построения графика.

1. Общее описание программы *grafics.py*:

* Программа предназначена для построения и сохранения двух графиков, отображающих зависимость чувствительности приёмника от различных параметров.
* Первый график показывает зависимость чувствительности приёмника от скорости.
* Второй график показывает зависимость чувствительности приёмника от значения модуляции.

2. Используемые библиотеки:

* *pandas*: используется для чтения данных из CSV-файлов.
* *matplotlib*: используется для построения и сохранения графиков.
* *os*: используется для получения пути к текущему каталогу.

3. Функции:

* graphs(x, y, title, xlabel, ylabel, filename):
* функция для построения и сохранения графика.
* принимает следующие аргументы:
* x: данные для оси X.
* y: данные для оси Y.
* title: заголовок графика.
* xlabel: подпись для оси X.
* ylabel: подпись для оси Y.
* filename: имя файла, в который будет сохранен график.
* Внутри функции происходит построение графика с заданными параметрами, получение пути к текущему каталогу и сохранение графика в этом каталоге.
* graf\_1():
* Функция для построения первого графика, отображающего зависимость чувствительности приёмника от скорости.
* Считывает данные из файлов outputRp.txt и outputrxp.txt, передает их в функцию graphs() для построения и сохранения графика.
* graf\_2():
* Функция для построения второго графика, отображающего зависимость чувствительности приёмника от значения модуляции.
* Считывает данные из файлов outputQmp.txt и outputrxp.txt, передает их в функцию graphs() для построения и сохранения графика.

4. Главный код:

* Выводит на экран список доступных графиков.
* Запрашивает у пользователя ввод номера желаемого графика.
* Вызывает соответствующую функцию (graf\_1() или graf\_2()) для построения и сохранения графика.
* Если пользователь вводит некорректный номер, выводит сообщение об ошибке.

5. Файлы данных:

* outputRp.txt: Файл, содержащий данные для построения первого графика.
* outputrxp.txt: Файл, содержащий данные для построения обоих графиков.
* outputQmp.txt: Файл, содержащий данные для построения второго графика.

6. Выходные данные:

* Программа сохраняет построенные графики в формате PNG в текущем каталоге.
* Файлы графиков называются graf1.png и graf2.png.

Системные требования

* Компилятор C++ (например, GCC, Clang или Visual C++)
* Git >=2.45.2
* Наличие текстовых файлов с таблицами кодирования («*Table 5.1.3.1-1 MCS index table 1 for PDSCH.txt*», «*Table 5.1.3.1-2 MCS index table 2 for PDSCH.txt*», «*Table 5.1.3.1-3 MCS index table 3 for PDSCH.txt», «TBS\_for\_Ninfo\_less\_then\_3824.txt*») в одной папке с исполняемым файлом программы.
* Python: Версия Python 3.6 или выше. Код использует современные библиотеки Python, которые могут не работать с более ранними версиями.
* Библиотеки Python:
* pandas: Версия 1.0.0 или выше. Эта библиотека используется для чтения данных из CSV-файлов.
* matplotlib: Версия 3.2.0 или выше. Эта библиотека используется для построения и сохранения графиков.

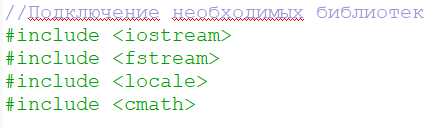
**Примечания**

* Программный модуль расположен в гитхабе в репозитории <https://github.com/Anastasija666/kursovaya.git.> Скачать его на свой компьютер можно при наличии git с помощью команды *git clone* [*https://github.com/Anastasija666/kursovaya.git*](https://github.com/Anastasija666/kursovaya.git)
* Программа использует встроенные функции C++, такие как setlocale(), ifstream, ofstream, pow(), floor(), ceil() и round().
* Программа предполагает, что все необходимые для ее работы файлы (*Table 5.1.3.1-1 MCS index table 1 for PDSCH.txt, Table 5.1.3.1-2 MCS index table 2 for PDSCH.txt, Table 5.1.3.1-3 MCS index table 3 for PDSCH.txt*) находятся в той же папке, что и исполняемый файл программы.
* Программа сохраняет результаты расчетов в файлы, которые будут перезаписываться при каждом новом запуске.

# Описание алгоритма решения задачи

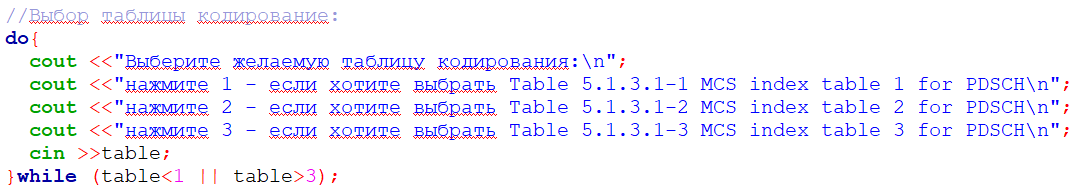
Алгоритм вычислительной часть программы, написанный на с++:

1. Подключение необходимых библиотек: iostream, fstream, locale, cmath.



1. Выбор таблицы кодирования:

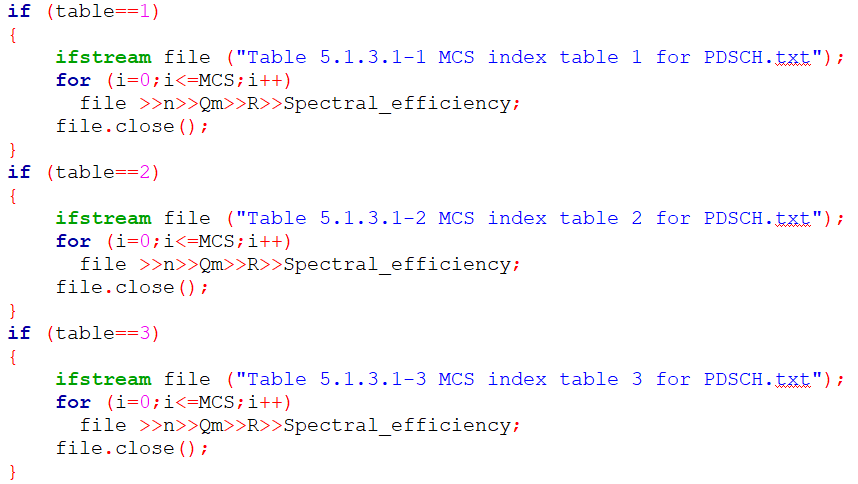
* Пользователю предлагается выбрать одну из трех таблиц кодирования для PDSCH.
* Проверяется корректность ввода (число от 1 до 3)



1. Ввод дополнительных данных (код можно посмотреть в приложении 1 под соответствующим комментарием):

* Максимальное количество слоев.
* Общее количество выделенных для пользовательского устройства физических ресурсных блоков.
* Количество запланированных OFDM-символов.
* Количество ресурсных элементов, выделенных для опорных сигналов демодуляции на один физический ресурсный блок.
* Накладные расходы, сконфигурированные более высоким уровнем.

1. Начало цикла для всех возможных значений индекса модуляции и кодирования (MCS от 0 до 31).
2. Считывание из файла значений порядка модуляции, кодовой скорости и спектральной эффективности для выбранной таблицы кодирования.



1. Вычисление количества информационных бит (Ninfo):
2. Вычисление количества ресурсных элементов, которые могут быть использованы для передачи данных(Nre2):

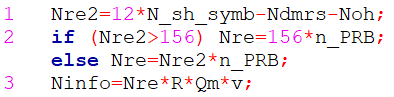
* Nre2 вычисляется как произведение количества ресурсных элементов на OFDM символ (12) и количества запланированных OFDM символов (N\_sh\_symb), за вычетом количества ресурсных элементов, выделенных для опорных сигналов демодуляции (Ndmrs) и накладных расходов, сконфигурированных более высоким уровнем (Noh).

1. Вычисление Nre:

* Если Nre2 больше максимального количества ресурсных элементов на физический ресурсный блок (156), то Nre принимает значение 156 \* общее количество выделенных физических ресурсных блоков (n\_PRB ).
* Если Nre2 меньше или равно 156, то Nre принимает значение Nre2 \* n\_PRB.

1. Вычисление количества информационных бит, которые могут быть переданы (Ninfo):

* Ninfo вычисляется как произведение Nre, R (скорость кодирования), Qm (порядок модуляции) и v (максимальное количество слоев).



1. Вычисление размера транспортных блоков (TBS):

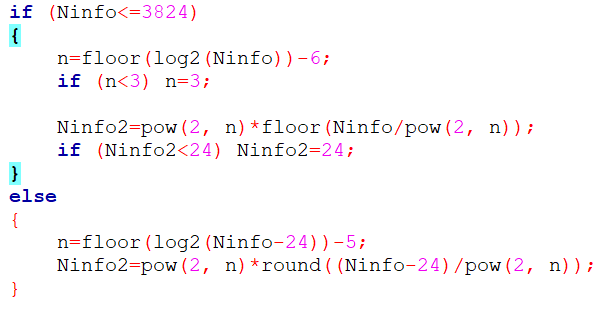
* Вычисление Ninfo2:

Если Ninfo меньше или равно 3824:

* Вычисляется n как целая часть от логарифма по основанию 2 от Ninfo минус 6. А если n меньше 3, то n принимает значение 3.
* Вычисляется Ninfo2 как произведение 2 в степени n и целой части от деления Ninfo на 2 в степени n. Но если Ninfo2 меньше 24, то Ninfo2 принимает значение 24.

Если Ninfo больше 3824:

* Вычисляется n как целая часть от логарифма по основанию 2 Ninfo-24 минус 5.
* Вычисляется Ninfo2 как произведение 2 в степени n и округленного до ближайшего целого значения частного от деления Ninfo-24 на 2 в степени n.



* Вычисление TBS:

Если скорость кодирования R меньше или равна 0.25:

* Вычисляется C как целое число, округленное вверх, от частного (Ninfo2+24)/3816.
* TBS вычисляется как 8 \* C \* округленное вверх значение (Ninfo2+24)/(8\*C) минус 24.

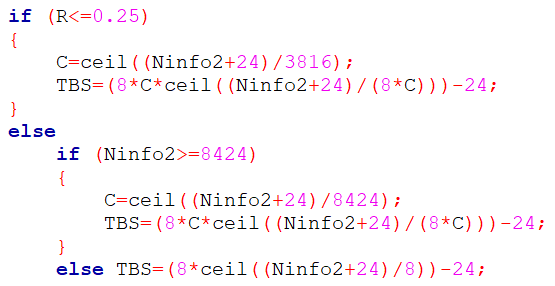
Если скорость кодирования R больше 0.25:

Если Ninfo2 больше или равно 8424:

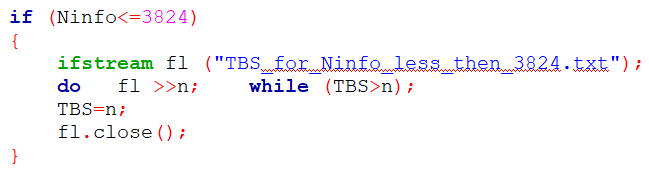
* Вычисляется C как целое число, округленное вверх, от частного (Ninfo2+24)/8424.
* TBS вычисляется как 8 \* C \* округленное вверх значение (Ninfo2+24)/(8\*C) минус 24.

Если Ninfo2 меньше 8424:

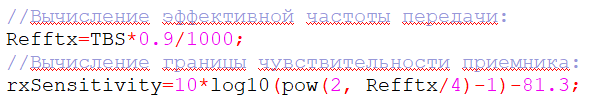
* TBS вычисляется как 8 \* округленное вверх значение (Ninfo2+24)/8 минус 24.



* Если Ninfo меньше или равно 3824:
* Открывается файл "TBS\_for\_Ninfo\_less\_then\_3824.txt".
* Из файла считываются значения до тех пор, пока TBS не станет меньше или равен очередному значению.
* TBS принимает значение считанного из файла числа.
* Файл закрывается.



1. Вычисление эффективной частоты передачи (Refftx) как 0.9 \* TBS / 1000 и границы чувствительности приемника (rxSensitivity) по формуле 10 \* log10(pow(2, Refftx/4) - 1) - 81.3.



1. Вывод полученных результатов в консоль и запись в файлы (код можно посмотреть в приложении 1 под соответствующим комментарием).

Программа позволяет пользователю выбрать таблицу кодирования, ввести необходимые исходные данные и рассчитать ряд параметров, связанных с эффективностью передачи данных в сетях 5G NR.

Алгоритм графической части программы, написанный на python:

1. Подключение необходимых библиотек: pandas, matplotlib.pyplot и os:

import pandas as pd  
import matplotlib.pyplot as plt  
import os

1. Функция graphs():

* Эта функция принимает следующие аргументы: x (данные для оси X), y (данные для оси Y), title (заголовок графика), xlabel (метка для оси X), ylabel (метка для оси Y) и filename (имя файла, в который будет сохранен график).
* Внутри функции:
* Создается новый рисунок с размерами 10x6 дюймов.
* Создается линейный график с фиолетовым цветом, штриховой линией и маркерами в виде окружностей.
* Устанавливаются заголовок, метки для осей X и Y.
* Включается сетка на графике.
* Получается путь к текущему каталогу, в котором находится файл.
* График сохраняется в файл с заданным именем в текущем каталоге.
* Рисунок закрывается.

def graphs(x, y, title, xlabel, ylabel, filename):  
 plt.figure(figsize=(10, 6))  
 plt.plot(x, y, color='purple', linestyle='-.', linewidth=2, marker='o', markersize=5)  
 plt.title(title)  
 plt.xlabel(xlabel)  
 plt.ylabel(ylabel)  
 plt.grid(True)  
  
 # Получаем путь к текущему каталогу  
 current\_dir = os.path.dirname(os.path.abspath(\_\_file\_\_))  
  
 # Сохраняем график в текущий каталог  
 plt.savefig(os.path.join(current\_dir, filename))  
 plt.close()

1. Функция graf\_1():

* Эта функция загружает данные из файлов outputRp.txt и outputrxp.txt с использованием библиотеки Pandas.
* Извлекаются значения для осей X и Y из полученных данных.
* Вызывается функция graphs() с соответствующими аргументами для создания первого графика, который отображает зависимость чувствительности приемника от скорости.

def graf\_1():  
 df1 = pd.read\_csv('outputRp.txt', sep=' ', header=None)  
 df2 = pd.read\_csv('outputrxp.txt', sep=' ', header=None)  
 x = df1[0]  
 y = df2[0]  
 graphs(x, y, 'Зависимость чувствительности приёмника от скорости', 'Значение скорости',  
 'Значение чувствительность приёмника', 'graf1.png')

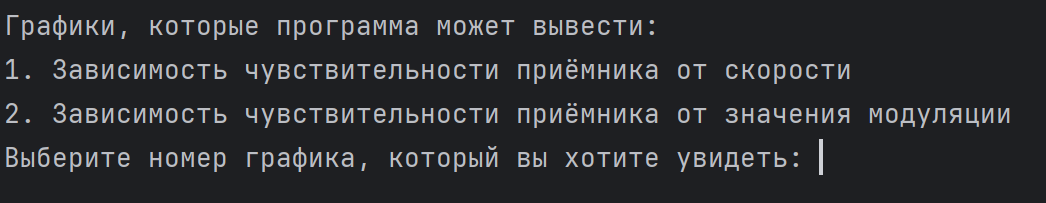
1. Функция graf\_2():

* Эта функция загружает данные из файлов outputQmp.txt и outputrxp.txt с использованием библиотеки Pandas.
* Извлекаются значения для осей X и Y из полученных данных.
* Вызывается функция graphs() с соответствующими аргументами для создания второго графика, который отображает зависимость чувствительности приемника от значения модуляции.

def graf\_2():  
 df1 = pd.read\_csv('outputQmp.txt', sep=' ', header=None)  
 df2 = pd.read\_csv('outputrxp.txt', sep=' ', header=None)  
 x = df1[0]  
 y = df2[0]  
 graphs(x, y, 'Зависимость чувствительности приёмника от значения модуляции', 'Значение модуляции',  
 'Значение чувствительности приёмника', 'graf2.png')

1. Взаимодействие с пользователем:

* Выводится сообщение с описанием двух доступных графиков.
* Пользователю предлагается ввести номер выбранного графика.
* В зависимости от выбора пользователя, вызывается либо graf\_1(), либо graf\_2().
* Если пользователь вводит неверный номер, выводится сообщение об ошибке.



# Список использованных источников

* 5g-tools.com. 5G NR TBS Transport Block Size Calculator [Электронный ресурс]. – Режим доступа: <https://5g-tools.com/5g-nr-tbs-transport-block-size-calculator/>
* Sharetechnote.com. 5G MCS, TBS, and Code Rate [Электронный ресурс]. – Режим доступа: <https://www.sharetechnote.com/html/5G/5G_MCS_TBS_CodeRate.html>
* Techplayon.com. 5G NR Modulation and Coding Scheme (Modulation and Code Rate) [Электронный ресурс]. – Режим доступа: <https://www.techplayon.com/5g-nr-modulation-and-coding-scheme-modulation-and-code-rate/>
* Techplayon.com. 5G NR Transport Block Size (TBS) Calculation [Электронный ресурс]. – Режим доступа: <https://www.techplayon.com/5g-nr-transport-block-size-tbs-calculation/>
* RFwireless-world.com. 5G NR MCS and TBS Table [Электронный ресурс]. – Режим доступа: <https://www.rfwireless-world.com/5G/5G-NR-MCS-and-TBS-Table.html>

# *Приложение 1*

Код вычислительной части программы:

//Подключение необходимых библиотек

#include <iostream>

#include <fstream>

#include <locale>

#include <cmath>

using namespace std;

int main()

{

setlocale(LC\_ALL, "Rus");

float Ninfo, Ninfo2, R, Refftx, rxSensitivity, N\_sh\_symb, Noh, v, Spectral\_efficiency;

int MCS, TBS, n, C, Nre, Nre2, Ndmrs, n\_PRB, Qm, i, table;

//Выбор таблицы кодирование:

do{

cout <<"Выберите желаемую таблицу кодирования:\n";

cout <<"нажмите 1 - если хотите выбрать Table 5.1.3.1-1 MCS index table 1 for PDSCH\n";

cout <<"нажмите 2 - если хотите выбрать Table 5.1.3.1-2 MCS index table 2 for PDSCH\n";

cout <<"нажмите 3 - если хотите выбрать Table 5.1.3.1-3 MCS index table 3 for PDSCH\n";

cin >>table;

}while (table<1 || table>3);

//Ввод дополнительных данных:

do{

cout <<"Введите максимальное количество слоев: ";

cin >>v;

}while (v<0);

do{

cout <<"Введите общее количество выделенных для пользовательского устройства физических ресурсных блоков: ";

cin >>n\_PRB;

}while (n\_PRB<0);

do{

cout <<"Введите количество запланированных OFDM символов: ";

cin >>N\_sh\_symb;

}while (N\_sh\_symb<0);

do{

cout <<"Введите количество ресурсных элементов, выделенных\nдля опорных сигналов демoдуляции на один физический ресурсный блок: ";

cin >>Ndmrs;

}while(Ndmrs<0);

do{

cout <<"Введите накладные расходы, сконфигурированные более высоким уровнем: ";

cin >>Noh;

}while(Noh<0);

for (MCS=0;MCS<32;MCS++)

{

//Ввод индекса MCS и считывание с файла порядка модуляци и кодовой скорости:

if (table==1)

{

ifstream file ("Table 5.1.3.1-1 MCS index table 1 for PDSCH.txt");

for (i=0;i<=MCS;i++)

file >>n>>Qm>>R>>Spectral\_efficiency;

file.close();

}

if (table==2)

{

ifstream file ("Table 5.1.3.1-2 MCS index table 2 for PDSCH.txt");

for (i=0;i<=MCS;i++)

file >>n>>Qm>>R>>Spectral\_efficiency;

file.close();

}

if (table==3)

{

ifstream file ("Table 5.1.3.1-3 MCS index table 3 for PDSCH.txt");

for (i=0;i<=MCS;i++)

file >>n>>Qm>>R>>Spectral\_efficiency;

file.close();

}

R=R/1024;

//Вычисление Ninfo:

Nre2=12\*N\_sh\_symb-Ndmrs-Noh;

if (Nre2>156) Nre=156\*n\_PRB;

else Nre=Nre2\*n\_PRB;

Ninfo=Nre\*R\*Qm\*v;

//Вычисление размера транспортных блоков:

if (Ninfo<=3824)

{

n=floor(log2(Ninfo))-6;

if (n<3) n=3;

Ninfo2=pow(2, n)\*floor(Ninfo/pow(2, n));

if (Ninfo2<24) Ninfo2=24;

}

else

{

n=floor(log2(Ninfo-24))-5;

Ninfo2=pow(2, n)\*round((Ninfo-24)/pow(2, n));

}

if (R<=0.25)

{

C=ceil((Ninfo2+24)/3816);

TBS=(8\*C\*ceil((Ninfo2+24)/(8\*C)))-24;

}

else

if (Ninfo2>=8424)

{

C=ceil((Ninfo2+24)/8424);

TBS=(8\*C\*ceil((Ninfo2+24)/(8\*C)))-24;

}

else TBS=(8\*ceil((Ninfo2+24)/8))-24;

if (Ninfo<=3824)

{

ifstream fl ("TBS\_for\_Ninfo\_less\_then\_3824.txt");

do fl >>n; while (TBS>n);

TBS=n;

fl.close();

}

//Вычисление эффективной частоты передачи:

Refftx=TBS\*0.9/1000;

//Вычисление границы чувствительности приемника:

rxSensitivity=10\*log10(pow(2, Refftx/4)-1)-81.3;

//Вывод полученных результатов в консоль:

cout <<"MCS = "<<MCS<<endl;

cout <<"Порядок модуляции: "<<Qm<<"\nСкорость кодирования [\*1024]: "<<R\*1024<<endl;

if (R==0) cout <<"5G NR TBS (размер транспортного блока): reserved"<<endl<<endl;

else

{

cout <<"TBS = "<<TBS<<endl;

cout <<"Эффективная частота передачи = "<<Refftx<<endl;

cout <<"Граница чувствительности приемника = "<<rxSensitivity<<endl<<endl;

}

//Вывод полученных данных в файл:

if (MCS==0)

{

ofstream fl ("output.txt");

fl <<"MCS = "<<MCS<<endl;

if (R==0) fl <<"5G NR TBS (размер транспортного блока): reserved"<<endl<<endl;

else

{

fl <<"Порядок модуляции: "<<Qm<<"\nСкорость кодирования [\*1024]: "<<R\*1024<<endl;

fl <<"TBS = "<<TBS<<endl;

fl <<"Эффективная частота передачи = "<<Refftx<<endl;

fl <<"Граница чувствительности приемника = "<<rxSensitivity<<endl<<endl;

}

fl.close();

ofstream flp ("rxSensitivity.xls");

flp <<"Индекс MCS\tПорядок модуляции\tCкорость кодирования [\*1024]\tГраница чувствительности приемника"<<endl;

flp <<MCS<<"\t"<<Qm<<"\t"<<R\*1024<<"\t"<<rxSensitivity<<endl;

flp.close();

ofstream flp1 ("outputQmp.txt");

flp1 <<Qm<<endl;

flp1.close();

ofstream flp2 ("outputRp.txt");

flp2 <<R\*1024<<endl;

flp2.close();

ofstream flp3 ("outputrxp.txt");

flp3 <<rxSensitivity<<endl;

flp3.close();

}

else

{

ofstream fl ("output.txt", ios::app);

fl <<"MCS = "<<MCS<<endl;

if (R==0) fl <<"5G NR TBS (размер транспортного блока): reserved"<<endl<<endl;

else

{

fl <<"Порядок модуляции: "<<Qm<<"\nСкорость кодирования [\*1024]: "<<R\*1024<<endl;

fl <<"TBS = "<<TBS<<endl;

fl <<"Эффективная частота передачи = "<<Refftx<<endl;

fl <<"Граница чувствительности приемника = "<<rxSensitivity<<endl<<endl;

}

fl.close();

ofstream flp ("rxSensitivity.xls", ios::app);

if (R==0) flp <<MCS<<"\t"<<"5G NR TBS (размер транспортного блока): reserved"<<endl;

else flp <<MCS<<"\t"<<Qm<<"\t"<<R\*1024<<"\t"<<rxSensitivity<<endl;

flp.close();

if (R!=0)

{

ofstream flp1 ("outputQmp.txt", ios::app);

flp1 <<Qm<<endl;

flp1.close();

ofstream flp2 ("outputRp.txt", ios::app);

flp2 <<R\*1024<<endl;

flp2.close();

ofstream flp3 ("outputrxp.txt", ios::app);

flp3 <<rxSensitivity<<endl;

flp3.close();

}

}}

return 0;

}

# *Приложение 2*

Код программы:

import pandas as pd  
import matplotlib.pyplot as plt  
import os  
  
  
def graphs(x, y, title, xlabel, ylabel, filename):  
 *"""  
 Функция для построения и сохранения графика.  
  
 Args:  
 x (pandas.Series): Данные для оси X.  
 y (pandas.Series): Данные для оси Y.  
 title (str): Заголовок графика.  
 xlabel (str): Подпись оси X.  
 ylabel (str): Подпись оси Y.  
 filename (str): Название файла, в который будет сохранен график.  
  
 Returns:  
 None  
 """* plt.figure(figsize=(10, 6))  
 plt.plot(x, y, color='purple', linestyle='-.', linewidth=2, marker='o', markersize=5)  
 plt.title(title)  
 plt.xlabel(xlabel)  
 plt.ylabel(ylabel)  
 plt.grid(True)  
  
 # Получаем путь к текущему каталогу  
 current\_dir = os.path.dirname(os.path.abspath(\_\_file\_\_))  
  
 # Сохраняем график в текущий каталог  
 plt.savefig(os.path.join(current\_dir, filename))  
 plt.close()  
  
  
def graf\_1():  
 *"""  
 Построение первого графика, отображающего зависимость чувствительности приёмника от скорости.  
  
 Returns:  
 None  
 """* df1 = pd.read\_csv('outputRp.txt', sep=' ', header=None)  
 df2 = pd.read\_csv('outputrxp.txt', sep=' ', header=None)  
 x = df1[0]  
 y = df2[0]  
 graphs(x, y, 'Зависимость чувствительности приёмника от скорости', 'Значение скорости',  
 'Значение чувствительность приёмника', 'graf1.png')  
  
  
def graf\_2():  
 *"""  
 Построение второго графика, отображающего зависимость чувствительности приёмника от значения модуляции.  
  
 Returns:  
 None  
 """* df1 = pd.read\_csv('outputQmp.txt', sep=' ', header=None)  
 df2 = pd.read\_csv('outputrxp.txt', sep=' ', header=None)  
 x = df1[0]  
 y = df2[0]  
 graphs(x, y, 'Зависимость чувствительности приёмника от значения модуляции', 'Значение модуляции',  
 'Значение чувствительности приёмника', 'graf2.png')  
  
  
print("Графики, которые программа может вывести:")  
print("1. Зависимость чувствительности приёмника от скорости")  
print("2. Зависимость чувствительности приёмника от значения модуляции")  
choice = int(input("Выберите номер графика, который вы хотите увидеть: "))  
if choice == 1:  
 graf\_1()  
elif choice == 2:  
 graf\_2()  
else:  
 print("Неверный ввод.")

1. <https://www.rfwireless-world.com/5G/5G-NR-MCS-and-TBS-Table.html> [↑](#footnote-ref-2)