МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**КУРСОВАЯ РАБОТА**

**СИСТЕМА ГЕНЕРАЦИИ QR КОДА**

Работу выполнил К.С. Минин

(подпись)

Направление подготовки 02.03.02 Фундаментальная информатика и информационные технологии

Курс 3

Научный руководитель

канд. пед. наук, доц. Н.Ю. Добровольская

(подпись, дата)

Нормоконтролер

ст. преп. А.В. Харченко

(подпись, дата)

Краснодар

2021

**РЕФЕРАТ**

В курсовой работе 51 страница, 45 рисунков, 11 таблиц, 7 источников.

Ключевые слова: QR CODE, АЛГОРИТМ РИДА-СОЛОМОНА, PIL, БАЙТОВОЕ КОДИРОВАНИЕ, ПОЛЕ ГАЛУА.

Цель курсовой работы – реализовать программу, генерирующую QR code заданного текста.

При подготовке курсовой работы изучена структура QR code, алгоритмы кодирования информации, а также алгоритм Рида-Соломона для создания байтов коррекции.

В работе выполнен реферативный обзор нескольких сервисов, позволяющих генерировать QR code различных видов, выделены их достоинства и недостатки.

В практической части курсовой работы реализовано приложение на языке программирования Python с использованием модуля PIL для создания изображения QR кода, позволяющее получить QR code текста, ссылки, номера мобильного телефона, СМС-сообщения, электронного письма, сети Wi-Fi.

**СОДЕРЖАНИЕ**

[Введение 4](#_Toc90018285)

[1 Обзор готовых сервисов для генерации QR-кодов 5](#_Toc90018286)

[1.1 Сервис QR Coder.ru 5](#_Toc90018287)

[1.2 Сервис qrcode.website 7](#_Toc90018288)

[1.3 Сервис qrcod.ru 11](#_Toc90018289)

[2 Задача генерации QR кода 17](#_Toc90018290)

[2.1 Основные определения 17](#_Toc90018291)

[2.2 Структура символов QR code 19](#_Toc90018292)

[2.3 Кодирование данных 22](#_Toc90018293)

[2.4 Добавление служебной информации и заполнение 25](#_Toc90018294)

[2.5 Разделение информации на блоки 28](#_Toc90018295)

[2.6 Создание байтов коррекции 32](#_Toc90018296)

[2.7 Объединение информационных и корректирующих блоков 35](#_Toc90018297)

[2.8 Размещение полученных данных в символе QR Code 36](#_Toc90018298)

[3 Результаты работы приложения 47](#_Toc90018299)

[Заключение 51](#_Toc90018300)

[Список использованных источников 52](#_Toc90018301)

# ВВЕДЕНИЕ

Современная деятельность человека во всех областях максимально цифровизирована. В настоящее время широкое распространение получает механизм идентификации объекта – QR код. В отличие от штрих-кода, который используется длительное время, QR код имеет следующие преимущества: позволяет кодировать больше информации, чем линейные штрих коды, при таком же или меньшем размере изображения; легко распознается сканирующим оборудованием под любым углом; может быть прочитан, даже если повреждения кода составляют около 30 %.

Существует достаточно много свободно распространяемых приложений генерации QR кодов. Однако QR код может быть составляющим некоторого комплекса программ, решающих конкретную задачу. В этом случае генерация кода сторонним приложением неудобна и небезопасна. Например, механизм QR кодов можно использовать для учета посещаемости студентов.

В предлагаемой курсовой работе реализован первый этап исследования – генерация QR кода.

# 1 Обзор готовых сервисов для генерации QR-кодов

QR код ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Quick Response code — код быстрого реагирования; [сокр.](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D0%B1%D1%80%D0%B5%D0%B2%D0%B8%D0%B0%D1%82%D1%83%D1%80%D0%B0) QR code) — тип двухмерных [штриховых кодов](https://ru.wikipedia.org/wiki/%D0%A8%D1%82%D1%80%D0%B8%D1%85%D0%BE%D0%B2%D0%BE%D0%B9_%D0%BA%D0%BE%D0%B4), который легко считывается цифровым устройством и хранит информацию в виде серии пикселей в квадратной сетке, внешне выглядящий как черно-белый узор. Был разработан для использования в автомобильной промышленности, однако  благодаря возможности быстрого считывания и большей емкости по сравнению со штрихкодами [стандарта UPC](https://ru.wikipedia.org/wiki/Universal_Product_Code) (Universal Product code) очень быстро получил распространение во многих других областях жизни людей по всему миру [1].

В открытом доступе сети интернет существует множество онлайн-сервисов для создания QR кодов, причем каждый из них имеет индивидуальный функционал и интерфейс.

## **Сервис QR Coder.ru**

Первый из рассмотренных сервисов обладает простым и понятным интерфейсом (рисунок 1) [2]. Выбор типов данных, для которых можно создать QR код, не очень большой. Однако, здесь представлены основные и часто используемые виды информации: текст, ссылка на сайт, визитная карточка (при сканировании предлагается создать контакт с заполненными данными), и sms-сообщения (при сканировании будет предложено отправить СМС в приложении «сообщения»).



Рисунок 1 – Главная страница сервиса QR Coder.ru

При создании QR кода также можно выбрать его размер – увеличение/уменьшение изображения готового кода (рисунок 2).



Рисунок 2 – Страница создания QR кода текста сервиса QR Coder.ru

Данный сервис генерирует не только изображение, но и постоянную ссылку на страницу сайта, которая содержит QR код. Также создается HTML-код для вставки сгенерированного QR кода в сайт.

К недостаткам рассматриваемого сервиса можно отнести ограниченную функциональность, а также ничем не примечательный интерфейс. Однако, основную задачу по генерации QR кода он выполняет полностью.

## **Сервис qrcode.website**

Второй из рассмотренных сервисов является крупным проектом, который имеет привлекательный сайт (рисунок 3) [7]. Здесь можно создать свою учетную запись, чтобы сохранять сделанные ранее QR коды. Однако, сгенерировать код можно и без регистрации.

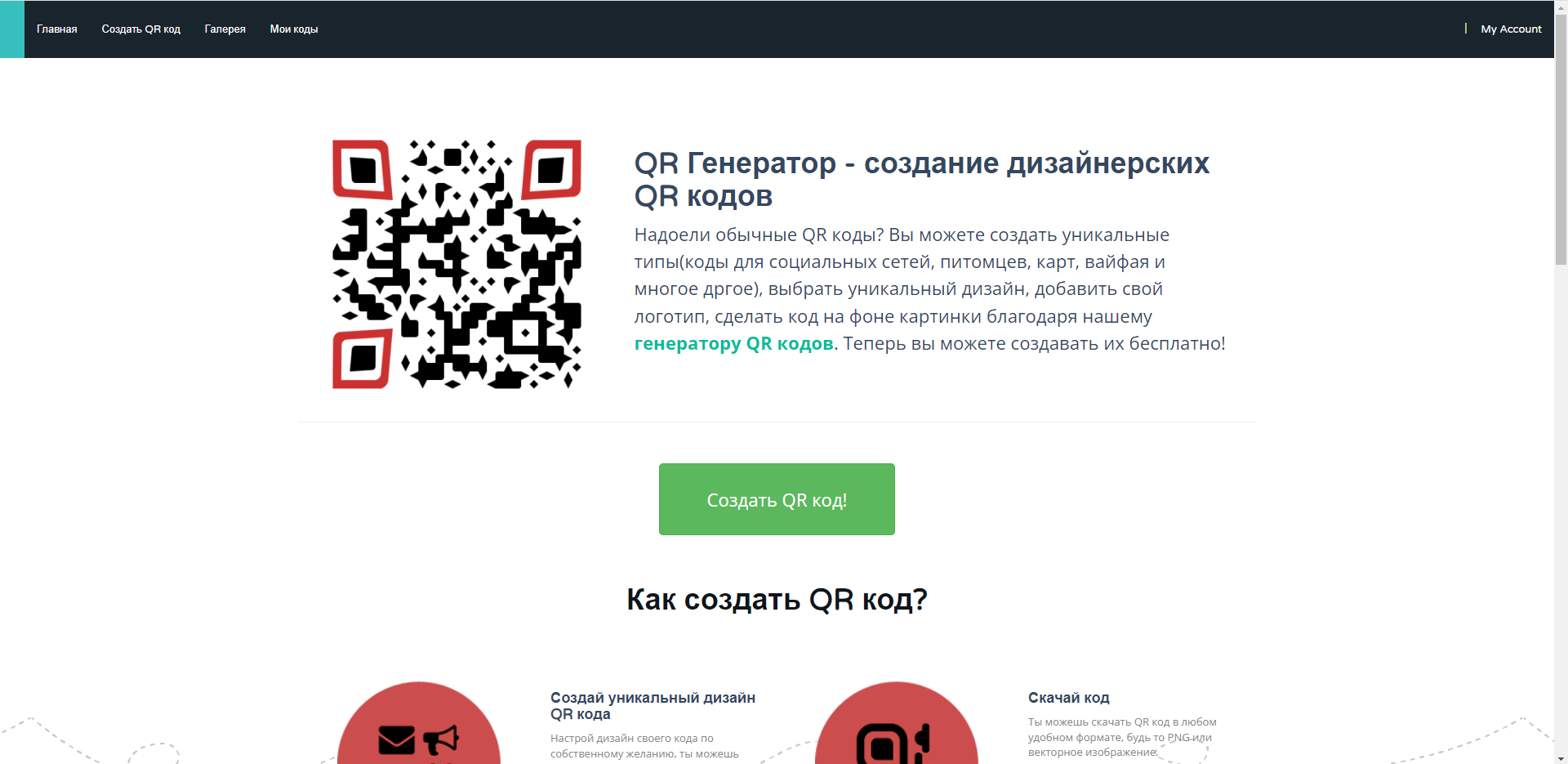


Рисунок 3 – Главная страница сервиса qrcode.website

В этом сервисе пользователь может выбрать один из следующих типов QR кода: ссылка (при сканировании предлагается перейти на сайт); текст (при сканировании отображается текст); телефон (при сканировании будет предложено совершить звонок через приложение «телефон»); визитка (при сканировании будет предложено создать контакт в телефонной книге); купон, питомец, товар, Twitter, Facebook, PlayMarket, AppStore, карта (при сканировании будет предложено перейти на сайт, либо созданный генератором, либо по указанному URL); Email (при сканировании будет создано предзаполненное электронное письмо); SMS (при сканировании будет предложено отправить СМС в приложении «сообщения») (рисунок 4).

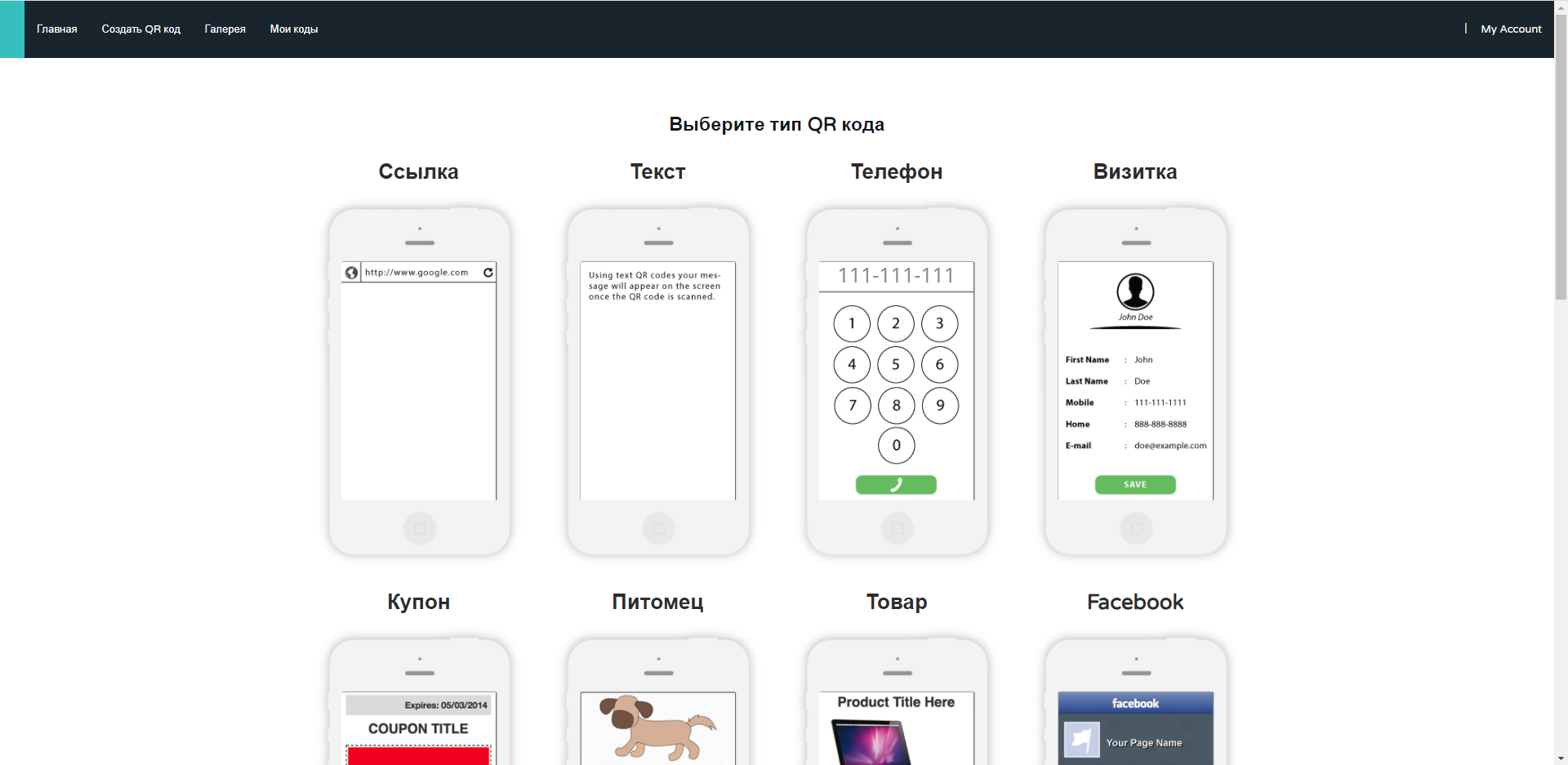


Рисунок 4 – Разновидности генерируемых кодов сервиса qrcode.website

При выборе одного из типов кода, произойдет переход на окно ввода данных, которые будут содержаться в QR коде. Это окно отличается в зависимости от выбранного вида кода (рисунок 5).

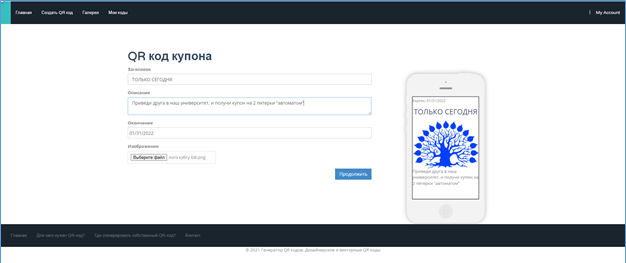


Рисунок 5 – Страница создания QR кода купона сервиса qrcode.website

Например, создан шуточный QR код купон для студентов Кубанского Государственного Университета (рисунок 6).



Рисунок 6 – Созданный QR-код купона сервиса qrcode.website

Qrcode.website обладает большим выбором настроек внешнего вида QR кода. Так, например, пользователь может добавить картинку на фон кода (рисунок 7).



Рисунок 7 – Код с фоновой картинкой сервиса qrcode.website

Есть возможность изменить форму и цвет точек, а также поисковых узоров (рисунок 8).



Рисунок 8 – Код с измененными точками сервиса qrcode.website

Сервис предоставляет возможность добавления логотипа на полотно QR кода и, соответственно, объединение всех возможных изменений внешнего вида кода (рисунок 9).



Рисунок 9 – Код с измененным внешним видом сервиса qrcode.website

У пользователей сервиса есть возможность показать созданные ими QR коды всем посетителям данного сайта. Для этого есть возможность добавления кода из личного кабинета в раздел «галерея» (рисунок 10).

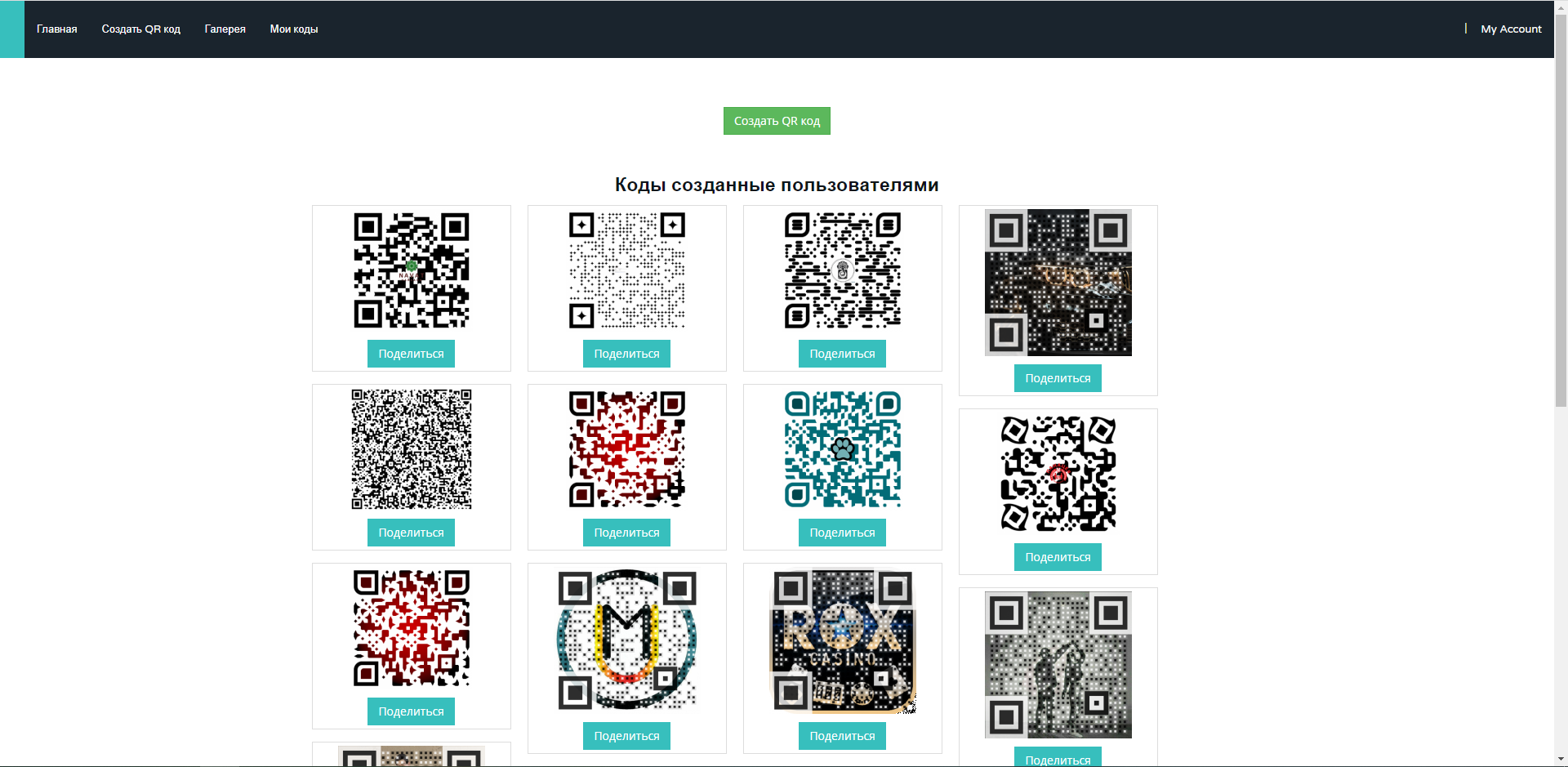


Рисунок 10 – Раздел «галерея» сервиса qrcode.website

Несомненно, сервис имеет очень большой функционал – есть возможность создания кодов разных типов, их индивидуализации и дизайна; сохранения QR кодов в личном аккаунте; выставка кодов в галерее.

Очевидным минусом является то, что информация в большинстве типов кодов кодируется не напрямую, а через ссылки на страницы внутри qrcode.website. То есть у пользователя, сканирующего код, не будет возможности просмотреть зашифрованную информацию (даже текстовую) без подключения к сети интернет.

Еще один минус был выявлен на этапе тестирования созданных в данном сервисе кодов. При использовании большого количества «дизайнерских» элементов (измененных поисковых узоров и точек, фоновой картинки, логотипа и т.д.) только двум из трех используемых сканеров QR кодов удалось расшифровать информацию.

## **Сервис qrcod.ru**

Первое преимущество данного сервиса над остальными – разнообразие видов создаваемых QR кодов (рисунок 11) [6].

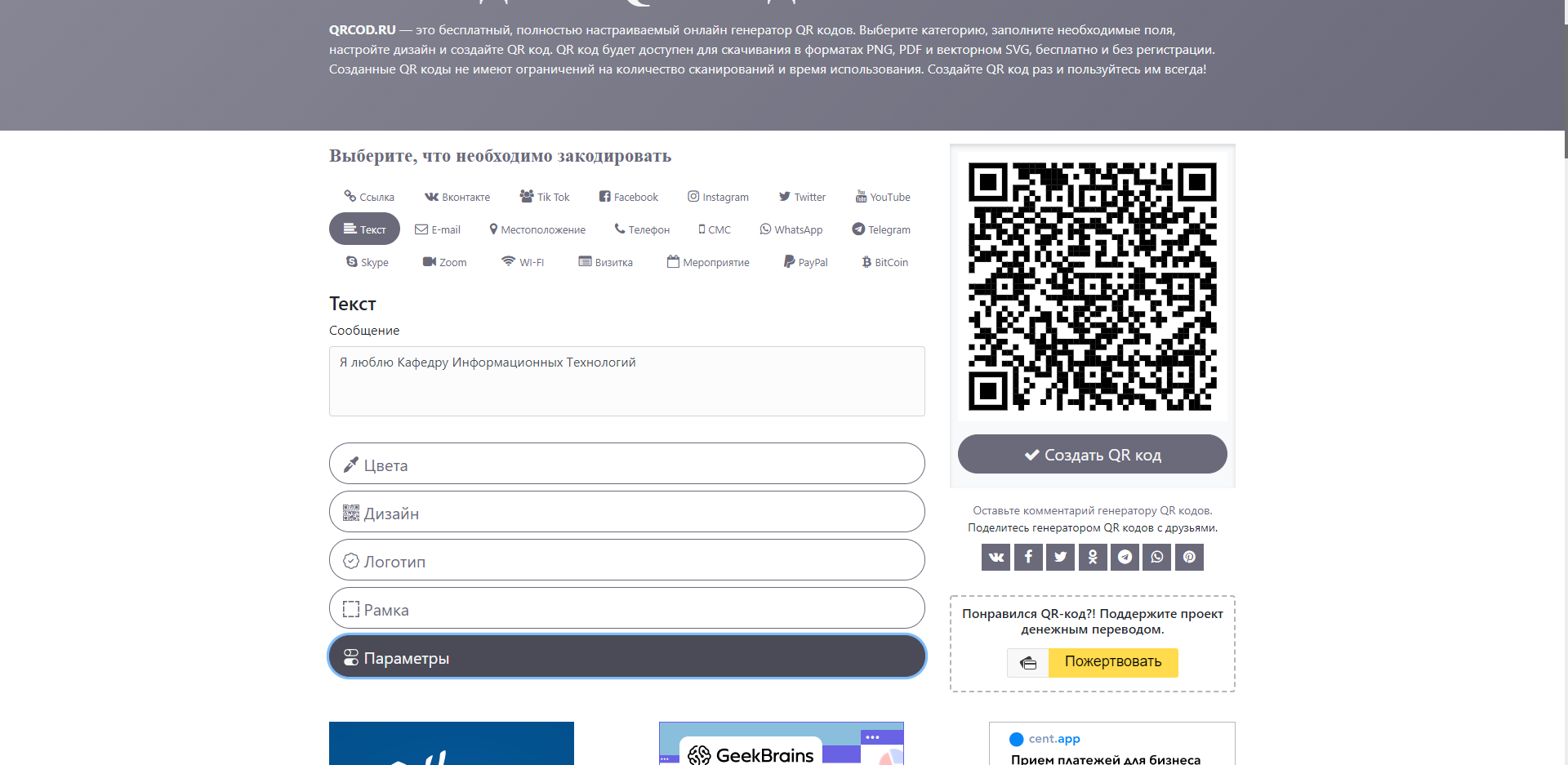


Рисунок 11 –Виды QR кодов в сервисе qrcod.ru

Несмотря на то, что большинство типов кодов совпадает с предложенными сервисом qrcode.website, рассмотренным в предыдущей части, на сайте qrcod.ru присутствуют и другие виды, что несомненно делает его более привлекательным для пользователей.

Например, при сканировании кода «мероприятие» пользователю предлагается добавить отметку о событии в календарь (рисунок 12).

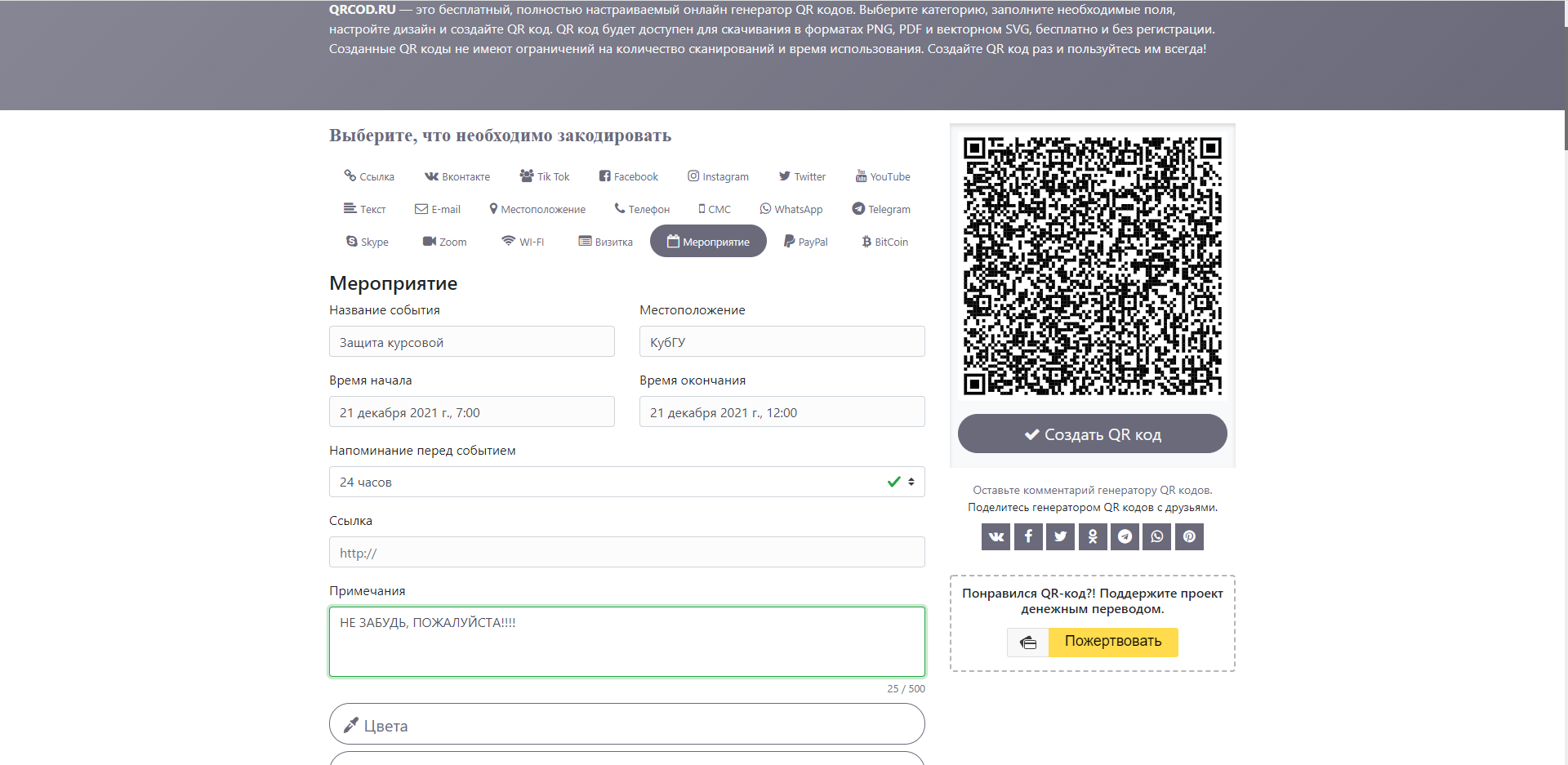


Рисунок 12 – Создание кода мероприятия в сервисе qrcod.ru

При сканировании QR кода типа «Wi-Fi» система предложит пользователю подключиться к соответствующей беспроводной сети (рисунок 13). Единственный недостаток – при попытке сканирования такого кода, одному из трех сканеров не удалось подключиться к сети Wi-Fi, однако я полагаю, что это недостаток программы-сканера, а не созданного QR кода.

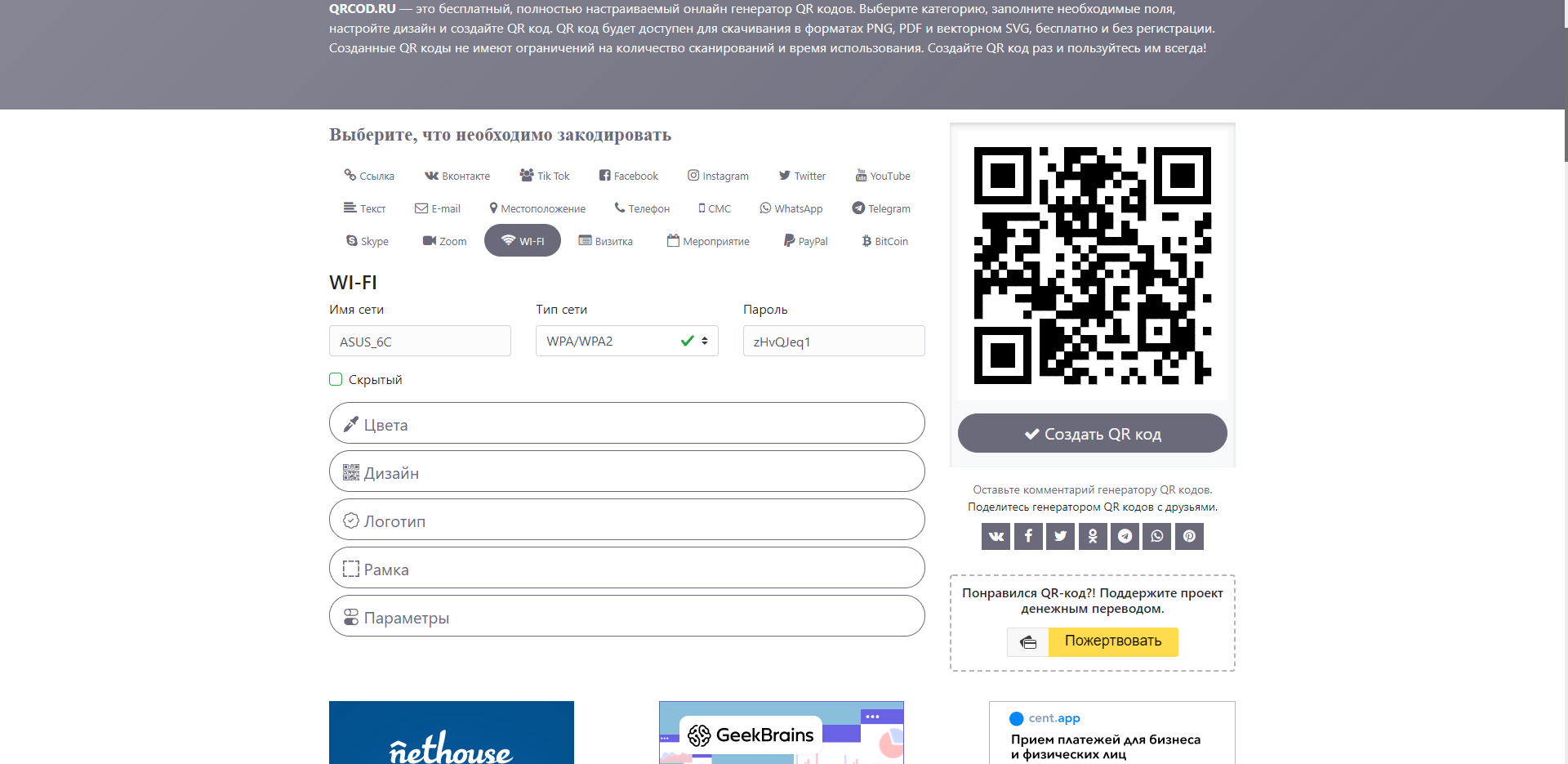


Рисунок 13 – Создание кода wi-fi в сервисе qrcod.ru

При сканировании кода типа «WhatsApp» открывается чат в одноименном приложении и предзаполненное сообщение. Виды «Telegram», «Skype», «Zoom», «PayPal», «BitCoin», «YouTube», «Вконтакте», «Instagram» кодируют ссылки на соответствующие сервисы, содержащие введенные в генераторе данные.

Возможности изменения внешнего вида кода почти не отличаются от предложенных предыдущим сервисом.

На сайте qrcod.ru пользователь имеет возможность выбора «качества» QR кода (рисунок 14). Этот пункт настроек отвечает за уровень коррекции ошибок, который будет применятся в генерируемом коде.

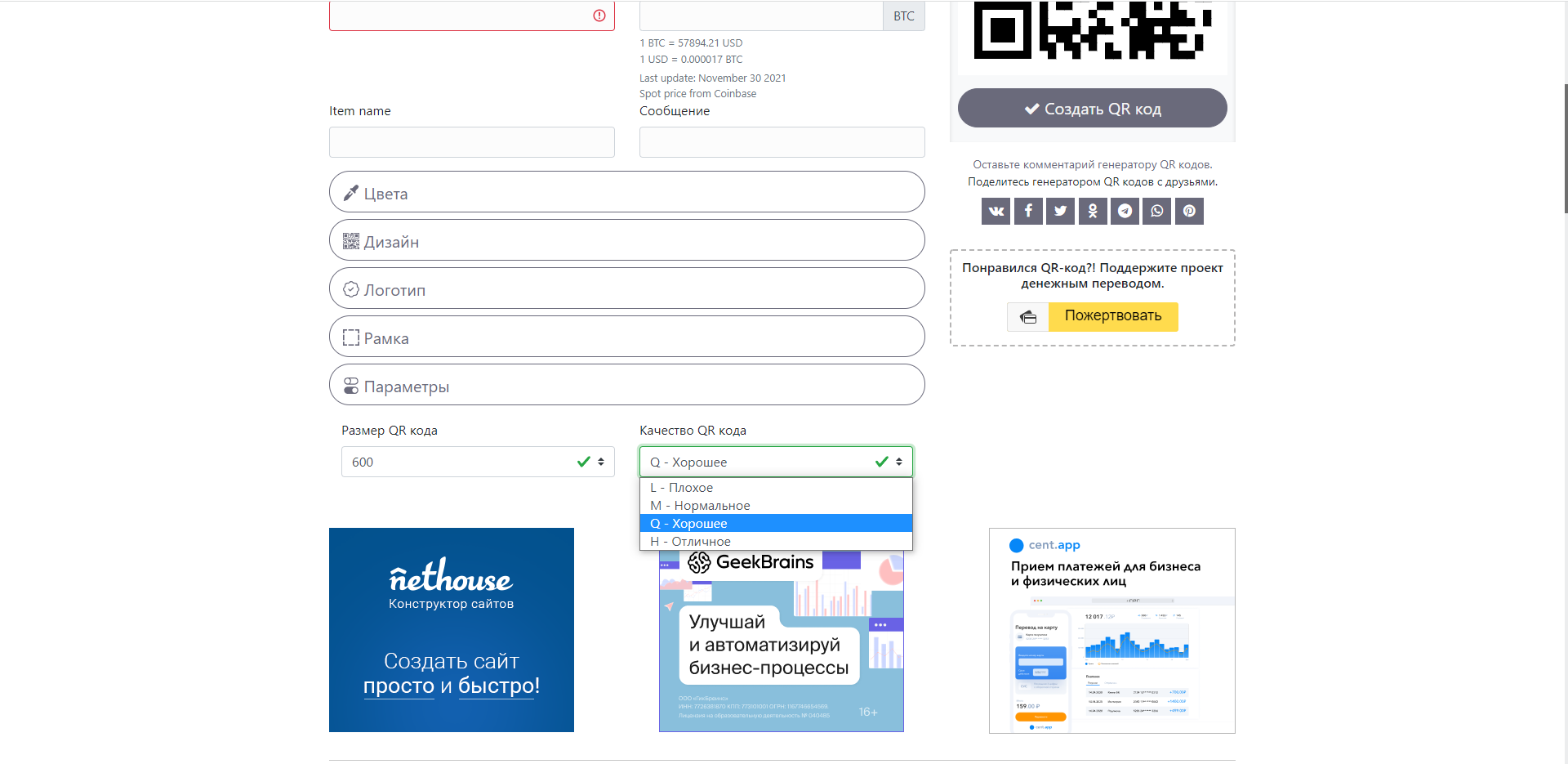


Рисунок 14 – Выбор «качества» кода в сервисе qrcod.ru

Несомненным плюсом является наличие информации для пользователей, которая находится внизу страницы (рисунок 15).

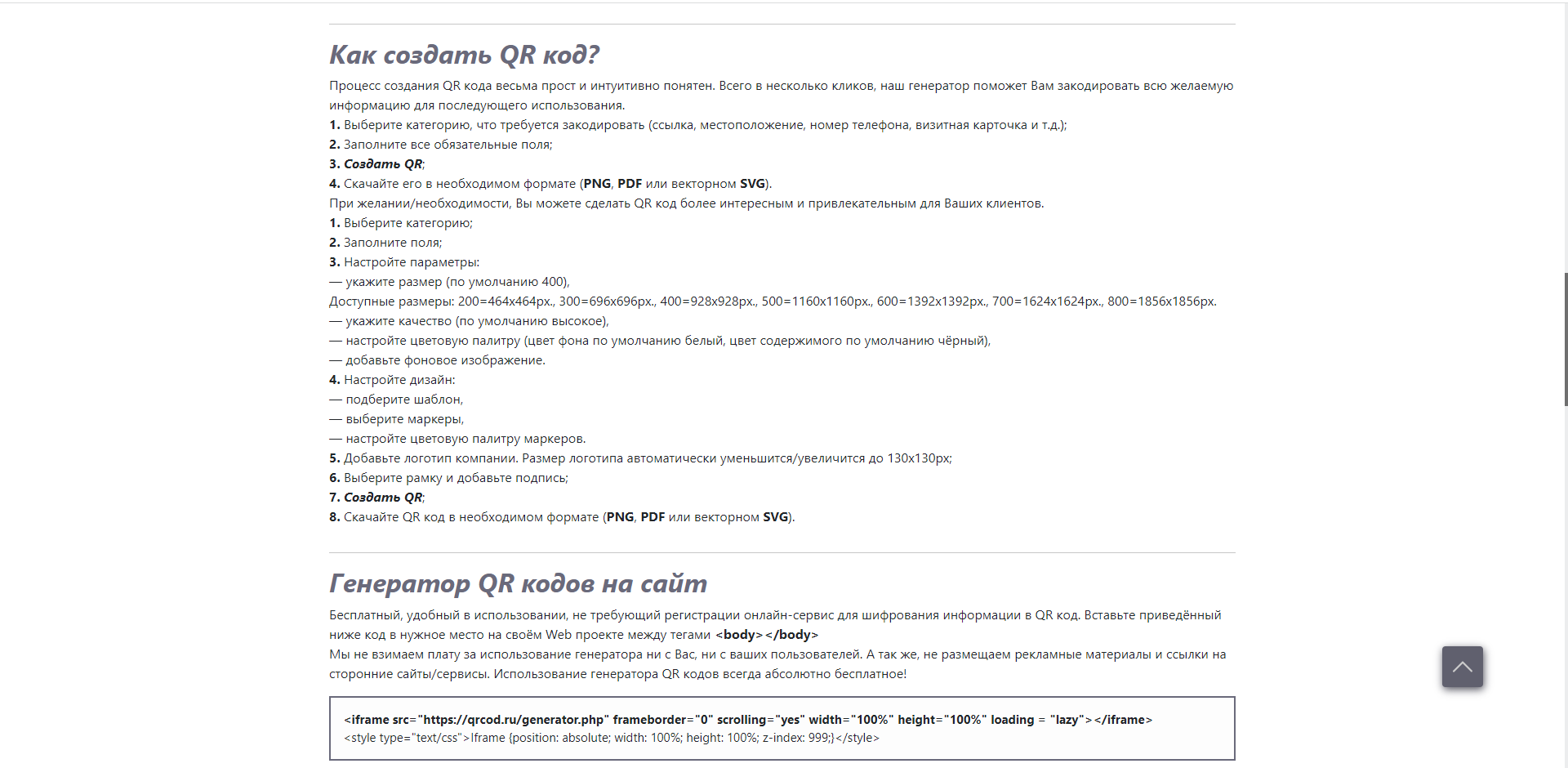


Рисунок 15 – Дополнительная информация в сервисе qrcod.ru

Здесь же посетители сайта могут найти и комментарии других людей, которые уже испытали возможности этого генератора QR кодов.

При тестировании данного сервиса, также был выявлен один «баг»: при кодировании текста, содержащего символы букв русского алфавита, а также специальные символы, создается код текста, расположенного до специальных символов (рисунок 16).

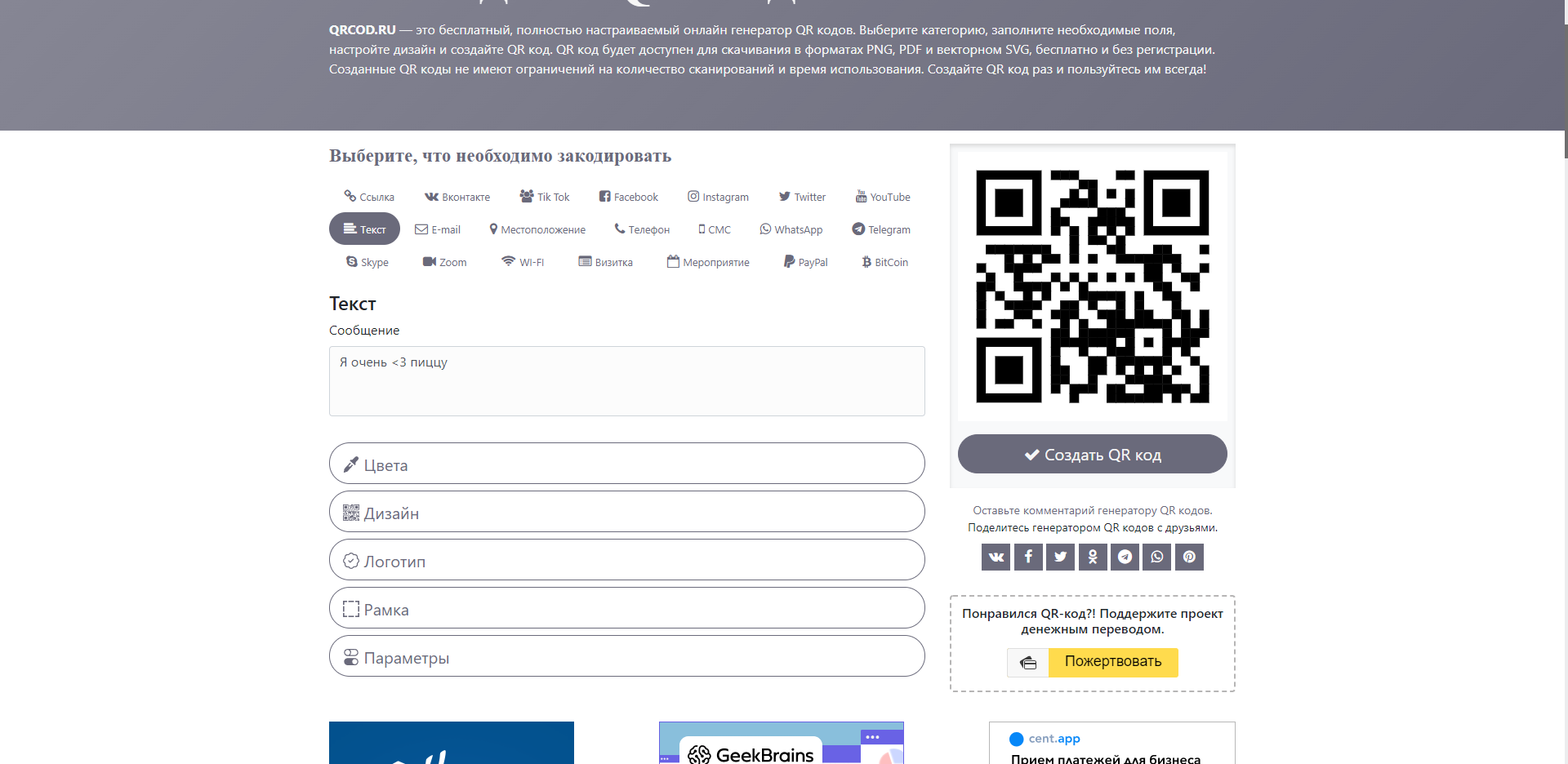


Рисунок 16 – Код с багом в сервисе qrcod.ru

Результат сканирования такого кода приведен на рисунке 17.

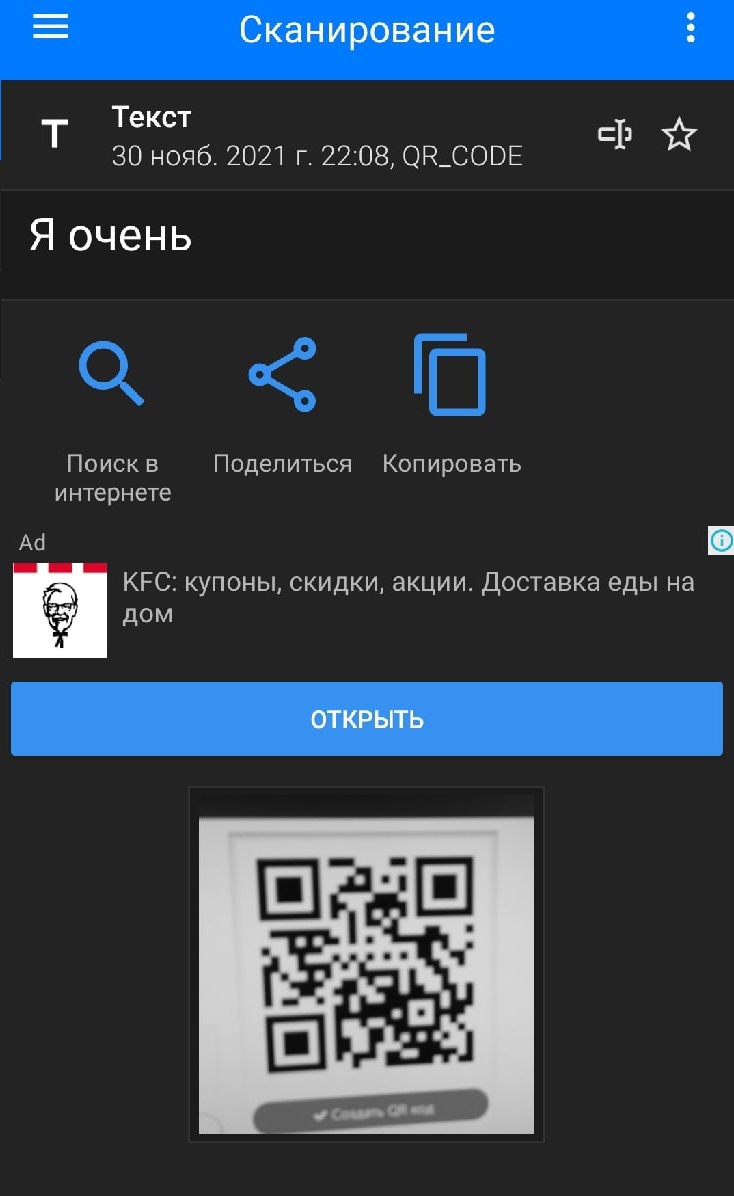


Рисунок 17 – Сканирование неправильного кода

Несмотря на недостатки в виде небольших «багов», данный сервис имеет красивый и удобный интерфейс, а также большую функциональность, что несомненно делает его более привлекательным для пользователей, по сравнению с другими рассмотренными сервисами.

# 2 Задача генерации QR кода

В общем случае задача создания QR кода включает в себя следующие этапы: кодирование данных; добавление служебной информации и заполнение; разделение информации на блоки; создание байтов коррекции; объединение информационных и корректирующих блоков; размещение полученных данных на QR коде.

## **2.1 Основные определения**

QR Code – матричная символика. Символ состоит из массива номинально квадратных модулей, структурированных в регулярную квадратную матрицу, включая уникальные шаблоны поиска, размещенные в трех углах символа (в Micro QR Code – только в одном углу) и предназначенные для упрощения определения места нахождения, размера и наклона символа [3]. Предусмотрен широкий диапазон размеров и четыре уровня исправления ошибок. Размер символа в модулях устанавливает пользователь для обеспечения производства символов различными методами.

Формат Micro QR Code – вариант QR Code с уменьшенным числом служебных модулей и ограниченным диапазоном размеров, позволяющим представлять малые объемы данных в символах небольшого размера, особенно подходящий для прямого маркирования деталей и компонентов и для применений с ограниченным пространством для размещения символа.

Функциональный шаблон (function pattern) – дополнительные компоненты символа (шаблоны поиска, разделитель, шаблоны синхронизации, направляющие шаблоны), которые требуются для уточнения места нахождения символа или определения его параметров с целью обеспечения декодирования.

Область кодирования (encoding region) – область символа, свободная от функциональных шаблонов и предназначенная для кодирования данных и кодовых слов исправления ошибок и информации о версии и формате символа.

Маскирование (masking) ­– процесс применения операции XOR к комбинации битов в какой-либо области символа с использованием шаблона маски для уменьшения комбинаций, которые могут препятствовать быстрой обработке изображения.

Указатель шаблона маски данных (data mask pattern reference) – трехбитовый идентификатор, указывающий шаблон маски данных, используемый в символе.

Бит-заполнитель (padding bit) – нулевой бит, не представляющий данные, предназначенный для заполнения оставшихся незаполненными позиций последнего кодового слова после ограничителя в двоичной строке данных. Кодируемые данные пакуются в двоичный поток с точностью до бита, а алгоритмы размещения данных в символе и расчета исправления ошибок используют 8-битовые кодовые слова. Поэтому в случае, когда последнее кодовое слово остается заполненным не до конца, используют биты-заполнители для обеспечения корректной работы алгоритмов.

Остаточный бит (remainder bit) – нулевой бит, не представляющий данные, используемый для заполнения оставшихся незаполненными позиций в области кодирования после последнего знака символа в случае, если область кодирования, предназначенная для знаков символа, не полностью распределена для размещения 8-битовых знаков символа.

Режим (mode) – способ преобразования последовательности кодируемых знаков в двоичную строку.

Индикатор режима (mode indicator) – идентификатор, состоящий из четырех битов, указывающий режим, в котором закодирована следующая за ним последовательность данных.

Версия (version) – размер символа в соответствии с его позицией в последовательности допустимых размеров символов; символы Micro QR Code имеют размеры (в модулях) от 11х11 (версия М1) до 17х17 (версия М4), а символы QR Code - от 21х21 (версия 1) до 177х177 (версия 40). Уровень исправления ошибок, применимый к символу, может быть добавлен к обозначению версии, например, версия 4-L или версия M3-Q.

В QR коде может использоваться один из четырех уровней исправления ошибок Рида-Соломона (обозначаемые L, M, Q и H в порядке увеличения занимаемой емкости): L - 7%; M - 15%; Q - 25%; H - 30 % от числа кодовых слов символа [3].

Для символов Micro QR Code уровень исправления ошибок Н не используют. Для Micro QR Code версии M1 алгоритм Рида-Соломона обеспечивает только возможность обнаружения ошибок;

## **2.2 Структура символов QR code**

Каждый символ QR Code состоит из номинально квадратных модулей, структурированных в регулярную матрицу, и включает область кодирования и функциональные шаблоны, называемые шаблонами поиска, разделителями, шаблонами синхронизации и направляющими шаблонами [3]. Функциональные шаблоны не содержат данных. Символ должен быть окружен со всех четырех сторон свободной зоной. На рисунке 18 представлена структура символа версии 7.

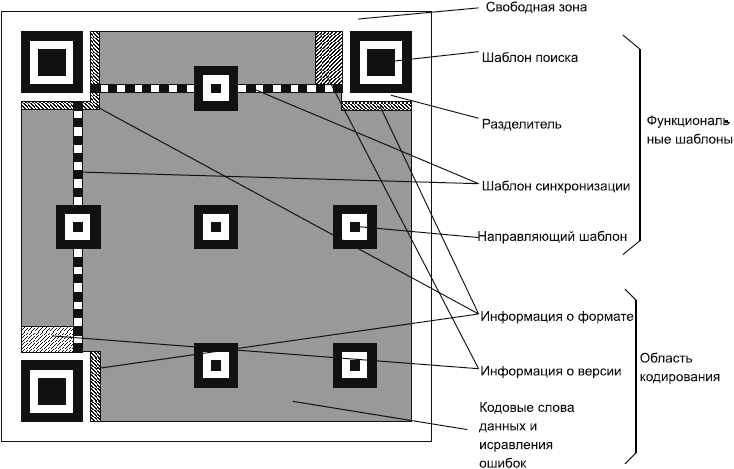


Рисунок 18 – Структура символа QR code версии 7

Свободная зона должна быть свободной от всех других маркировок и окружать символ со всех четырех сторон. Коэффициент отражения свободной зоны должен соответствовать коэффициенту отражения светлых модулей. Для символов QR Code ширина свободной зоны должна быть равна 4Х. Для символов Micro QR Code ширина свободной зоны должна быть равна 2Х.

Разделитель (separator) – функциональный шаблон, состоящий из светлых модулей, шириной один модуль, отделяющий шаблоны поиска от остального содержимого символа.

Шаблон синхронизации (timing pattern) – чередующаяся последовательность темных и светлых модулей, предназначенная для определения координат модулей в символе [3].

Направляющие шаблоны присутствуют только в символах QR Code версии 2 или версии с большим номером. Каждый шаблон представляет собой суперпозицию трех вложенных друг в друга квадратов и состоит из темного квадрата размерами (в модулях) 5х5, светлого размерами (в модулях) 3х3 и единственного темного модуля в центре. Число направляющих шаблонов зависит от версии символа и шаблоны должны присутствовать во всех символах, начиная с версии 2 и версии с большим номером

Информация о формате (format information) – шаблон, в котором закодирована информация об основных параметрах, необходимых для обеспечения декодирования оставшейся части области кодирования.

Информация о версии (version information) – шаблон, закодированный в некоторых символах QR Code, содержащий информацию о версии символа вместе с битами исправления ошибок для этих данных.

В символе любого QR Code имеются три одинаковых шаблона поиска, размещенных в левом верхнем, правом верхнем и левом нижнем углах символа [3]. Каждый шаблон поиска выглядит как три вложенных друг в друга квадрата: темного размерами (в модулях) 7х7, светлого размерами (в модулях) 5х5 и темного размерами (в модулях) 3х3 (рисунок 19). Размеры в модулях в любом шаблоне поиска находятся в соотношении 1:1:3:1:1, как показано на рисунке 19. Представление кодируемых данных должно быть таким чтобы, что вероятность встретить такой шаблон среди кодируемых данных была очень мала, что позволяет быстро определить присутствие символа QR Code в поле обзора. Идентификация трех шаблонов, составляющих шаблон поиска, позволяет определить место нахождения и угловую ориентацию символа в поле обзора.

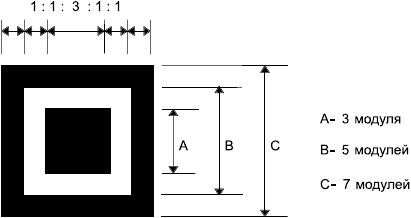


Рисунок 19 – Структура шаблона поиска

## **2.3 Кодирование данных**

Знаки данных должны быть преобразованы в двоичный поток в соответствии с правилами, установленными для действующего режима. При необходимости смены режима вставляют индикаторы режима в начале каждого нового сегмента, а в конце данных вставляют ограничитель. Результирующий двоичный поток преобразуется в последовательность 8-битовых кодовых слов. Добавляют необходимое число знаков-заполнителей для образования требуемого числа кодовых слов в соответствии с выбранной версией символа.

Стандарт QR Code поддерживает следующие режимы кодирования данных.

Числовой режим кодирует данные, состоящие из десятичных цифр (0-9). Вся последовательность символов разбивается на группы по 3 цифры, и каждая группа (трехзначное число) переводится в 10-битное двоичное число. В случае, когда общее количество символов не кратно трем, возможны следующие ситуации. Если остаток от деления общего числа символов на три равен двум, то последние две цифры исходной последовательности данных кодируются 7 битами. Если остаток от деления равен одному, то последний символ кодируется четырьмя битами.

Например, входная строка имеет вид: «11223344567». Она разделяется на группы «112», «233», «445» и «67». Двоичные представления будут равны соответственно «00001110000», «00011101001», «00110111101», «1000011». Далее двоичные последовательности объединяются в один поток данных: «0000111000000011101001001101111011000011».

Алфавитно-цифровой режим позволяет закодировать входные данные из набора 45 знаков: 10 десятичных цифр (0-9), 26 латинских букв (A-Z) и 9 специальных символов (SP, $, %, \*, +, -, ., /, :).

В этом случае каждому знаку входных данных должно быть присвоено значение от 0 до 44 в соответствии с таблицей 1. Исходная последовательность разбивается на группы по 2 символа, и каждая группа кодируется 11 битами. Значение для первого знака умножают на 45 и к нему прибавляют значение для второго знака. Сумму преобразуют в 11-битовое двоичное число. Если во входных данных находится нечетное число знаков, последний знак кодируют 6-битовым двоичным числом [1].

Таблица – Соответствие символов для алфавитно-цифрового режима

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Знак | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E |
| Значение | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| Знак | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
| Значение | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| Знак | U | V | W | X | Y | Z | Пробел | $ | % | \* | + | - | . | / | : |
| Значение | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 |

Например, входная строка имеет вид: «HIKUBSU». Она разделяется на группы: «HI», «KU», «BS» и «U». Из таблицы каждому символу в каждой группе находится соответствие: (17, 18), (20, 30), (11, 28) и (30). Далее находится одиннадцатибитное (или шестибитное) значение для каждой группы.

Первая группа: 17 \* 45 + 18 = 78310 = 11000011112.

Вторая группа: 20 \* 45 + 30 = 93010 = 011101000102.

Третья группа: 11 \* 45 + 28 = 52310 = 010000010112.

Четвертая группа: 3010 = 0111102.

Полученные двоичные последовательности объединяются в один поток данных: «11000011110111010001001000001011011110».

Байтовый режим – это универсальный способ кодирования, позволяющий закодировать любые символы. Однако при этом наблюдается относительно низкая плотность информации. В случае использования данного режима для кодирования данных, исходный текст представляется в любой кодировке (для правильного отображения Кириллицы и универсальности кода рекомендована UTF-8) и полученная последовательность байт переводится в один поток без изменений.

В реализованном приложении используется метод байтового кодирования исходных текстовых данных (рисунок 20). Входная строка преобразуется в поток объектов типа Bytes с помощью встроенного метода .encode(). Полученный поток байтов преобразуется в бинарное представление в методе binary\_encoding() с помощью встроенного метода bin(), который преобразует объект типа Bytes в двоичное число. В методе binary\_encoding() реализовано дополнение двоичного числа, длина которого меньше 8 бит, незначащими нулями.

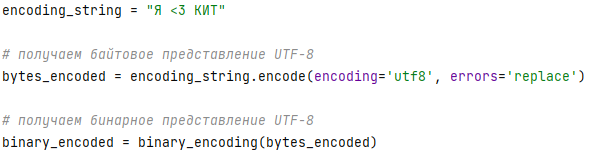


Рисунок 20 – Байтовое кодирование в реализованном приложении

Например, входная строка имеет вид: «Я <3 КИТ». Результат кодирования в реализованном приложении отображен на рисунке 21.

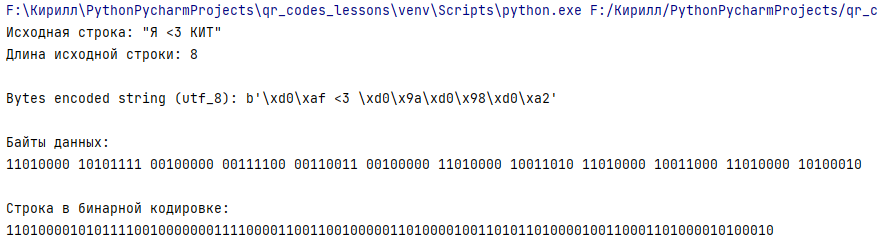


Рисунок 21 – Результат байтового кодирования в реализованном приложении

Режим кандзи позволяет эффективно кодировать знаки кандзи (китайские и японские иероглифы) в соответствии с системой Shift JIS. Каждый двухбайтовый знак кодируется в 13-битовое кодовое слово.

Символы QR Code также поддерживают смешанный режим кодирования. То есть информация может быть представлена в виде нескольких последовательностей дынных и каждая такая последовательность представлена в любом из режимов кодирования.

## **2.4 Добавление служебной информации и заполнение**

Служебная информация содержит в себе индикаторы режима кодирования информации (таблица 2) и числа знаков (таблица 3).

Таблица – Индикаторы режима для QR Code

|  |  |
| --- | --- |
| Режим | Символы QR Code |
| Числовой | 0001 |
| Алфавитно-цифровой | 0010 |
| Байтовый | 0100 |
| Кандзи | 1000 |

Таблица – Число битов в индикаторе числа знаков для QR Code

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Версия | Числовой режим | Алфавитно-цифровой режим | Байтовый режим | Режим кандзи |
| 1-9 | 10 | 9 | 8 | 8 |
| 10-26 | 12 | 11 | 16 | 10 |
| 27-40 | 14 | 13 | 16 | 12 |

Для определения количества бит в индикаторе числа знаков на данном этапе необходимо определить версию QR кода и уровень исправления ошибок.

Версия QR Code зависит от количества бит кодируемых данных. Для обеспечения минимального количества бит-заполнителей, необходимо выбирать версию, емкость данных которой максимально близка к объему закодированных данных (таблица 4).

Таблица ­– Емкость входных данных для QR Code (часть таблицы)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Вер- сия | Уровень исправления ошибок | Число кодовых слов данных | Число битов для данных | Емкость данных для разных режимов | | | |
| Цифровой | Алфавитно- цифровой | Байтовый | Кандзи |
| 1 | L | 19 | 152 | 41 | 25 | 17 | 10 |
|  | M | 16 | 128 | 34 | 20 | 14 | 8 |
|  | Q | 13 | 104 | 27 | 16 | 11 | 7 |
|  | H | 9 | 72 | 17 | 10 | 7 | 4 |
| 2 | L | 34 | 272 | 77 | 47 | 32 | 20 |
|  | M | 28 | 224 | 63 | 38 | 26 | 16 |
|  | Q | 22 | 176 | 48 | 29 | 20 | 12 |
|  | H | 16 | 128 | 34 | 20 | 14 | 8 |
| 3 | L | 55 | 440 | 127 | 77 | 53 | 32 |
|  | M | 44 | 352 | 101 | 61 | 42 | 26 |
|  | Q | 34 | 272 | 77 | 47 | 32 | 20 |
|  | H | 26 | 208 | 58 | 35 | 24 | 15 |
| 4 | L | 80 | 640 | 187 | 114 | 78 | 48 |
|  | M | 64 | 512 | 149 | 90 | 62 | 38 |
|  | Q | 48 | 384 | 111 | 67 | 46 | 28 |
|  | H | 36 | 288 | 82 | 50 | 34 | 21 |

После определения индикатора режима и индикатора числа данных, необходимо добавить их в начало закодированного потока данных.

В реализованном программном продукте был выбран байтовый режим кодирования. Ему соответствует индикатор «0100». Количество байт данных определяется с помощью встроенных методов len(), примененных к строке, полученной после кодирования методом binary\_encode() (рисунок 22). Также было принято решение о генерации QR кодов только с уровнем исправления ошибок H.

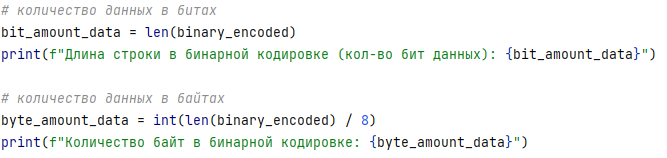


Рисунок 22 – Определение количества данных в битах и байтах в реализованном приложении

Выбор необходимой версии QR кода и определение количества бит в индикаторе числа данных в реализованном приложении происходит при помощи методов get\_version() и get\_length\_amount\_data\_byte\_code() соответственно (рисунок 23). Метод get\_version() работает с заранее заданным списком max\_data\_amount\_h\_corr, который содержит максимальное количество бит информации для каждой версии QR кода. Второй метод является простой проверкой номера версии и возвращает значение равное 8 (в случае, когда номер версии меньше 10) или равное 16 (если номер версии больше либо равен 10).

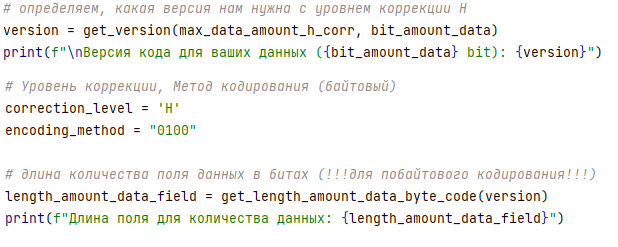


Рисунок 23 – Методы определения версии и длины индикатора числа данных в реализованном приложении

В случае, если длина полученного потока бит не кратна 8, необходимо добавить незначащие нули в конец потока таким образом, чтобы его длина стала кратна 8. Например, поток имеет вид: <последовательность бит, длина которой кратна 8> 100110101001. Так как остаток от деления длины потока на 8 равен 4, необходимо добавить 4 незначащих нуля в конец этого потока: <последовательность бит, длина которой кратна 8> 1001101010010000.

Если сообщение не заполняет целиком емкость символа соответствующей версии и уровня исправления ошибок в соответствии с таблицей 4, он должен быть расширен путем поочередного добавления к его окончанию кодовых слов-заполнителей 11101100 и 00010001.

В реализованном программном продукте дополнение данных происходит при помощи метода fill\_data(), код которого изображен на рисунке 24.

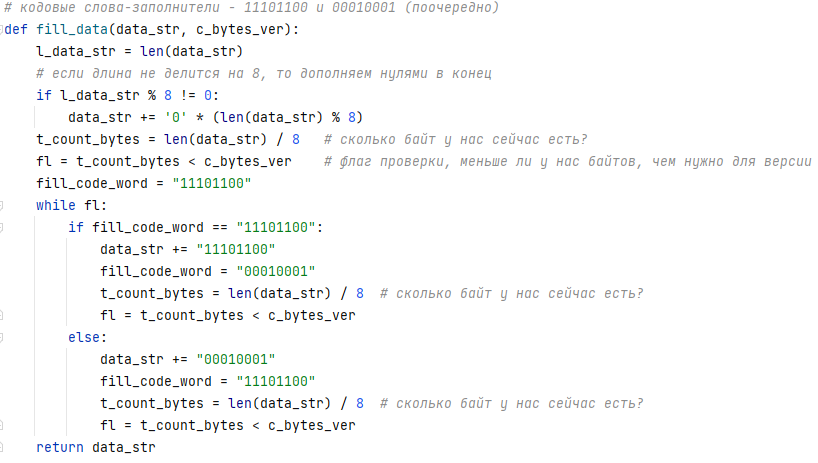


Рисунок 24 – Код метода fill\_data() в реализованном приложении

## **2.5 Разделение информации на блоки**

В зависимости от версии и уровня исправления ошибок, полученная на предыдущем этапе последовательность должна быть поделена на блоки для реализации алгоритма Рида-Соломона и создания байтов исправления ошибок (таблица 5). Если количество блоков равно одному, то разделения не происходит и последовательность данных принимается в исходном виде.

Таблица – Количество блоков для различных версий QR Code

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Версия  Уро-  вень  коррекции | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| L | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 4 |
| M | 1 | 1 | 1 | 2 | 2 | 4 | 4 | 4 | 5 | 5 |
| Q | 1 | 1 | 2 | 2 | 4 | 4 | 6 | 6 | 8 | 8 |
| H | 1 | 1 | 2 | 4 | 4 | 4 | 5 | 6 | 8 | 8 |
| Версия  Уро-  вень  коррекции | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| L | 4 | 4 | 4 | 4 | 6 | 6 | 6 | 6 | 7 | 8 |
| M | 5 | 8 | 9 | 9 | 10 | 10 | 11 | 13 | 14 | 16 |
| Q | 8 | 10 | 12 | 16 | 12 | 17 | 16 | 18 | 21 | 20 |
| H | 11 | 11 | 16 | 16 | 18 | 16 | 19 | 21 | 25 | 25 |
| Версия  Уро-  вень  коррекции | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| L | 8 | 9 | 9 | 10 | 12 | 12 | 12 | 13 | 14 | 15 |
| M | 17 | 17 | 18 | 20 | 21 | 23 | 25 | 26 | 28 | 29 |
| Q | 23 | 23 | 25 | 27 | 29 | 34 | 34 | 35 | 38 | 40 |
| H | 25 | 34 | 30 | 32 | 35 | 37 | 40 | 42 | 45 | 48 |
| Версия  Уро-  вень  коррекции | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| L | 16 | 17 | 18 | 19 | 19 | 20 | 21 | 22 | 24 | 25 |
| M | 31 | 33 | 35 | 37 | 38 | 40 | 43 | 45 | 47 | 49 |
| Q | 43 | 45 | 48 | 51 | 53 | 56 | 59 | 62 | 65 | 68 |
| H | 51 | 54 | 57 | 60 | 63 | 66 | 70 | 74 | 77 | 81 |

С целью определения количества байт данных в каждом блоке необходимо разделить общее число байт данных для данной версии и уровня исправления ошибок на количество блоков [1]. Остаток от деления показывает, сколько блоков данных будут дополнены. Важно отметить, что дополняются последние блоки.

В реализованном приложении заранее определен список, содержащий информацию о том, на какое количество блоков необходимо разделить данные для каждой из версий QR Code. Подсчет количества байт данных для каждого блока происходит с помощью простых арифметических операций (рисунок 25).

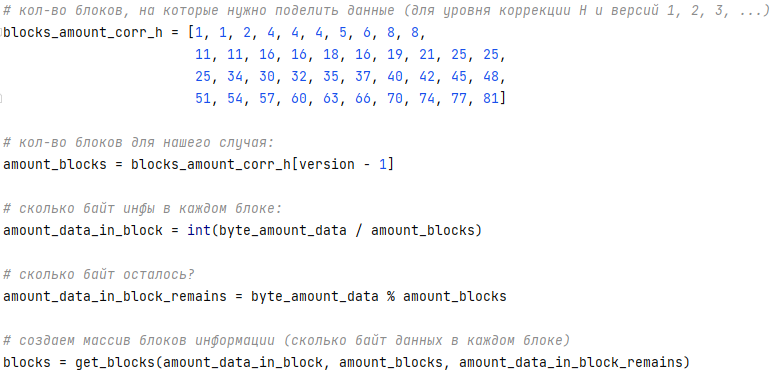


Рисунок 25 – Часть реализованного приложения, разделяющая поток данных на блоки

Например, для версии 8 и уровня коррекции H количество данных — 71 байт. Тогда при выводе полученных программой данных в консоль будет напечатана информация, отображенная на рисунок 26



Рисунок 26 – Результат работы реализованного приложения для 8 версии и уровня коррекции H

Далее блоки заполняются подготовленными ранее байтами данных. Когда очередной блок полностью заполнен, начинается заполнение следующего блока.

В реализованном программном продукте заполнение блоков данными происходит с помощью метода get\_data\_by\_blocks(), результатом работы которого является двумерный список, в котором содержатся заполненные блоки (рисунок 27).

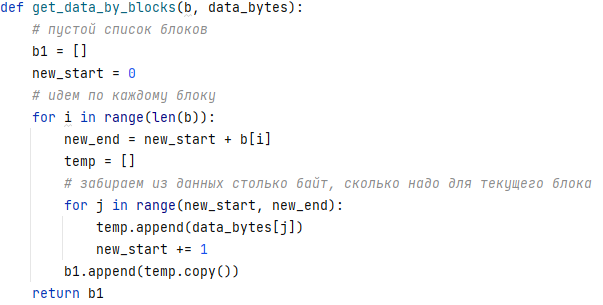


Рисунок 27 – Метод get\_data\_by\_blocks() в реализованном приложении

## **2.6 Создание байтов коррекции**

Кодовые слова исправления ошибок вычисляют отдельно для каждого блока и добавляют к кодовым словам данных [3].

Арифметические операции полиномов для QR Code для вычислений использует побитовый подсчет по модулю 2 и побайтовый подсчет по модулю 100011101. Это поле Галуа 28, где 100011101 соответствует простому минимальному полиному поля *x8+x4+x3+x2*+1 [5].

Кодовые слова данных - коэффициенты членов полинома, начиная с коэффициента при члене со старшей степенью, являющегося первым кодовым словом данных и заканчивая коэффициентом при члене с младшей степенью, являющегося последним кодовым словом данных перед первым кодовым словом исправления ошибок.

Для генерации кодовых слов исправления ошибок в символике QR Code используют 36 различных порождающих полиномов, часть из которых приведена в таблице 6.

Таблица – Порождающие полиномы для кодовых слов исправления ошибок по Риду-Соломону

|  |  |
| --- | --- |
| Число кодовых слов исправления ошибок | Порождающие полиномы |
| 2 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |

Кодовые слова исправления ошибок получают путем деления полинома кодовых слов данных, где каждое кодовое слово является коэффициентом полинома в порядке убывания степени, на порождающий полином. Коэффициенты полинома (остатка от этого деления) являются значениями кодовых слов исправления ошибок [1].

Генерация может быть выполнена с помощью схемы, приведенной на рисунке 28. Регистры от b0 до bk-1 инициализируют нулями. Существуют две стадии генерации кодирования. На первой стадии при положении ключа в нижней позиции кодовые слова данных передаются как на выход, так и на вход схемы. Первая стадия завершается за n синхронизирующих импульсов. На второй стадии (n+1 ... n+k синхронизирующих импульсов), при положении ключа в верхнем положении кодовые слова исправления ошибок εk-1 ... ε0 генерируются путем выдачи значений из всех регистров по порядку с сохранением нулевых данных на входе [4].

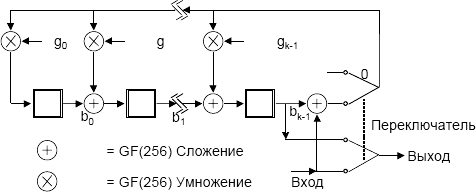


Рисунок 28 – Схема кодирования кодовых слов исправления ошибок

В реализованном программном продукте представленный алгоритм создания байтов коррекции для каждого блока реализован в методе get\_corr\_bytes() (рисунок 29). В качестве аргументов он принимает поделенные на блоки данные, количество байтов коррекции для каждого блока (эта величина заранее известна и явно задана в виде списка), а также генерирующий многочлен (выбирается в зависимости от необходимого количества байтов коррекции). Также для упрощения нахождения байтов исправления ошибок в программе заранее явно заданы поля GF(256) и обратное ему в виде списков. Метод get\_corr\_bytes() использует значения полей для выполнения сложения, показанного в схеме на рисунке 28.

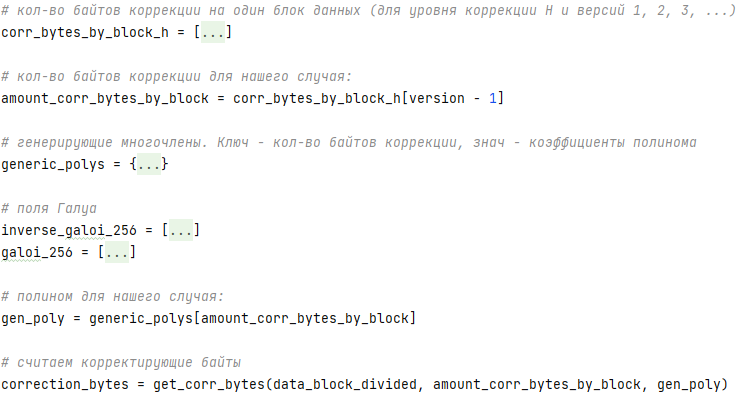


Рисунок 29 – Часть реализованного приложения, генерирующая байты исправления ошибок

## **2.7 Объединение информационных и корректирующих блоков**

На данном этапе имеются блоки данных и соответствующие им байты коррекции. Заключительная последовательность составляется из кодовых слов данных и исправления ошибок следующим образом: блок данных 1, кодовое слово 1; блок данных 2, кодовое слово 1; блок данных 3, кодовое слово 1; ... вплоть до блока данных n-1, последнее кодовое слово; блок данных n, последнее кодовое слово; затем блок исправления ошибок 1, кодовое слово 1, блок исправления ошибок 2, кодовое слово 1 и т.д. до блока исправления ошибок n-1, последнее кодовое слово; блок исправления ошибок n, последнее кодовое слово. Если в текущем блоке уже нет байт, то он пропускается.

Символы QR Code состоят из блоков данных и исправления ошибок, которые полностью заполняют емкость символа. Однако в некоторых версиях QR Code, где число модулей области кодирования не делится на 8 без остатка, к завершенному двоичному потоку может быть добавлено 3, 4 или 7 остаточных битов, чтобы полностью заполнить область кодирования символа.

В реализованном программном продукте объединение блоков данных и байтов исправления ошибок происходит с помощью метода combine\_sys\_data\_corr\_bytes() (рисунок 30). В качестве аргументов данный метод использует байты информации и байты исправления ошибок.

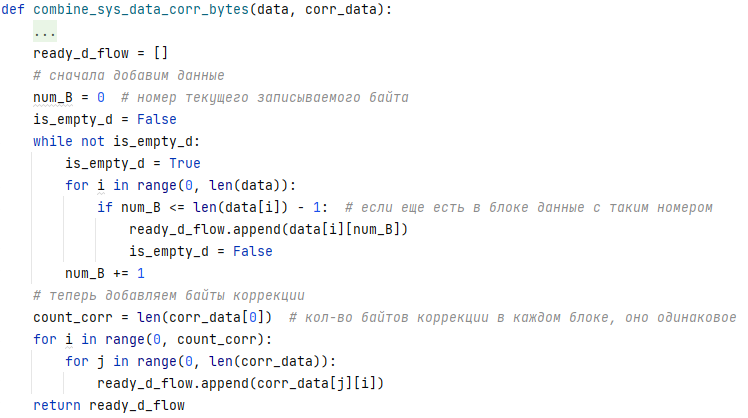


Рисунок 30 – Метод combine\_sys\_data\_corr\_bytes() реализованного программного продукта

## **2.8 Размещение полученных данных в символе QR Code**

Изначально создается пустая квадратная матрица с числом модулей по горизонтали и вертикали согласно используемой версии. Необходимо учесть, что рамка вокруг символа QR кода шириной в 4 модуля должна быть заполнена белыми модулями. Размер символа (в модулях) без учета рамки будет равен сумме последней координаты для соответствующей версии из таблицы 7 и числа 7. Производится сумма именно с числом семь вследствие того, что в стандарте QR Code однозначно определено количество модулей, которые могут быть расположены далее чем последний правый нижний выравнивающий узор.

Таблица – Координаты (столбца/строка) центрального модуля направляющих шаблонов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Версия | Число направляющих шаблонов | Координаты (столбец/строка) центрального модуля | | |
| 1 | 0 | - |  |  |
| 2 | 1 | 6 | 18 |  |
| 3 | 1 | 6 | 22 |  |
| 4 | 1 | 6 | 26 |  |
| 5 | 1 | 6 | 30 |  |
| 6 | 1 | 6 | 34 |  |
| 7 | 6 | 6 | 22 | 38 |

Примечание – в данной таблице представлены координаты только для части версий QR Code.

В реализованном приложении в первую очередь происходит определение координат направляющих шаблонов. Затем происходит подсчет размера холоста в модулях (рисунок 31).

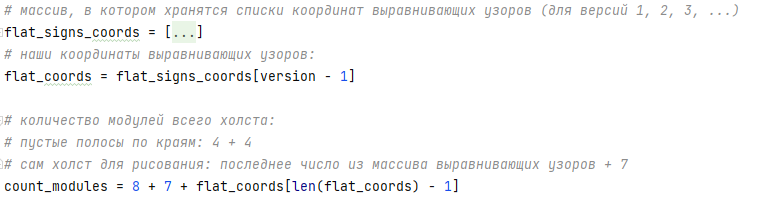


Рисунок 31 – Определение размера холста в реализованном приложении

В первую очередь на символе QR кода располагаются поисковые узоры и полосы синхронизации. Их расположение и вид всегда одинаковы. Определив координаты центральных модулей направляющих узоров из таблицы 7, необходимо расположить их в соответствующих местах символа.

Если версия QR Code выше 7, то на символе необходимо расположить код версии. Часть кодов версий отображена в таблице 8.

Таблица – Коды версий символов QR Code

|  |  |
| --- | --- |
| Версия | Код версии |
| 7 | 000010 011110 100110 |
| 8 | 010001 011100 111000 |
| 9 | 110111 011000 000100 |
| 10 | 101001 111110 000000 |
| 11 | 001111 111010 111100 |
| 12 | 001101 100100 011010 |

На рисунке 32 представлено изображение частей символа QR Code, в которых необходимо располагать код версии.

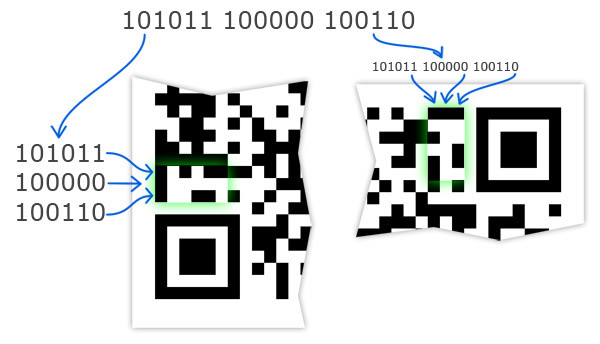


Рисунок 32 – Расположение кода версии 13 на символе QR Code

В реализованном программном продукте заполнение модулей, составляющих функциональные шаблоны, происходит при помощи отдельных методов. Методы color\_str\_matr() и color\_col\_matr() позволяют заполнить рамку из белых модулей, расположенную вокруг символа QR Code. С помощью методов draw\_left\_up\_search(), draw\_right\_up\_search() и draw\_left\_down\_search() происходит заполнение поисковых узоров. Методы draw\_horiz\_sync() и draw\_vert\_sync() заполняют синхронизирующий функциональный шаблон. Метод draw\_flat\_signs() позволяет заполнить все направляющие узоры (если они необходимы для выбранной версии кода).

Все из перечисленных выше методов работают с матрицей, являющейся представлением символа QR кода в виде модулей. В качестве примера, на рисунке 33 представлен код метода заполнения верхнего правого поискового узора.

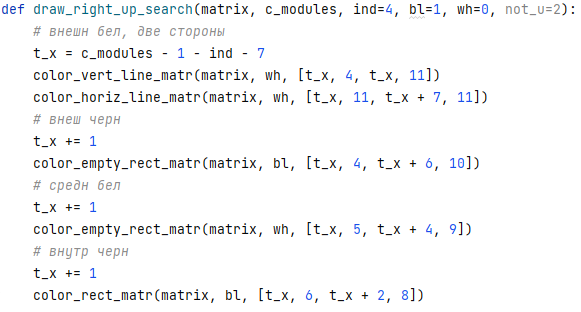


Рисунок 33 – Метод заполнения верхнего правого поискового узора в реализованном приложении

Далее необходимо определить информацию о формате. Это последовательность из 15 бит, 5 из которых – данные, 10 – биты исправления ошибок, вычисляемые по коду БЧХ (15, 5).

Первые два бита данных содержат уровень исправления ошибок символа, обозначенный соответствующим образом (таблица 9).

Таблица – Индикаторы уровня исправления ошибок для символов QR

|  |  |
| --- | --- |
| Уровень исправления ошибок | Двоичный индикатор |
| L | 01 |
| M | 00 |
| Q | 11 |
| H | 10 |

Биты с третьего по пятый содержат указатель шаблона маски данных (таблица 10). Шаблон маски данных формируют путем определения любого модуля в области кодирования (исключая область, сохраненную для информации о формате и информации о версии) как темного, для которого условие является истинным; при условии, что i относится к позиции строки рассматриваемого модуля и j к позиции его столбца, причем (i, j) = (0, 0) для верхнего левого модуля символа (рисунок 34).

Таблица – Условия генерации шаблона маски данных

|  |  |
| --- | --- |
| Указатель шаблона маски данных для символов QR Code | Условие |
| 000 | (*i*+*j*) mod 2=0 |
| 001 | *i* mod 2=0 |
| 010 | *j* mod 3=0 |
| 011 | (*i+j*) mod 3=0 |
| 100 | ((*i* div 2)+(*j* div 3)) mod 2=0 |
| 101 | (*i j*) mod 2+((*i j*) mod 3=0 |
| 110 | ((*i j*) mod 2+(*i j*) mod 3) mod 2=0 |
| 111 | (*i+j*) mod 2+(*i j*) mod 3) mod 2=0 |

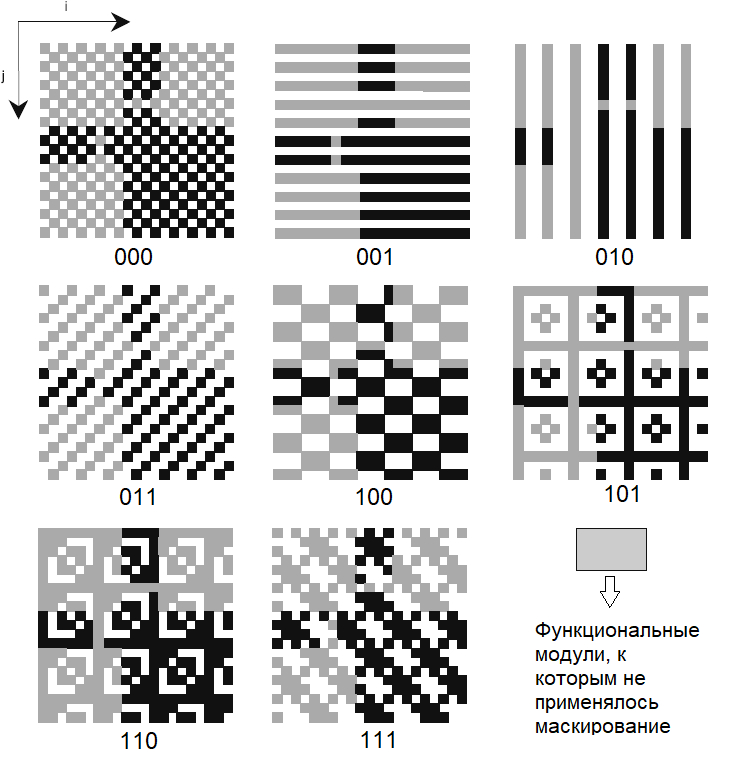


Рисунок 34 – Шаблоны маски данных для символа версии 1

В таблице 11 в качестве примера для уровня коррекции M представлены итоговые битовые последовательности информации о формате.

Таблица – Надлежащие битовые последовательности информации о формате (для уровня коррекции M)

|  |  |  |  |
| --- | --- | --- | --- |
| Последовательность до маскирования | | Последовательность после маскирования (символы QR Code) | |
| Биты данных | Биты исправления ошибок | Двоичная | Шестнад- цатиричная |
| 00000 | 0000000000 | 101010000010010 | 5412 |
| 00001 | 0100110111 | 101000100100101 | 5125 |
| 00010 | 1001101110 | 101111001111100 | 5Е7C |
| 00011 | 1101011001 | 101101101001011 | 5B4B |
| 00100 | 0111101011 | 100010111111001 | 45F9 |
| 00101 | 0011011100 | 100000011001110 | 40CE |
| 00110 | 1110000101 | 100111110010111 | 4F97 |
| 00111 | 1010110010 | 100101010100000 | 4AA0 |

К 15 битам информации о формате применяют операцию XOR с маской 101010000010010, чтобы гарантировать, что никакая комбинация уровня исправления ошибок и указателя шаблона маски данных не имеет в результате 15 нулевых битов.

В результате информация о формате должна быть размещена в области, зарезервированной для этого в символе в соответствии с рисунком 35. Следует обратить внимание, что информация о формате появляется в символе дважды, обеспечивая избыточность, так как ее правильное декодирование необходимо для декодирования завершенного символа. Младший бит информации о формате расположен в модуле с номером 0, а старший бит в модуле с номером 14 (рисунок 35). Модуль в позиции (4V+9, 8), где V - номер версии, должен всегда быть темным и не является частью информации о формате [3].

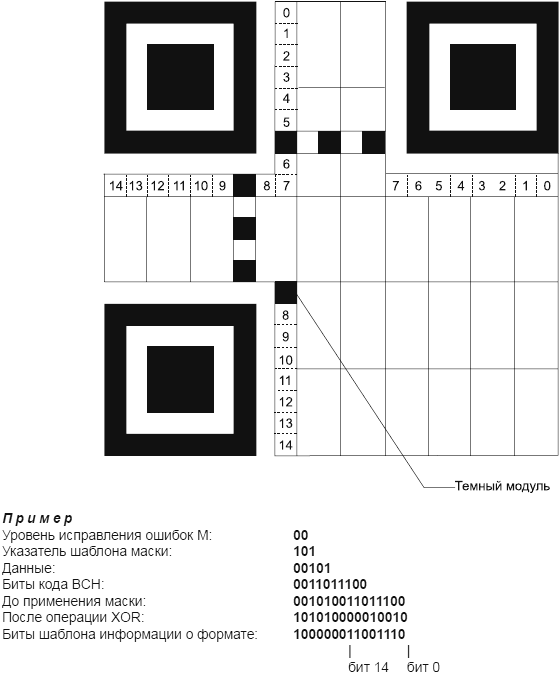


Рисунок 35 – Позиция информации о формате

В реализованном программном продукте добавление информации о версии, уровне исправления ошибок и маске происходит при помощи методов draw\_ver\_code() и draw\_masc\_cor\_code() соответственно (рисунок 36). Вызов метода draw\_ver\_code() происходит только если номер версии больше 6.

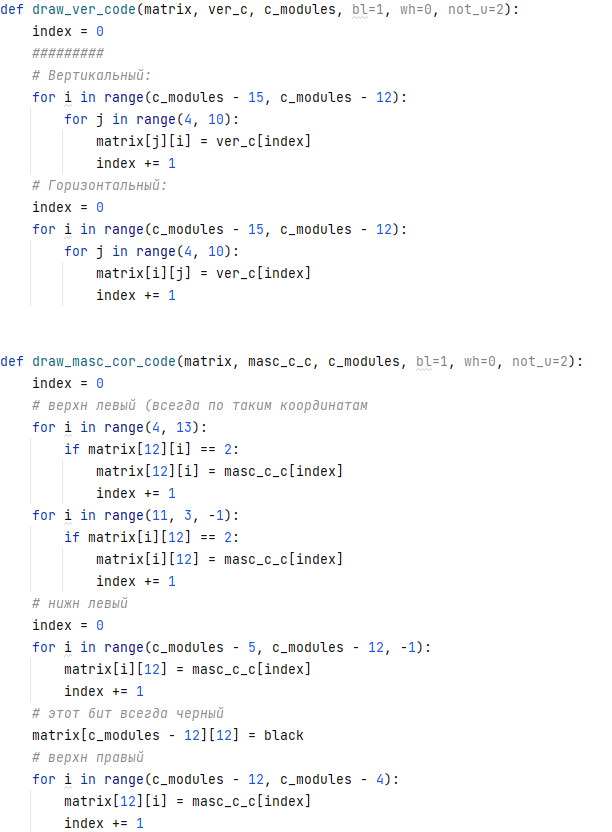


Рисунок 36 – Методы добавления информации о версии, маске и уровне коррекции в реализованном приложении

На последнем этапе происходит добавление последовательности данных и байтов исправления ошибок, полученной на предыдущем этапе, а также маскирование этой информации на символе QR Code. Заполнение начинается из правого нижнего угла символа.

Метод размещения информации в символе заключается в рассмотрении последовательности кодовых слов как единого двоичного потока. Он размещается (начиная со старшего бита) в столбцах шириной два модуля поочередно вверх и вниз и справа налево (рисунок 37). В каждом столбце биты размещают поочередно в правых и левых модулях, перемещая вверх или вниз согласно направлению размещения и пропуская области, занятые функциональными шаблонами, изменяя направление вверху или внизу столбца. Каждый бит всегда должен быть размещен в первую доступную позицию для модуля.

Когда емкость данных символа такая, что не делится без остатка на целое число 8-битовых знаков символа, следует использовать соответствующее число остаточных битов (3, 4 или 7), чтобы заполнить емкость символа. Эти остаточные биты должны всегда иметь значение 0 перед маскированием данных.

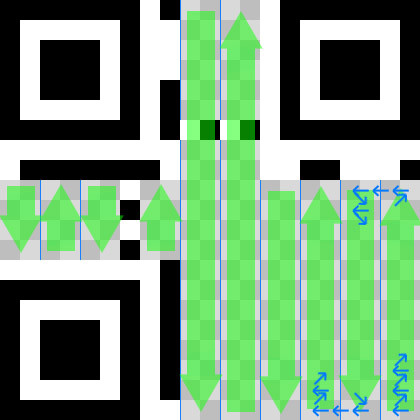


Рисунок 37 – Расположение битов данных в символе QR Code

После расположения данных на символе необходимо применить к модулям, не являющимся функциональными шаблонами, выбранную маску (рисунок 34). Модули, которые совпадают с темными областями маски, необходимо инвертировать.

Применение маски является конечным шагом в размещении информации на символе QR Code.

Добавление данных в реализованном приложении происходит при вызове метода draw\_data() (рисунок 38).

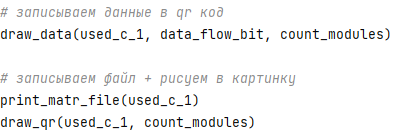


Рисунок 38 – Часть реализованного приложения, заполняющая символ данными и производящая отрисовку символа

На вход в качестве аргумента данный метод принимает единый двоичный поток и заполняет символ QR Code по алгоритму, описанному выше. Одновременно с добавлением очередного бита информации происходит проверка условия маскирования.

Метод draw\_qr() создает файл изображения символа QR Code, определяет размер холста и единичного квадратного модуля а также переносит информацию из полученной матрицы на плоскость изображения (рисунок 39).

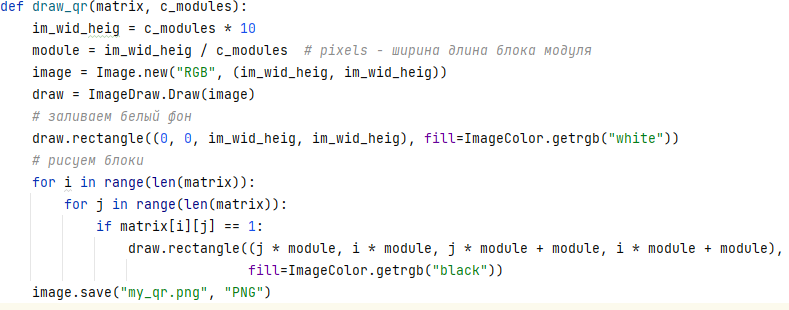


Рисунок 39 – Метод отрисовки QR кода в реализованном приложении

# 3 Результаты работы приложения

В предлагаемой курсовой работе представлено приложение, позволяющее генерировать QR коды на основе байтового кодирования. Разработано на языке программирования Python с использованием библиотеки PIL, позволяющей создавать холст определенного размера (в пикселях), отображать на нем квадратные модули символа QR Code, а также сохранять изображение в необходимом формате.

Ниже приведены примеры работы приложения – сгенерированные QR коды и их расшифровка.

На рисунке 40 представлен QR код текста и его расшифровка с помощью сканера. В тексте была закодирована информация о занятии в университете.

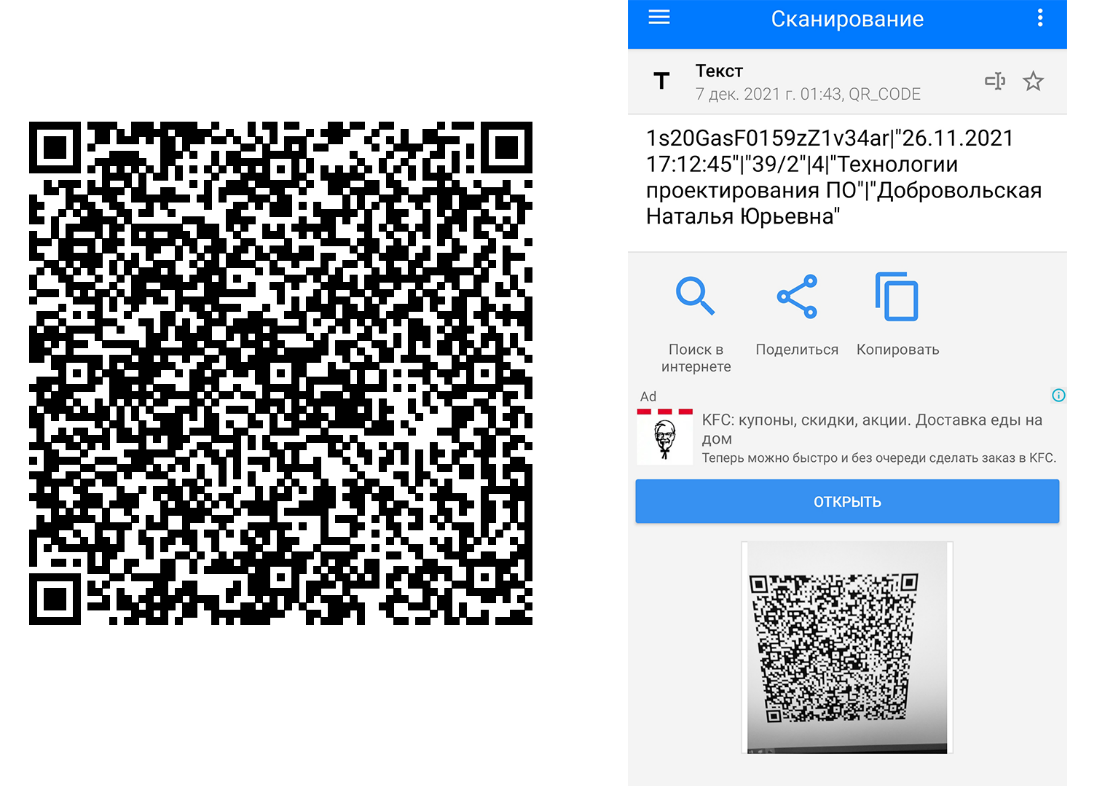


Рисунок 40 – Сгенерированный QR код текста и его расшифровка

На рисунке 41 изображен QR код ссылки на сайт Кубанского Государственного университета и результаты его расшифровки.

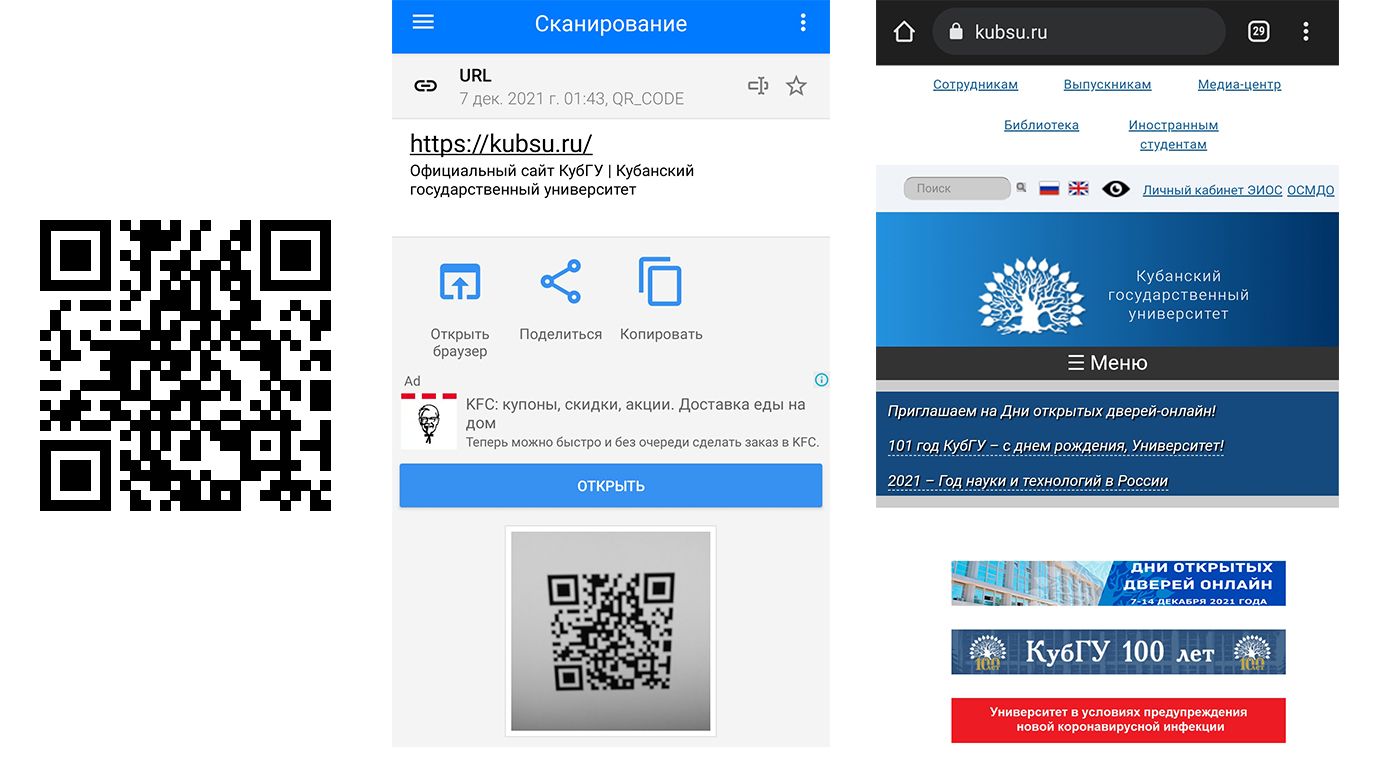


Рисунок 41 – Сгенерированный QR код ссылки и его расшифровка

На рисунке 42 представлен QR код электронной почты с предзаполненным электронным сообщением, а также результат его расшифровки.

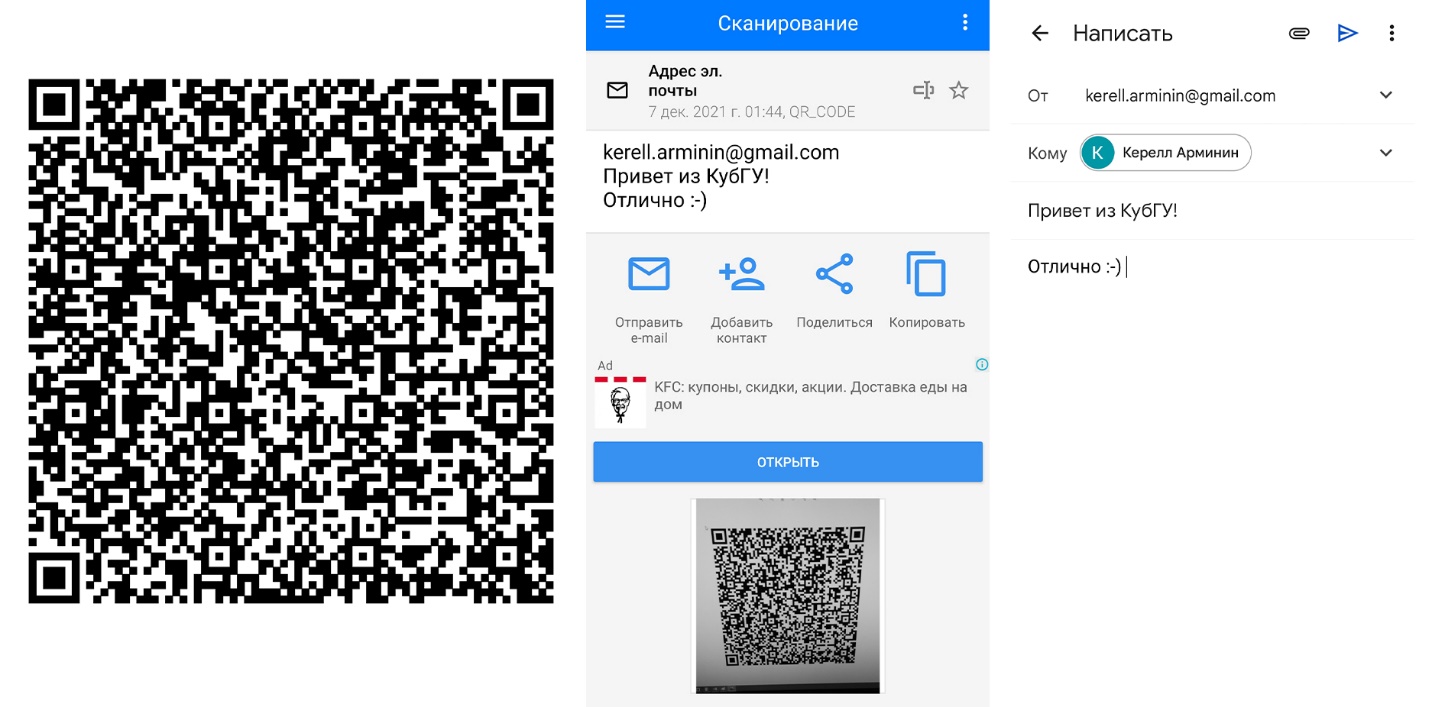


Рисунок 42 – Сгенерированный QR код электронного письма и его расшифровка

На рисунке 43 изображен QR код номера телефона и его расшифровка.

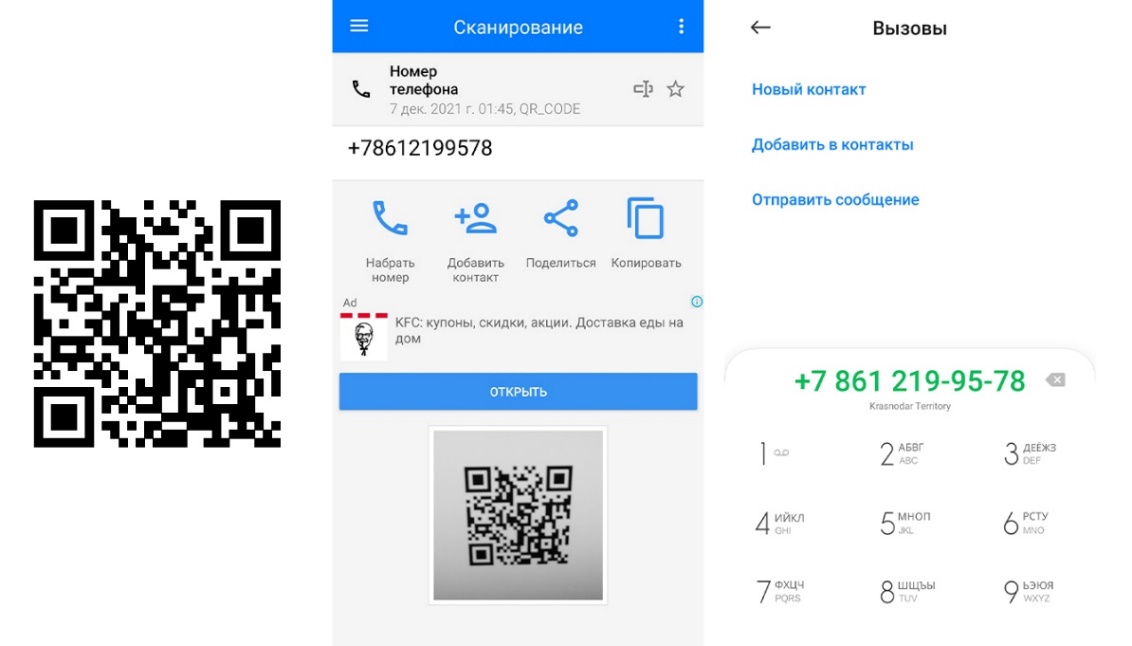


Рисунок 43 – Сгенерированный QR код номера телефона и его расшифровка

На рисунке 44 представлен QR код предзаполненного текстового сообщения на определенный номер телефона и его расшифровка.

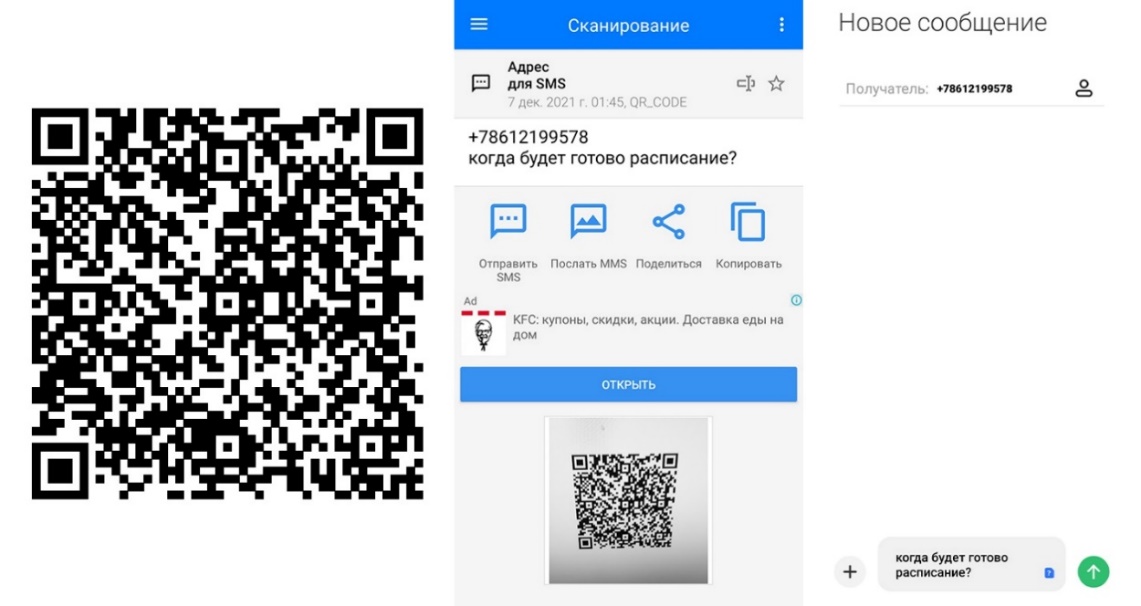


Рисунок 44 – Сгенерированный QR СМС-сообщения и его расшифровка

На рисунке 45 изображен QR код сети Wi-Fi и его расшифровка. Также на некоторых смартфонах есть возможность сканирования QR кодов беспроводных сетей напрямую из меню настроек.

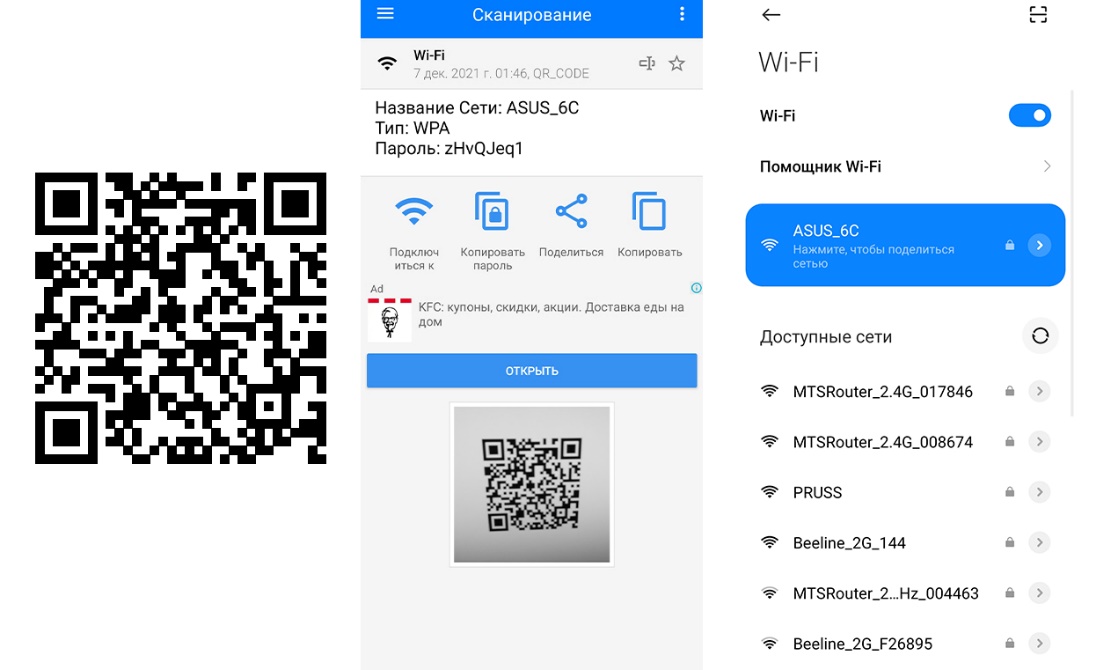


Рисунок 45 – Сгенерированный QR код сети Wi-Fi и его расшифровка

# ЗАКЛЮЧЕНИЕ

Цель курсовой работы – реализовать программу, генерирующую QR code заданного текста – достигнута.

В результате выполнения курсовой работы изучена структура QR code, алгоритмы кодирования информации, а также алгоритм Рида-Соломона для создания байтов коррекции.

В работе выполнен реферативный обзор нескольких сервисов, позволяющих генерировать QR code различных видов, выделены их достоинства и недостатки, произведено тестирование QR кодов, сгенерированных с помощью этих сервисов.

В практической части курсовой работы реализовано приложение на языке программирования Python с использованием модуля PIL для создания изображения QR кода, позволяющее получить QR code текста, ссылки, номера мобильного телефона, СМС-сообщения, электронного письма, сети Wi-Fi. Также произведено сканирование созданных в реализованном программном продукте QR кодов

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Алгоритм генерации QR-кода: [электронный ресурс]. – URL: https://habr.com/ru/post/172525/ (05.10.2021).

2 Генератор QR кодов: [электронный ресурс] – URL: http://qrcoder.ru (07.10.2021).

3 ГОСТ Р ИСО/МЭК 18004-2015. Информационные технологии. Технологии автоматической идентификации и сбора данных. Спецификация символики штрихового кода QR Code. Дата введения 2016-02-01.

4 Математические основы теории помехоустойчивого кодирования. Курс лекций: [электронный ресурс]. – URL: http://www.opds.spbsut.ru/data/\_uploaded/mu/motpuk-lect-08.pdf (05.10.2021).

5 Постников, М.М. Теория Галуа / М.М. Постников. – М.: «Факториал Пресс», 2003. – 304 с. – ISBN 5-88688-063-1.

6 Создать QR код онлайн: [электронный ресурс] – URL: https://qrcod.ru (07.10.2021).

7 QR Генератор – создание дизайнерских QR кодов: [электронный ресурс]. – URL: https://qrcode.website (05.10.2021).