МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра Информационных технологий и систем безопасности

# ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

На тему «Разработка лабораторного практикума на основе сети Фейстеля»

**Исполнитель _____Балицкий Георгий Викторович_____**

(фамилия, имя, отчество)

**Руководитель _____кандидат военных наук_____**

(ученая степень, ученое звание)

**_____Козлов Юрий Викторович_____**

(фамилия, имя, отчество)

**«К защите допускаю»**

**Заведующий кафедрой _____**

(подпись)

**_____доктор технических наук доцент_____**

(ученая степень, ученое звание)

**_____Лепешкин Олег Михайлович_____**

(фамилия, имя, отчество)

«___»_____20__г.

Санкт–Петербург 2026

МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**федеральное государственное бюджетное образовательное учреждение высшего образования**

**«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»**

**Кафедра** Информационных технологий и систем безопасности

«УТВЕРЖДАЮ»

Заведующий кафедрой

_____

Лепешкин Олег Михайлович

(подпись) (фамилия, имя, отчество)

«___»_____20___года

## Задание

## на выпускную квалификационную работу

студенту Балицкому Георгию Викторовичу

(фамилия, имя, отчество)

1. **Тема** “ Разработка лабораторного практикума на основе сети Фейстеля ”
закреплена приказом ректора Университета от « » года,№_____
2. **Срок сдачи законченной работы «____» _____20___года**
3. **Исходные данные к выпускной квалификационной работе:**

**Перечень вопросов, подлежащих разработке (краткое содержание работы:** Среда разработки IDE Visual Studio 2019, язык программирования C#. Технология разработки модуля системы FNSC – Microsoft.Windows Forms (.NET Framework 4.7.2), т. е. для построения пользовательского интерфейса применяются стандартные компоненты, определенные пространством имен Microsoft.Windows.Forms. Разработано программное обеспечение с методическим материалом к нему.

Введение. Обоснование выбора темы, актуальность, цели и задачи ВКР

Глава 1. Роль и место криптографической защиты информации в обеспечении информационной безопасности

(наименование главы)

Глава 2. Разработка программного обеспечения лабораторного практикума на основе сети Фейстеля

(наименование главы)

Глава 3. Разработка методического обеспечения лабораторного практикума на основе сети Фейстеля

(наименование главы)

Заключение. Выводы по работе в целом. Оценка степени решения поставленных задач. Практические рекомендации.

4. **Перечень материалов, представляемых к защите:**

–Пояснительная записка;

- Листинг программы

**7. Дата выдачи задания: « »_____**

_____

**Руководитель выпускной квалификационной работы:**

кандидат военных наук Козлов Юрий Викторович

(должность, ученая степень, ученое звание, фамилия, имя, отчество) (подпись)

Задание принял к исполнению « » 2025 года

Студент: Балицкий Георгий Викторович

РЕФЕРАТ

Дипломная работа: 67 с., 26 рис., 4 табл., 20 источников литературы.

КРИПТОГРАФИЧЕСКИЙ АНАЛИЗ СЕТИ ФЕЙСТЕЛЯ В БЛОЧНЫХ ШИФРАХ, РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА, МЕТОДОЧЕСКОЕ ПОСОБИЕ.

Объект исследования: является лабораторный практикум для выполнения заданий по дисциплине «Методы и средства криптографической защиты информации» на основе сети Фейстеля.

Предметом исследования: является программный комплекс для выполнения практических заданий лабораторного практикума

Цель работы: разработка программного обеспечения для выполнения практических заданий лабораторного практикума на основе сети Фейстеля

В дипломной работе проводится анализ блочных шифров их структуризация для дальнейшей разработки программного обеспечения для выполнения практических заданий лабораторного практикума на основе сети Фейстеля

Разработана программа и методическое пособие для выполнения практики лабораторного практикума на основе сети Фейстеля. На основе языка C# в среде разработки IDE Visual Studio 2019.

ОГЛАВЛЕНИЕ

# ВВЕДЕНИЕ

Проблема защиты информации в настоящее время очень актуальна, поскольку сейчас широкое развитие получила информатизация и «оцифровка» практически всех сфер деятельности человека, в том числе и телекоммуникационные системы, и глобальные информационные системы, посредством которых зачастую необходимо передавать важную и закрытую информацию. Перехват ценной информации, проходящей по таким системам, может иметь различного рода негативные последствия. В итоге понятно, что вопросам надежного шифрования данных необходимо уделять достаточно внимания для того, чтобы используемые алгоритмы обладали достаточной криптографической устойчивостью. Таким образом, крайне важно наличие программной поддержки методических комплексов контроля и обучения современным методам шифрования.

Объектом исследования настоящей работы является лабораторный практикум для выполнения заданий по дисциплине «Методы и средства криптографической защиты информации» на основе сети Фейстеля.

Предметом исследования является программный комплекс для выполнения практических заданий лабораторного практикума по дисциплине «Методы и средства криптографической защиты информации» на основе сети Фейстеля.

Цель работы – разработка программного обеспечения для выполнения практических заданий лабораторного практикума на основе сети Фейстеля.

Для достижения намеченной цели в работе поставлены следующие задачи:

1) Обосновать тему лабораторного практикума, привести теоретический обзор и дать основные понятия блочным шифрам и сети Фейстеля, провести сравнительный анализ алгоритмов блочного шифрования и выбрать один из них для реализации в практикуме.

2) Разработать программный комплекс для выполнения практических заданий лабораторного практикума по дисциплине «Методы и средства криптографической защиты информации» на основе сети Фейстеля.

3) Разработать методический комплекс (пособие) для выполнения лабораторного практикума по дисциплине «Методы и средства криптографической защиты информации» с использованием функций разработанного программного обеспечения.

Код / обозначение проекта разрабатываемого программного обеспечения – FNSC (Feistel Network Studying Complex). Для разработки проекта FNSC использован следующий стек технологий:

- язык программирования высокого уровня C# 9.0;
- фреймворк .NET 4.7.2;
- среда разработки IDE Visual Studio 2019;
- CASE-система Enterprise Architect 15.0;
- СУБД – не требовалось.

# 1.ГЛАВА. ОБОСНОВАНИЕ ВЫБОРА ТЕМЫ ЛАБОРАТОРНОГО ПРАКТИКУМА ПО ДИСЦИПЛИНЕ «МЕТОДЫ И СРЕДСТВА КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ ИНФОРМАЦИИ»

## 1.1 АНАЛИЗ РОЛИ И МЕСТА КРИПТОГРАФИЧЕСКИХ СРЕДСТВ В ОБЕСПЕЧЕНИИ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

В современном мире все больше процессов подвергается массовой цифровизации. Большинство сфер деятельности уже невозможно представить себе без использования средств информационных технологий. Информация стала ключевым атрибутом всех процессов человеческой деятельности. Взаимодействие людей с помощью информационных потоках в различных прикладных системах справедливо породило проблему обеспечения информационной безопасности. Информация, которая используется во взаимодействиях, хранится в базах данных, и, если прикладные системы имеют доступ к этой информации, то существуют каналы и способы получения доступа к этой информации сторонним системам. Чтобы защитить доступ к информации, возникла целая прикладная отрасль информатики – обеспечение защиты информации. В целом информационная безопасность – это стратегия, основанная на следующих задачах [1]:

– гарантия конфиденциальности информации (секретная информация не должна ни коим образом попасть тем, кому она не предназначена);

– обеспечение целостности (гарантия того, что вся информация дойдет до адресата в полном объеме и в оригинальном виде);

– реализация доступности (обеспечение доступа к информации легитимному пользователю в соответствии с установленным режимом доступа и правами пользователя).

Защита информации является прикладной реализацией стратегии информационной безопасности, она включает комплекс средств, мер и действий, направленных на реализацию целей и задач информационной безопасности конкретного объекта / процесса. Система защиты информации может включать в себя:

– системы контроля и управления доступом (СКУД, биометрические устройства, политики информационной безопасности, многофакторная аутентификация и т. д.);

– аппаратное обеспечение (брандмауэры, сетевые экраны);

– агенты (процессы, работающие в фоновом режиме: антивирусные программы, сканнеры вредоносных процессов и т. д.);

– системы DLP для предотвращения утечек [2];

– SIEM-системы, позволяющие управлять инцидентами информационной безопасности [3];

– средства, встроенные в прикладное программное обеспечение (алгоритмы, методы).

Последний аспект защиты информации является ключевым, поскольку именно в нем заложена математическая основа защиты информации – криптографические средства. Криптография – это «наука о методах обеспечения конфиденциальности и аутентичности информации» [4]. Криптография сейчас является неотъемлемой частью информационных процессов, которые происходят в современном информационном обществе. Криптографические алгоритмы позволяют защищать данные и сообщения от потенциальных злоумышленников. В рамках криптографии производится изменение одним из методов (рис. 1.1) оригинальной информации так, чтобы даже в случае перехвата злоумышленник полученную информацию никогда не смог бы восстановить в оригинальном виде.


Рисунок 1.1 – Методы изменения преобразования в криптографии

Измененная информация на стороне легитимного получателя подвергается обратному преобразованию для представления ее в оригинальном виде.

Одним из наиболее надежных методов преобразования данных в криптографии является шифрование – «метод защиты информации от несанкционированного доступа, попытки ее изменения, а также для передачи сообщения через незащищенный канал» [5]. Шифрование позволяет преобразовывать информацию в прямом и обратном направлении при наличии универсального ключа, который создается до процесса шифрования или после него. В любом случае, сообщение может быть расшифровано правильно только тем пользователем, который обладает корректным ключом (рис 1.2).

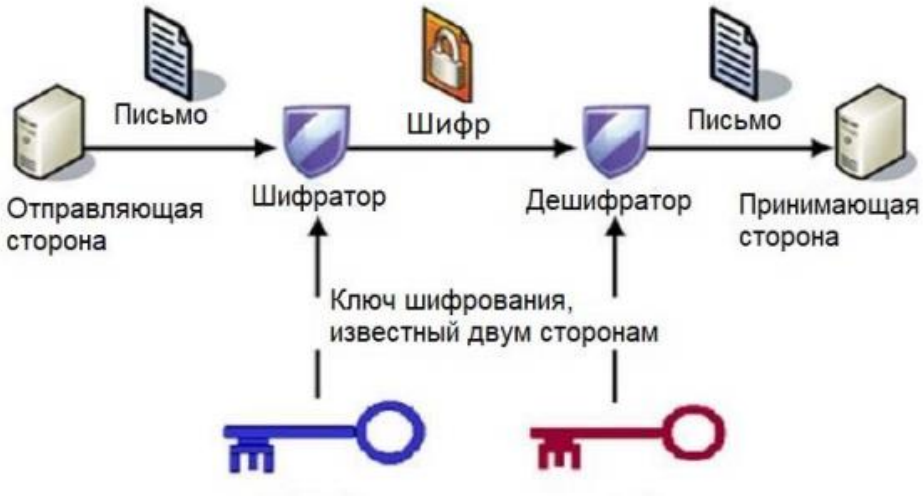


Рисунок 1.2 – Схема шифрования

Все алгоритмы шифрования делятся на блочные и поточные. Блочные шифры – высоконадежные, предназначены для безопасного хранения данных, в то время как поточные шифры более производительны и идеально подходят для работы в реальном времени. Блочное шифрование подразумевает разделение исходного сообщения на блоки равного размера и шифрование с помощью ключа каждого из этих блоков, которые затем складываются в единое зашифрованное сообщение. К блочным алгоритмам шифрования в настоящий момент относятся Blowfish, AES / AES-256, Triple DES, ГОСТ Р 34.12–2015 («Магма», «Кузнечик») и другие. Блочные шифры обладают таким свойствами,

как: установленный фиксированный размер шифроблока, итеративное выполнение, конфузия и диффузия (лавинный эффект), обратимость, симметричность при использовании ключа.

В настоящее время наиболее распространенными в блочном шифровании являются две фундаментальные архитектуры: сеть Фейстеля и SP-сеть. Реализации этих архитектур имеют некоторые отличия в принципе делении блоков, функции раундов, алгоритмах шифрования и дешифрования. Главной отличительной характеристикой среди указанных архитектур является сложность и производительность. Для слабых вычислителей (например, устройства IoT) оптимальным будет применение сети Фейстеля; для, наоборот, высокопроизводительных систем лучше выбрать SP-сеть. Сеть Фейстеля и SP-сеть лежат в основе национального стандарта шифрования – ГОСТ 34.12–2015 [6] (первая – в алгоритме «Магма», вторая – в «Кузнечике»).

## 1.2 СРАВНИТЕЛЬНЫЙ АНАЛИЗ АЛГОРИТМОВ БЛОЧНОГО ШИФРОВАНИЯ

Преобразование сетью Фейстеля.

Преобразования блоков шифротекста сетью Фейстеля выполняется в ходе итераций, т. е. многократно. Каждая итерация называется раундом. Весь текст делится на блоки с фиксированным размером (например, 64 бит) [7]. Преобразования Сетью Фейстеля заключается в разбиении 64-битных блоков на левую и правую половины и применении специальной функции преобразования. Эта функция называется раундовой функцией – это математическая основа алгоритма преобразования. В сети Фейстеля (в отличие от SP-сети к раундовой функции не предъявляется требование обратимости). В сети Фейстеля раундовая функция также называется функцией Фейстеля. В рамках итераций также применяются операции исключающего «ИЛИ» (XOR) и перестановок. Операция XOR здесь обеспечивает свойство конфузии («запутывания», которое разрывает связи между входным текстом и зашифрованным). Таким образом, алгоритм получил название «сеть»,

поскольку он как бы имитирует прохождение входного сигнала через множество узлов с его преобразованием.

Сеть Фейстеля использует ключ в качестве атрибута, обеспечивающего уникальность выполненного преобразования. При этом на основе основного ключа в сети создается набор раундовых ключей, уникальных для каждой итерации. Для генерации раундовых ключей применяется отдельный алгоритм.

Так, для шифрования сообщения применяется прямое преобразование, а для дешифрования – соответственно, обратное преобразование сетью Фейстеля (рис. 1.3) [8].

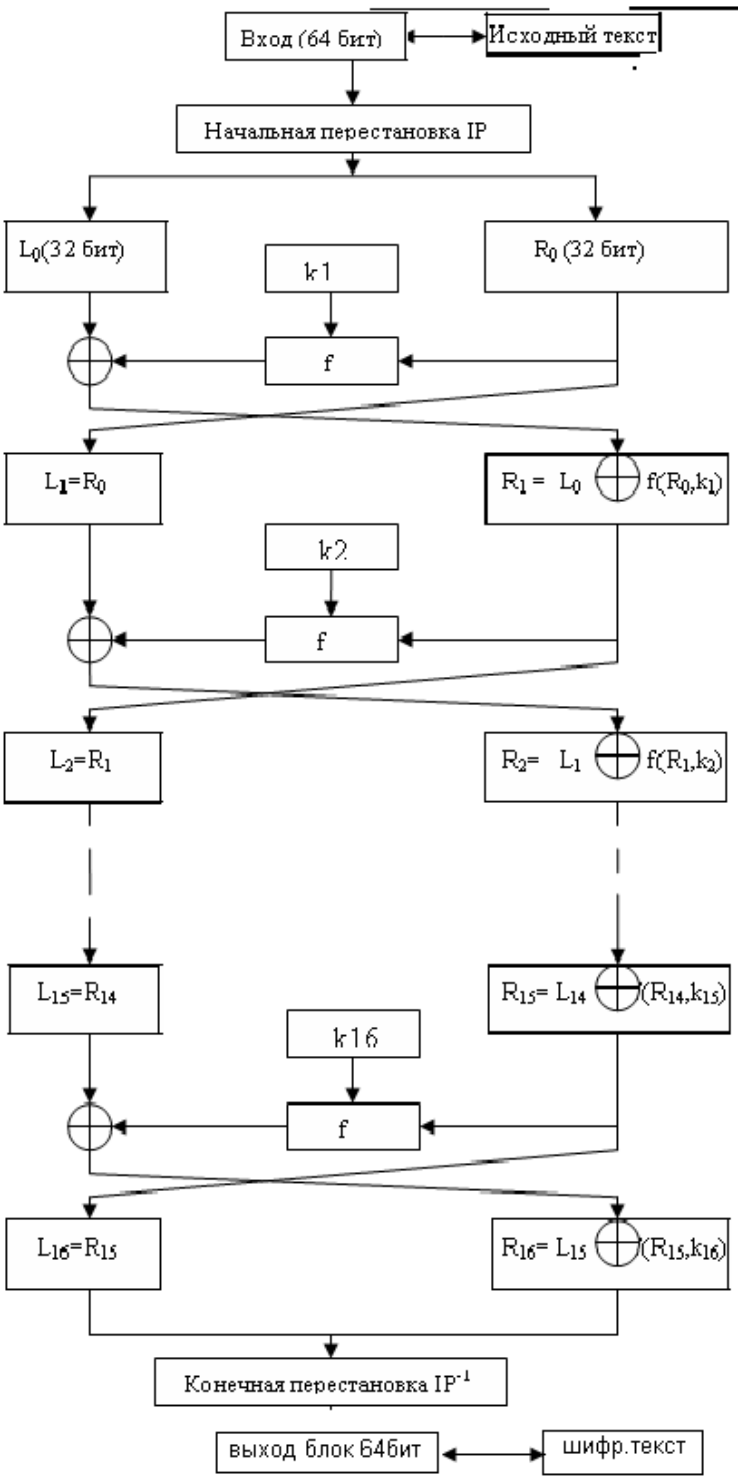


Рисунок 1.3 – Схема алгоритма шифрования, основанного на сети Фейстеля

В приведенном на рисунке 1.3 алгоритме после начальной перестановки над блоком (64 бита) осуществляется 16 раундов процедуры шифрования с помощью функции Фейстеля. Блок (64 бита) разбивается пополам: $L_0, R_0$ (L – левая часть блока, R – правая). Каждая итерация определяется следующими действиями [9]:

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$$

Функции Фейстеля принимает следующие аргументы:

– 32-битный блок $R_{i-1}$;

– 48-битный ключ $K_i$.

Ключи $K_i$ (48 бит) формируются из исходного 56-битного ключа по определенному алгоритму.

Функция Фейстеля предполагает последовательное применение следующих операций и действий:

– расширение блока до 48 бит посредством функции перестановок с дополнениями E;

– применение операции XOR (исключающее ИЛИ) с ключом Ki;

– применение 8-ми S-преобразований;

– завершающая перестановка P.

В результате применения функции расширения E исходный блок длиной 32-бита преобразуется в расширенный блок длиной 48 бит. Это обеспечивается следующими действиями:

– перестановкой бит;

– дублированием некоторых бит.

После расширения 48-битный блок складывается с ключом раунда по модулю 2 (операция исключающее ИЛИ). Затем результат делится на восемь 6-битных блоков:

$$\mathrm{E}(R_{i-1}) \oplus k_i = B_1 B_2 ... B_8$$

К каждому 6-битному блоку применяется процедура S-преобразования, которая уменьшает его до размера в 4 бита (S-таблицы задаются стандартом DES).

Уменьшение происходит по следующей схеме:

– выбирается таблица $S_i$, где i – номер 6-битного блока B;

– выбирается строка в таблице $S_i$ – номер строки равен десятичному числу, полученному из 1-го и последнего битов блока $B_i$;

– выбирается столбец в таблице $S_i$ – номер строки равен десятичному числу, полученному из средних 4-х битов блока $B_i$;

– выбирается число из таблицы $S_i$ на пересечении полученных строки и столбца – его двоичная запись и будет результатом выходного уменьшенного 4-битного блока $B_i$.

Конечный результат функции Фейстеля – 32-битный блок определяется завершающей перестановкой P, которая применяется к полученному уменьшенному 32-битному блоку по схеме:

$$f(R_{i-1}, k_i) = P(B'_1 B'_2 ... B'_8)$$

В конце процедуры шифрования осуществляется конечная перестановка, которая обратна начальной. Конечная перестановка реализуется в соответствии с таблицей, заданной стандартом алгоритма блочного шифрования.

Для преобразований Фейстеля на каждый раунд должен быть сгенерирован свой 48-битный ключ, полученный на основе исходного 56-битного ключа. Для этого 56-битный ключ разбивается на 8 неполных 7-битных блока, к каждому из которых добавляется 1 бит так, чтобы получившийся байт содержал нечетное количество единиц. После этого выполняется перестановка расширенного ключа, которая обходит стороной добавленные биты (8-й, 16-й, 24-й, 32-й, 40-й, 48-й, 56-й, 64-й).

Полученный ключ делится на 2 компоненты $C_0$ и $D_0$. Последующие ключи $C_i$ и $D_i$ формируются посредством левых циклических сдвигов. После сдвига выполняется конечное преобразование ключа раунда посредством конечной перестановки.

При выполнении процедуры дешифрования повторяются все те же действия, что и при шифровании, только в обратном порядке. При этом используется не прямое, а обратное преобразование Фейстеля [9]:

$$R_{i-1} = L_i$$
$$L_{i-1} = R_i \oplus f(L_i, k_i)$$

В процессе дешифрования все таблицы преобразований и перестановок не меняются и применяются в том виде, в каком они были использованы для шифрования (рис. 1.4).

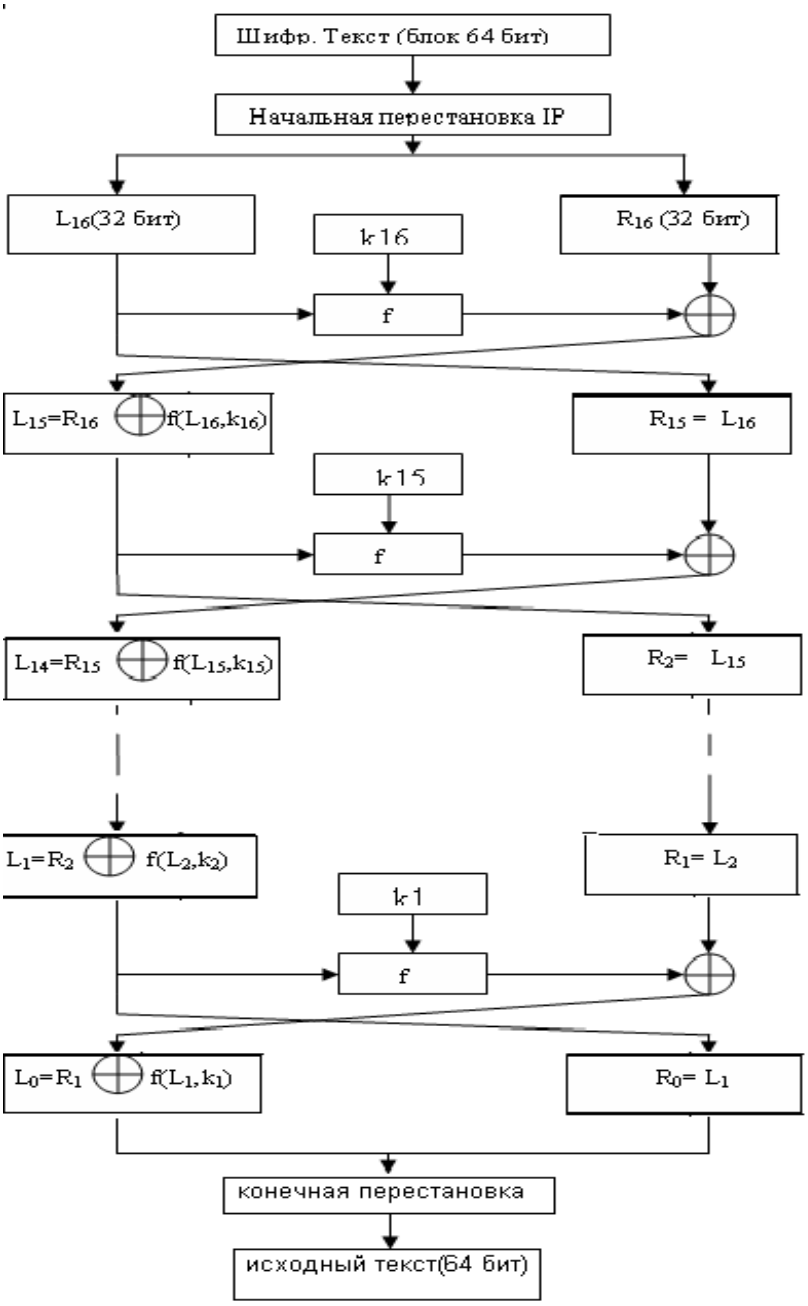


Рисунок 1.4 – Схема алгоритма дешифрования, основанного на сети Фейстеля

На основе преобразований сетью Фейстеля реализованы современные алгоритмы блочного шифрования. Среди них: Blowfish, ГОСТ 34.12–2015 («Магма»), CAST-128, XTEA. В таблице 1.1 приведена сравнительная характеристика этих алгоритмов [10, 11].

Таблица 1.1 – Сравнительная характеристика алгоритмов блочного шифрования

| | **Blowfish** | **ГОСТ 34.12-2015** | **CAST-128** | **XTEA** |
|---|---|---|---|---|
| **Год разработки** | 1993 | 1989 (обновлен: 2015 – «Магма») | 1996 | 1997 |
| **Размер блока** | 64 бит | 64 бит | 64 бит | 64 бит |
| **Длина ключа** | от 32 до 448 бит | 256 бит | от 40 до 128 бит | 128 бит |
| **Число раундов** | 16 | 32 | 12 / 16 | 64 |
| **Структура** | На основе преобразований сети Фейстеля | | | |
| **Криптоскойкость (от 1-5 баллов, 5 – высший балл)** | 4 | 5 | 4 | 3/4 |
| **Производительность (от 1-5 баллов, 5 – высший балл)** | 4 | 5 | 4 | 3/4 |
| **Ограничения** | «Медленная» инициализа-ция ключа | Фиксированные S-блоки | Ограничение ключа 128 бит | Слабый эффект диффузии |
| **Применимость** | Диски, VPN-каналы | КИИ, промышленный стандарт государства | PGP / GNU Privacy Guard (почта, файлы, цифровые подписи) | Устройства IoT, датчики |

Таким образом, алгоритм «Магма», опубликованный в ГОСТ 34.12–2015, является государственным стандартом РФ, включенным в перечень обязательных требований к построению современных систем, обеспечивающих промышленную безопасность и доверенную инфраструктуру государственного сектора [12]. Следовательно, основы и принципы работы сети Фейстеля для

студентов, изучающих основы информационной безопасности, рекомендуется проводить именно с учетом ее применения к алгоритму ГОСТ 34.12–2015: «Магма».

## 1.3 ТРЕБОВАНИЯ К МЕТОДИЧЕСКОМУ ОБЕСПЕЧЕНИЮ ЛАБОРАТОРНОГО ПРАКТИКУМА

Основываясь на теоретическом исследовании алгоритмов блочного шифрования, архитектурно-базирующихся на применении сети Фейстеля, их многообразии и областях применения, можно выделить для лабораторного практикума следующие цели:

– ознакомление с базовыми понятиями блочного шифрования;

– освоение принципов работы сети Фейстеля;

– ознакомление с ключевыми требованиями к криптографическим алгоритмам;

– изучение принципов лавинного эффекта и его реализации в сети Фейстеля;

– изучение основных алгоритмов блочного шифрования, основанных на применении сети Фейстеля;

– получение навыков оценки свойств криптографических алгоритмов, анализа их криптографической стойкости;

– освоение на практике принципов реализации алгоритма блочного шифрования ГОСТ 34.12–2015 («Магма»).

Данный перечень тем является базовым и необходимым для усвоения. Все последующие и более сложные структуры и функции основываются на перечисленных и не могут быть изучены без наличия названных базовых знаний. Таким образом, можно рекомендовать к изучению следующий курс лабораторных работ, охватывающих предложенные темы (таблица 1.2).

Таблица 1.2 – Перечень лабораторных работ

| № | Наименование работы | Цель работы | Сформированные компетенции |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 1 | Основы работы сети Фейстеля | Ознакомиться с базовыми понятиями блочного шифрования, изучить теоретический материал по работе сети Фейстеля и ее применению в алгоритмах шифрования, освоить на практике принципы работы и основные операции, производимые в рамках расчета сети Фейстеля | – знание принципов блочного шифрования и основных алгоритмов<br>– понимание принципов работы сети Фейстеля<br>– умение самостоятельно выполнять базовые операции, предусмотренные в работе сети Фейстеля |
| 2 | Реализация ключевых требований к криптографическим алгоритмам в сети Фейстеля | Ознакомиться с ключевыми требованиями к криптографическим алгоритмам. Изучить принципы лавинного эффекта и его реализации в сети Фейстеля | – знание требований к криптографическим алгоритмам (секретность, криптоскойкость, лавинный эффект, эффективность, корректность)<br>– понимание и умение анализировать стойкость криптографических алгоритмов<br>– практическое освоение способов реализации лавинного эффекта в сети Фейстеля |

Продолжение таблицы 1.2

| **№** | **Наименование работы** | **Цель работы** | **Сформированные компетенции** |
|---|---|---|---|
| 3 | Алгоритмы | Изучить основные | – знание стандартов ГОСТ |

| | | | |
|---|---|---|---|
| | блочного шифрования на основе сети Фейстеля. ГОСТ 34.12–2015 («Магма») | алгоритмы блочного шифрования, основанного на применении сети Фейстеля, их характеристики, достоинства и недостатки, применимость на практике. Освоить на практике принципы алгоритма | 34.12-2015, 34.13-2015 для понимания требований к современным криптографическим алгоритмам<br>– знание областей применения различных криптографических алгоритмов<br>– умение самостоятельно рассчитывать все этапы криптографического алгоритма (на примере алгоритма «Магма») |

Методические рекомендации для выполнения лабораторных работ должны содержать общетеоретические сведения по изучаемой теме, конкретную цель работ, описание порядка выполнения работ на конкретном примере, варианты индивидуальных заданий для самостоятельного выполнения и контрольные вопросы для самопроверки. Таким образом, можно построить структуру методических указаний (таблица 1.3).

Таблица 1.3 – Структура методических указаний

| **№ раздела** | **Наименование раздела (пункта)** | **Описание** |
|---|---|---|
| 1 | Наименование работы | Название работы (из таблицы 1.2) |
| 2 | Цель лабораторной работы | Постановка общей цели лабораторной работы (в соответствии с таблицей 1.2) |
| 3 | Задание на лабораторную работу | Общая для всех вариантов постановка задачи на лабораторную работу |

Продолжение таблицы 1.3

| **№ раздела** | **Наименование раздела (пункта)** | **Описание** |
|---|---|---|

| | | |
|---|---|---|
| 4 | Программно-аппаратное обеспечение | Требования к программному и аппаратному обеспечению, необходимому для выполнения лабораторной работы |
| 5 | Краткие теоретические сведения | Краткие теоретические сведения в соответствии с целью работы |
| 6 | Технология выполнения работы | Раздел, посвященный руководству по работе со специализированным программным обеспечением в ходе выполнения практической части лабораторной работы |
| 7 | Порядок выполнения работы | Общий раздел, содержащий описание действий, которые необходимо выполнить в рамках лабораторной работы |
| 8 | Требования к отчету | Описание разделов, которые требуется включить в отчет по лабораторной работе, с кратким описанием содержания |
| 9 | Варианты индивидуальных заданий для самостоятельного выполнения | Перечень вариантов индивидуальных заданий на выполнение лабораторной работы (12–20 штук) либо указать способ получения варианта |
| 10 | Контрольные вопросы для самопроверки | Перечень общетеоретических вопросов по теме лабораторной работы для самоконтроля |
| 11 | Рекомендуемая литература | Список рекомендуемой литературы к теме лабораторной работы |

Для контроля преподавателем выполненных работ каждый студент по каждой лабораторной работе подготавливает краткий отчет, который должен содержать:

- номер и наименование лабораторной работы;
- цель и задачи лабораторной работы;
- задание в соответствии с индивидуальным номером варианта;

– необходимые схемы (при их наличии);

– входные и выходные данные;

– фиксация хода выполнения работы в виде скриншотов экранов визуализации в специализированном программном обеспечении с описанием шагов выполнения, приведением необходимых расчетов (при их наличии) и полученных результатов;

– общие выводы по проделанной работе.

Для самостоятельного изучения можно порекомендовать студентам факультативно ознакомиться с дополнительными темами. Эти навыки расширят умения и возможности будущих специалистов, сделав их более подготовленными. Кроме того, самостоятельное изучение данных вопросов побудит студентов к обращению к источникам дополнительной литературы и задействует интерес и потенциалы обучения, что должно неизбежно оказать положительный эффект в ходе общего изучения дисциплины.

В качестве технических средств к лабораторным работам могут использоваться личные (при домашнем выполнении) или лабораторные (в классах) персональные компьютеры. В качестве программных средств могут использоваться:

– специализированное программное обеспечение лабораторного практикума (предмет исследоания в настоящей работе);

– редактор Microsoft Word версии 2016 и выше – для написания отчетов по выполненным работам;

– любая CAD-система для построения схем алгоритмов (например, MS Visio или онлайн-редактор draw.io).

# 2.ГЛАВА. РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ЛАБОРАТОРНОГО ПРАКТИКУМА НА ОСНОВЕ СЕТИ ФЕЙСТЕЛЯ

## 2.1 РАЗРАБОТКА ДИАГРАММЫ ПРЕЦЕДЕНТОВ

Разработка программного проекта начинается с уточнения функций [13]. В ходе проработки лабораторных работ можно определить функции программного обеспечения для выполнения лабораторного практикума. Эти функции формализовать и соотнести с категориями пользователей. Это можно сделать с помощью моделирования прецедентов в виде диаграммы вариантов использования, определенной в спецификации языка UML 2.5 [14]. На рис. 2.1 приведена диаграмма вариантов использования приложения для выполнения лабораторного практикума. Рабочее название приложения: Feistel Network Studying Complex, сокращенное название: FNSC.

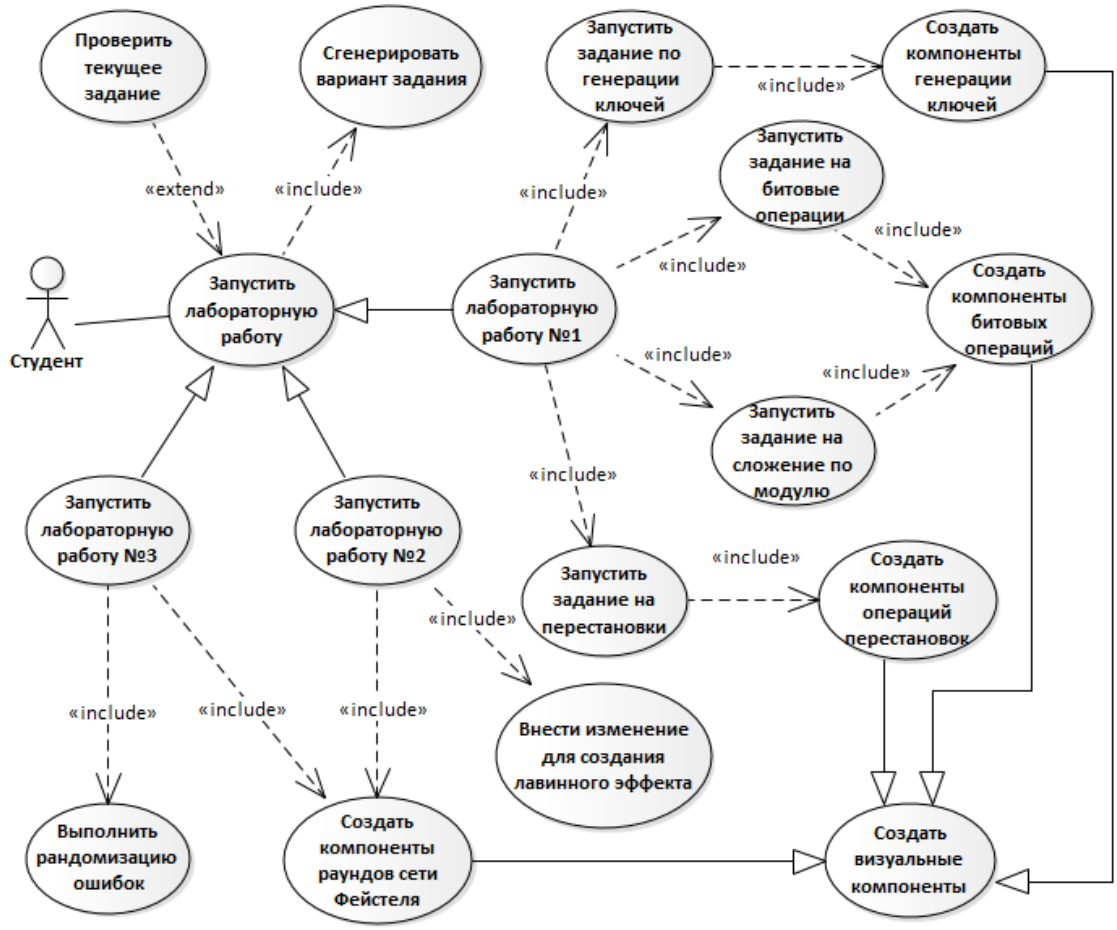

Рисунок 2.1 – Диаграмма вариантов использования приложения для выполнения лабораторного практикума

Диаграмма вариантов использования может рассматриваться как формальная модель пользовательских функциональных требований [15]. Опираясь на данные диаграммы прецедентов, можно определить функционал системы. Таким образом, видно, что основным пользователем приложения для лабораторного практикума будет студент. В рамках функций приложения предлагаются следующие прецеденты:

– запуск лабораторных работ (предусматриваются три лабораторные работы по соответствующим тематикам – см табл. 12);

– запуск заданий для лабораторных работ; в лабораторных работах предлагаются задания разной направленности в рамках общей темы (табл. 2.1).

Таблица 2.1 – Задания лабораторного практикума

| **Номер л. р.** | **Тема л. р.** | **Номер задания** | **Практическое задание, выполняемое в практикуме с помощью FNSC** |
|---|---|---|---|
| 1 | Основы работы сети Фейстеля | 1 | На основе сгенерированного базового ключа (256 бит) сформировать 32 раундовых ключа для сети Фейстеля в соответствии с алгоритмом «Магма» (ГОСТ 34.12–2015) |
| | | 2 | Выполнить в двоичном коде операцию сложения по модулю 32 (одна из операций функции Фейстеля) двух сгенерированных 32-битных блоков |
| | | 3 | Для сгенерированного 32-битного блока выполнить S-перестановку (часть функции Фейстеля, реализующая принцип конфузии) в соответствии с алгоритмом «Магма» (ГОСТ 34.12–2015) |
| | | 4 | Выполнить в двоичном коде базовые |

| | | | |
|---|---|---|---|
| | | | логические операции сети Фейстеля (сложения по модулю 2, сдвиг битов) для двух сгенерированных 32-битных блоков |

Продолжение таблицы 2.1

| **Номер л. р.** | **Тема л. р.** | **Номер задания** | **Практическое задание, выполняемое в практикуме с помощью FNSC** |
|---|---|---|---|
| 2 | Реализация ключевых требований к криптографическим алгоритмам в сети Фейстеля | 1 | Реализовать визуализацию лавинного эффекта. Для двух заданных (сгенерированных) 32-битных подблоков ($L_1$, $R_1$) программа вычисляет и визуализирует результаты всех этапов шифрования сетью Фейстеля (для двух первых раундов): результаты сложения по модулю 2 (с оппозиционным подблоком), 32 (с ключом), перестановки, битовый сдвиг. Затем программа случайным образом меняет один байт в исходных данных (в ключе или любом из подблоков). Задача студента – пересчитать все вычисления и на практике убедиться в лавинном эффекте (при изменении даже маленькой части результат изменяется с лавинообразным характером) |
| 3 | Алгоритмы блочного шифрования на основе сети Фейстеля. ГОСТ 34.12–2015 («Магма») | 1 | Реализовать студентом контроль работы алгоритма. Аналогично работе №2 для двух заданных (сгенерированных) 32-битных подблоков ($L_1$, $R_1$) программа вычисляет и визуализирует результаты всех этапов шифрования сетью |

| | | | |
|---|---|---|---|
| | | | Фейстеля (для двух первых раундов). Затем программа случайно добавляет в расчеты несколько ошибок, которые пользователь должен найти и исправить |

Программа должна предоставлять следующее задание в рамках текущей лабораторной работы только после выполнения предыдущего. Для выполнения заданий в программе должны генерироваться соответствующие визуальные компоненты.

Программа должна позволять генерировать новые исходные данные для каждого задания. При этом должен работать счетчик генераций. Для каждой новой генерации рекомендуется применять штраф к оценке за выполнение работы (например, вычитать 20 баллов из 100 возможных за каждую генерацию после первой).

Также программа должна проверять введенные пользователем результаты и индицировать неправильные ответы (например, красным цветом выделять некорректные блоки, зеленым – корректные). Задание считается выполненным, если все блоки введены правильно. Каждая попытка проверки запоминается счетчиком попыток, по которому рекомендуется также применять штраф к оценке за выполнение работы (например, вычитать 5 баллов из 100 возможных за каждую попытку после первой).

## 2.2 РАЗРАБОТКА СТРУКТУРЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Любая нетривиальная система не проектируется раз и навсегда. С течением времени меняются требования, условия эксплуатации, появляются новые технологии и дополнения к функциональности. Таким образом, каждая такая система должна проектироваться так, чтобы в случае необходимости в нее можно было легко вносить изменения, легко «читать» ее структуру, сопровождать ее и моделировать новые функциональные возможности. Такое

возможно только при применении модульного принципа построения архитектуры системы, при котором создаются пространства имен, функциональные пакеты и т. д. [15]. В функциональные пакеты компонуются семантически схожие модули. При этом интенсивно используются шаблоны (паттерны) проектирования, прошедшие проверку временем и опытом применения в успешных проектах. На данный момент известно несколько десятков таких шаблонов. Одним из основополагающих принципов построения эффективной архитектуры объектно-ориентированных систем является соблюдение принципов SOLID [16].

Так, в рамках проекта FNSC выделяются два базовых функциональных пакета (рис. 2.2):

– Cryptography – пакет, объединяющий в себе всю программную логику в виде математического обеспечения (криптоматематика системы) и основные объекты, с которыми работает эта математика;

– UI – пакет, консолидирующий модуль пользовательского интерфейса (слой представления системы).

–

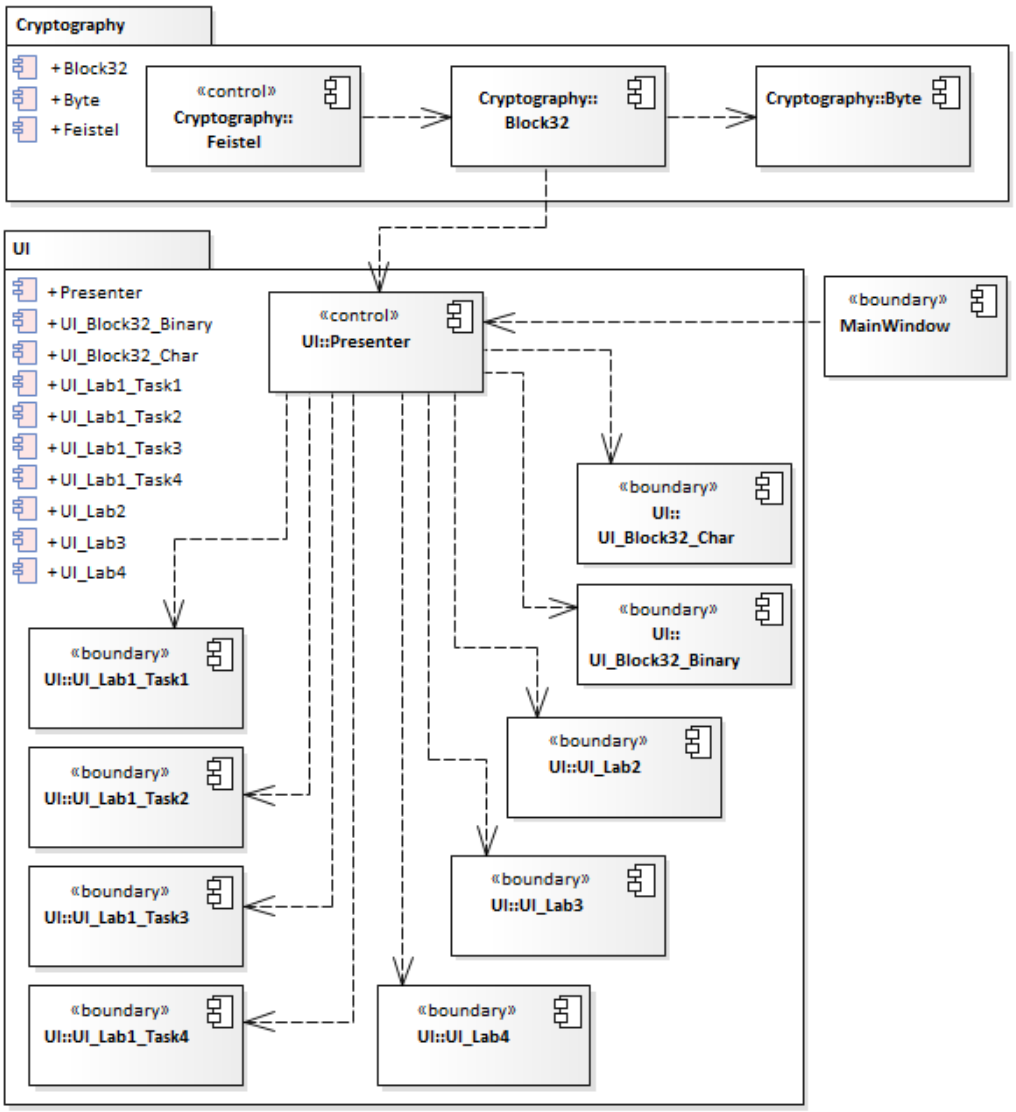


Рисунок 2.2 – Диаграмма компонентов приложения для лабораторного практикума

Пакеты наполняются модулями моделей, контроллеров и представлений в соответствии с их назначением [17]. Так, в рамках пакета Cryptography разработаны следующие основные модули:

– Byte – модуль объекта «байт», используемого в логике шифрования, предоставляющие все основные и вспомогательные операции по изменению, логике и отображению в системе;

– Block32 – модуль, представляющий объект блока, состоящего из 4 байт и являющегося основной операционной единицей в сети Фейстеля, используемой алгоритмом «Магма» (ГОСТ 3412-2015);

– Feistel – основной объект-контроллер, в котором реализована вся криптографическая математика.

Пакет UI содержит следующие программные компоненты (формы представления интерфейса пользователя):

– Presenter – объект-контроллер, позволяющий управлять генерацией компонентов пользовательского интерфейса;

– UI_Block32_Binary – объект, представляющий собой визуализацию 4-байтного блока в виде битов;

– UI_Block32_Char – объект, представляющий собой визуализацию 4-байтного блока в виде байт (символов);

– UI_Lab1_Task1 – компонент визуализации средств и инструментов для выполнения первого задания лабораторной работы №1;

– UI_Lab1_Task2 – компонент визуализации средств и инструментов для выполнения второго задания лабораторной работы №1;

– UI_Lab1_Task3 – компонент визуализации средств и инструментов для выполнения третьего задания лабораторной работы №1;

– UI_Lab1_Task4– компонент визуализации средств и инструментов для выполнения четвертого задания лабораторной работы №1;

– UI_Lab2 – компонент визуализации средств и инструментов для выполнения задания лабораторной работы №2;

– UI_Lab3 – компонент визуализации средств и инструментов для выполнения задания лабораторной работы №3;

– UI_Lab4 – компонент визуализации средств и инструментов для выполнения задания лабораторной работы №4;

– MainWindow – главное окно приложения.

Таким образом, диаграмма компонентов в нотации UML позволяет спроецировать архитектуру объектов системы на их реализацию в виде конечного приложения.

## 2.3. ПРОЕКТИРОВАНИЕ ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ

Технология разработки модуля системы FNSC – Microsoft.Windows Forms (.NET Framework 4.7.2), т. е. для построения пользовательского интерфейса применяются стандартные компоненты, определенные пространством имен Microsoft.Windows.Forms. Само приложение состоит из основного системного окна (формы), в котором размещены необходимые компоненты отображения. При сборке форм использована технология контейнеров. Т. е. все элементы пользовательского интерфейса компонуются в контейнерные элементы следующих типов:

- Panel;
- SplitContainer;
- FlowLayoutPanel;
- TableLayoutPanel.

Контейнер Panel представляет собой пространство формы или ее рабочую поверхность, на которой можно размещать элементы управления. Использование такого контейнера необходимо в случае, когда в одной форме или контейнере уровня выше необходимо использовать переключение видов. В этом случае в родительский контейнер можно реализовать на базе Panel, а содержимое контейнера переключать по заданной логике. Содержимое может быть выполнено отдельными компонентами на базе Microsoft.Windows.Control. В проекте FNSC контейнеры Panel использованы для размещения и организации компонентов интерфейса в едином пространстве с использованием функций группировки.

Контейнер SplitContainer используется для разделения области формы (контейнера уровня выше) на две составляющие горизонтально или вертикально. Основное свойство SplitContainer – наличие «плавающей» границы между разделяемыми областями. Таким образом, пользователь в процессе работы программы может динамически менять соотношение размеров разделяемых областей. Вкладывая элементы SplitContainer друг в друга, можно

организовать что-то типа сетки для организации компонентов форм. Однако для этой задачи уже лучше использовать контейнер TableLayoutPanel [18]. В проекте FNSC контейнеры SplitContainer использованы для разделения представлений в процессе выполнения заданий лабораторного практикума.

Контейнер FlowLayoutPanel используется для автоматической организации динамически добавляемых компонентов форм – его использование удобно, когда не требуется заботиться о виде организации компонентов, но задача стоит только в наполнении контейнера необходимыми объектами [19]. В проекте FNSC контейнеры FlowLayoutPanel использованы для наполнения панелей, представляющих инструменты и компоненты для выполнения заданий лабораторного практикума.

TableLayoutPanel применяется для упорядоченной организации элементов пользовательского интерфейса в виде таблицы / сетки с поддержкой объединения колонок / строк. Этот компонент позволяет организовать эргономичную компоновку формы, содержащей множество групп компонентов, при этом позволяет задать размеры или пропорции ячеек и автоматически подстраивать размеры групп в соответствии с размерами формы. В проекте FNSC контейнеры TableLayoutPanel использованы для представления и визуализации 32-битных блоков в двоичном виде. Компонент может заполняться динамически посредством соответствующих инструкций, записанных на программном коде. Для этого:

– создаются экземпляры компонентов, подлежащих размещению в контейнере;

– в контейнере создается новая строка / столбец;

– экземпляры размещаются в контейнере;

– посредством метода SetRow(Control, int) / SetColumn(Control, int) [20] задается размещение экземпляра в сетке контейнера, здесь: Control – размещаемый экземпляр, int – соответствующий номер строки / столбца.

Описанные принципы контейнерной сборки компонентов пользовательского интерфейса позволяют:

– организовать функции подсистем в одном контейнере по типу переключения страниц на рабочем столе пользователя;

– выполнить компоновку различных элементов на страницах пользователя;

– организовать поддержку динамического размещения компонентов в системе;

– решить задачу автоматического упорядоченного размещения множества компонентов на формах.

– Для представления визуальных и текстовых элементов отчета в системе предусмотрены следующие компоненты:

– System.Windows.Forms.Label – для размещения отдельных текстовых меток на формах;

– System.Windows.Forms.LinkLabel – для размещения команд типа «гиперссылка»;

– тулбоксы (System.Windows.Forms.ToolStrip, наборы команд) – реализуют меню подсистем, сервисное меню и контекстное меню управления данными в таблицах;

– поля ввода данных (System.Windows.Forms.TextBox) – текстовые поля и списки выбора, предназначенные для ввода данных в систему;

– картинки (System.Windows.Forms.PictureBox) – служат для отображения пиктограмм к командам и выполняют информативно-декоративную функцию;

– поля для ввода чисел (System.Windows.Forms.NumericUpDown) – поле, в которое вводится только числовое значение (снабжено также кнопками инкремента и декремента), которое ограничивается снизу и сверху.

На рисунке 2.3 приведен макет главной формы системы (в среде разработки MS Visual Studio).

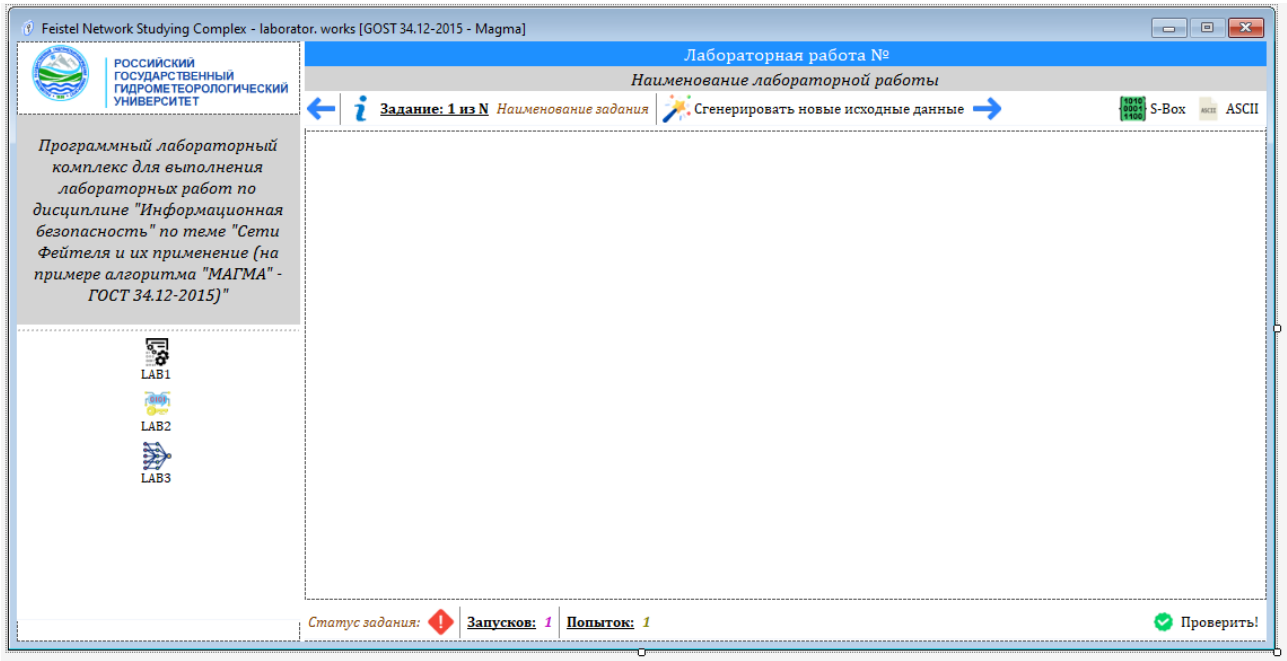


Рисунок 2.3 – Макет формы управления списком задач

Как видно из рисунка 2.3, главная форма разделена на следующие области (с помощью средств и контейнеров, описанных выше):

– область логотипа и названия программы;

– главное меню выбора лабораторной работы;

– область данных текущей лабораторной работы (меню и название);

– меню навигации по заданиям текущей лабораторной работы;

– область отображения компонентов, средств и инструментов выполнения задания текущей лабораторной работы;

– меню статуса выполнения текущего задания текущей лабораторной работы;

## 2.4. КОНТРОЛЬНЫЙ ПРИМЕР ВЫПОЛНЕНИЯ ПРОГРАММЫ

При запуске программного комплекса FNSC система генерирует главное меню с доступом к лабораторным работам – рис. 2.4.

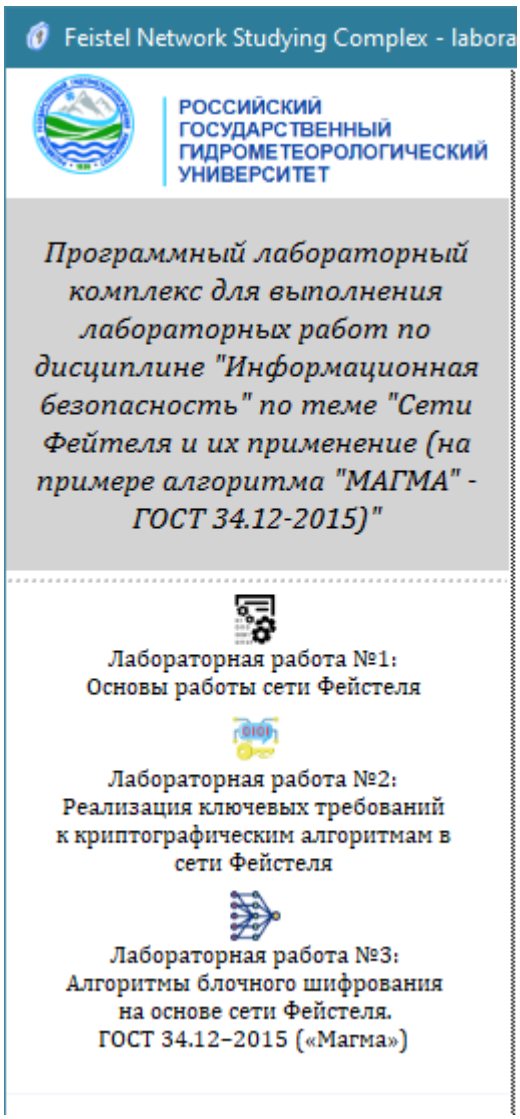


Рисунок 2.4 – Главное меню с доступом к лабораторным работам

Из рисунка видно, что при запуске системы созданы три пункта меню – каждый запускает соответствующие инструменты для выполнения указанной в пункте лабораторной работы.

Первая лабораторная работа состоит из четырех заданий. При запуске этой лабораторной открывается первое задание. Для выполнения этого задания предлагается на основе сгенерированного базового ключа (256 бит) сформировать 32 раундовых ключа для сети Фейстеля в соответствии с алгоритмом «Магма» (ГОСТ 34.12–2015). Для выполнения работы программа создает следующие компоненты:

– блок базового 32-байтного ключа;

– 32 блока для ввода данных раундовых 4-байтных ключей.

Исходные данные для задания лабораторной работы генерируются случайным образом. Результаты проверяются системой, при этом красным цветом подсвечиваются некорректные блоки, а зеленым – корректные. На рисунке 2.5 приведен пример выполнения первого задания лабораторной работы №1 в приложении FNSC для выполнения лабораторного практикума.

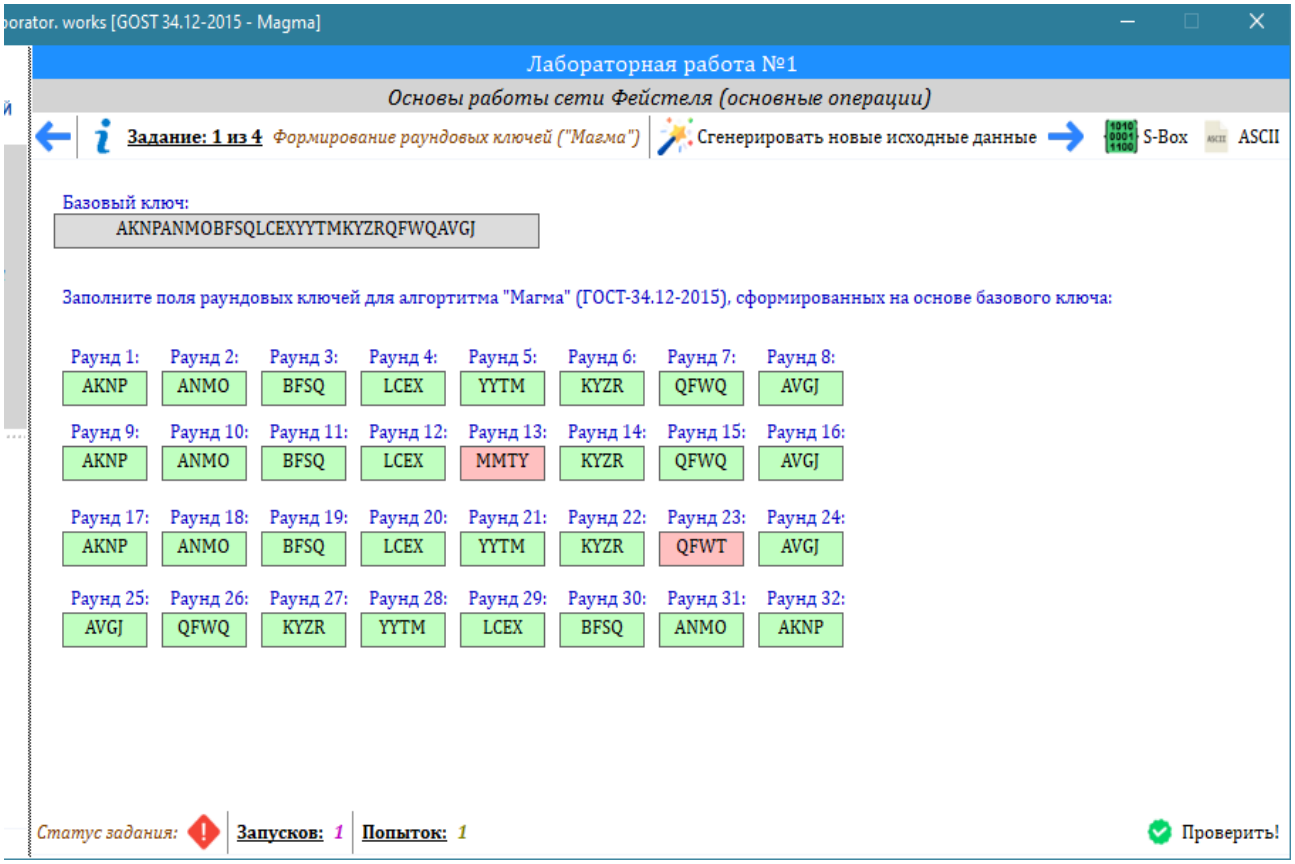


Рисунок 2.5 – Пример выполнения первого задания лабораторной работы №1 в приложении FNSC

Как видно из рисунка, статус представленного на нем задания помечен как «с ошибками» (красный ромб с восклицательным знаком), при этом красным цветом подсвечены места допущенных ошибок. Их легко проверить, если аналитическим способом (вручную) подобрать ключи в соответствии с правилами, установленными в алгоритме, на основе сгенерированного базового ключа.

Для выполнения второго задания лабораторной работы №1 предлагается выполнить в двоичном коде операцию сложения по модулю 32 (одна из

операций функции Фейстеля) двух сгенерированных 32-битных блоков программа создает следующие компоненты:

– два 4-байтных блока исходных данных для выполнения операции сложения по модулю 32 (блоки генерируются в символьном (байтовом) представлении);

– двоичный блок (32 бита) для записи первого слагаемого в двоичной системе счисления – двоичный вид сгенерированного байта основывается на ASCII-коде символа;

– двоичный блок (32 бита) для записи второго слагаемого в двоичной системе счисления – двоичный вид сгенерированного байта основывается на ASCII-коде символа;

– двоичный блок (32 бита) для записи результата сложения исходных блоков по модулю 32.

Исходные данные для задания лабораторной работы генерируются случайным образом. Результаты проверяются системой, при этом красным цветом подсвечиваются некорректные блоки, а зеленым – корректные. При этом для 32-битных блоков при наличии ошибки в одном бите красным будет подсвечиваться весь байт, содержащий ошибочный бит. Для перевода исходных блоков в двоичный вид предлагается воспользоваться встроенной в программу таблицей ASCII-кодов символов.

На рисунке 2.6 приведен пример выполнения второго задания лабораторной работы №1 в приложении FNSC для выполнения лабораторного практикума. Как видно из рисунка, статус представленного на нем задания помечен как «с ошибками» (красный ромб с восклицательным знаком), при этом красным цветом подсвечены места допущенных ошибок. Их легко проверить, если аналитическим способом (вручную) выполнить предложенные операции в соответствии с правилами бинарной логики.

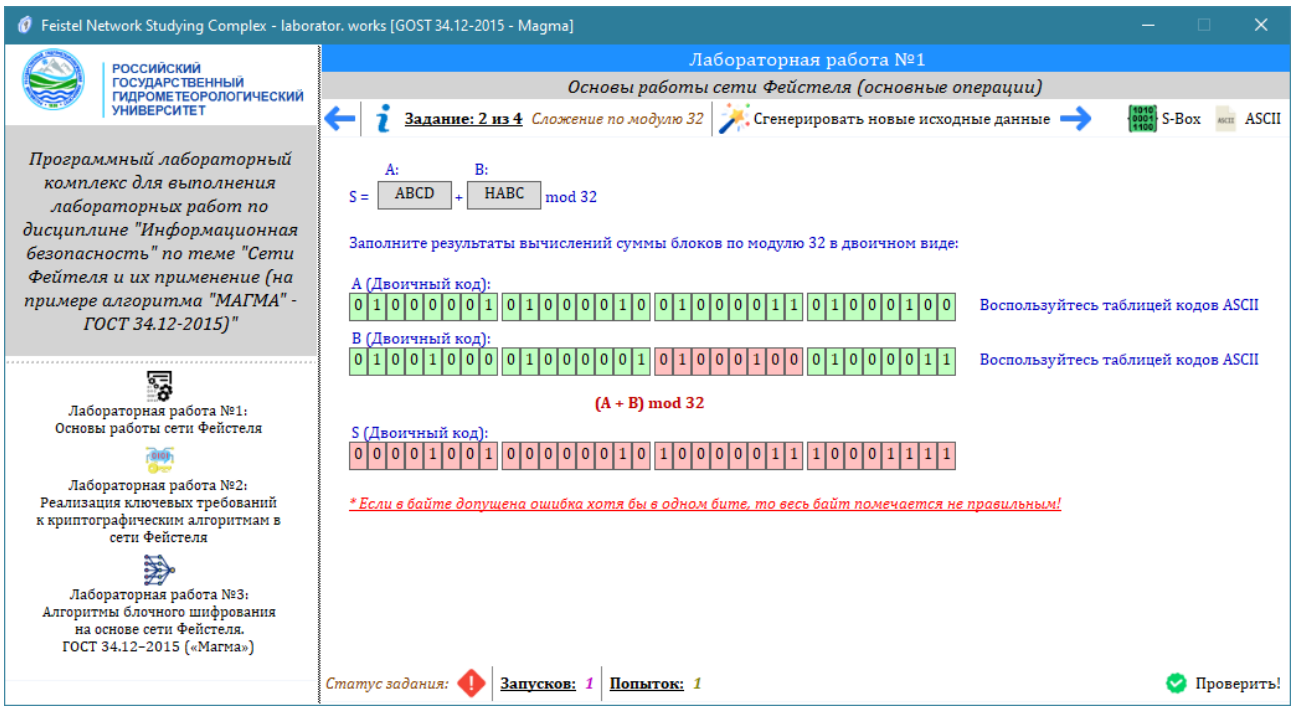


Рисунок 2.6 – Пример выполнения второго задания лабораторной работы №1 в приложении FNSC

Для выполнения третьего задания лабораторной работы №1 предлагается для сгенерированного 32-битного блока выполнить S-перестановку (часть функции Фейстеля, реализующая принцип конфузии) в соответствии с алгоритмом «Магма» (ГОСТ 34.12–2015). Для этого программа создает следующие компоненты:

– двоичный блок (32 бита) для записи исходного блока;

– двоичный блок (32 бита) для записи подстановок.

Исходные данные для задания лабораторной работы генерируются случайным образом. Результаты проверяются системой, при этом красным цветом подсвечиваются некорректные блоки, а зеленым – корректные. При этом для 32-битных блоков при наличии ошибки в одном бите красным будет подсвечиваться весь байт, содержащий ошибочный бит. Для реализации подстановок предлагается воспользоваться стандартом ГОСТ 34.12–2015 или же встроенной в программу таблицей S-подстановок. На рисунке 2.7 приведен пример выполнения третьего задания лабораторной работы №1 в приложении FNSC для выполнения лабораторного практикума.

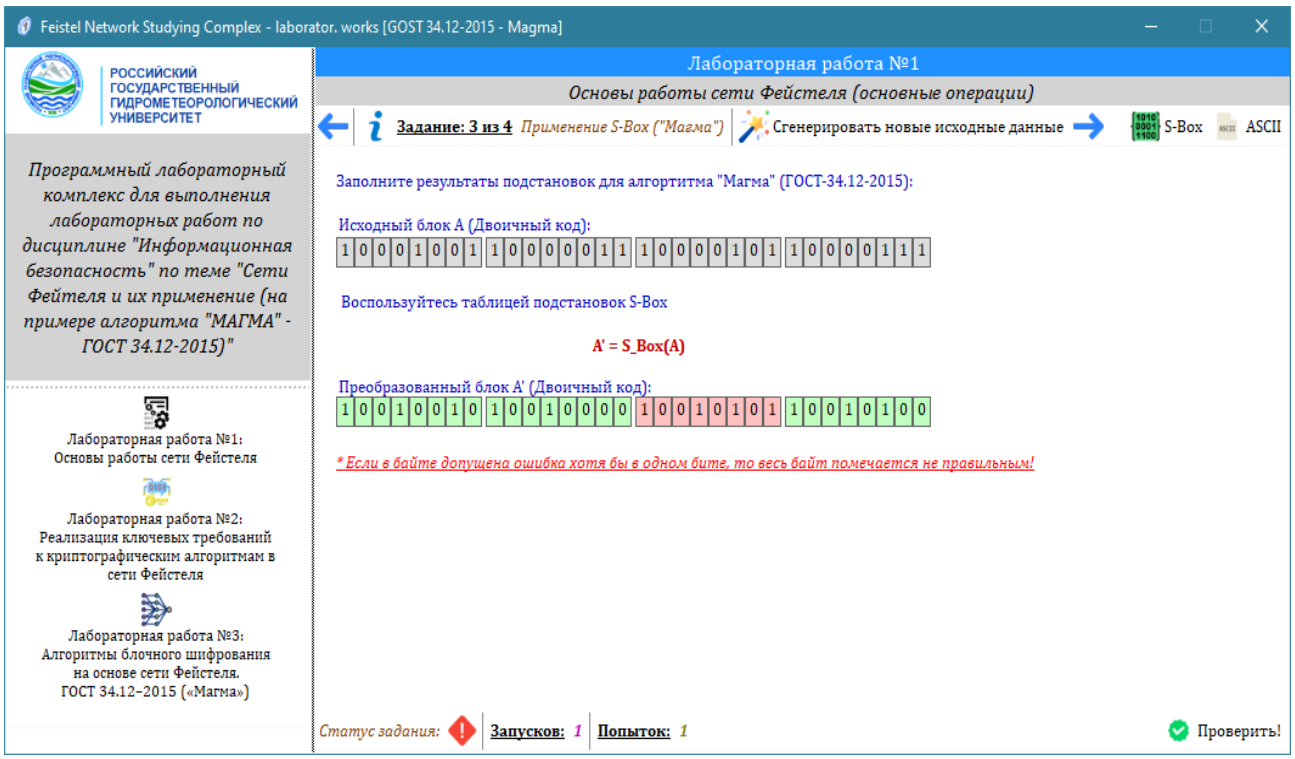


Рисунок 2.7 – Пример выполнения третьего задания лабораторной работы №1 в приложении FNSC

Как видно из рисунка, статус представленного на нем задания помечен как «с ошибками» (красный ромб с восклицательным знаком), при этом красным цветом подсвечены места допущенных ошибок. Их легко проверить, если аналитическим способом (вручную) выполнить предложенные операции в соответствии с правилом постановок.

В четвертом задании предлагается выполнить в двоичном коде базовые логические операции сети Фейстеля (сложения по модулю 2, сдвиг битов) для двух сгенерированных 32-битных блоков. Для выполнения этого задания лабораторной работы №1 программа создает следующие компоненты:

– двоичный блок (32 бита) для записи первого исходного блока;

– двоичный блок (32 бита) для записи второго исходного блока

– двоичный блок (32 бита) для записи результатов сложения типа XOR (по модулю 2);

– двоичный блок (32 бита) для записи результатов операции циклического сдвига.

Исходные данные для задания лабораторной работы генерируются случайным образом (исходные блоки-операнды, величина и направление циклического сдвига). Результаты проверяются системой, при этом красным цветом подсвечиваются некорректные блоки, а зеленым – корректные. При этом для 32-битных блоков при наличии ошибки в одном бите красным будет подсвечиваться весь байт, содержащий ошибочный бит. На рисунке 2.8 приведен пример выполнения четвертого задания лабораторной работы №1 в приложении FNSC для выполнения лабораторного практикума.

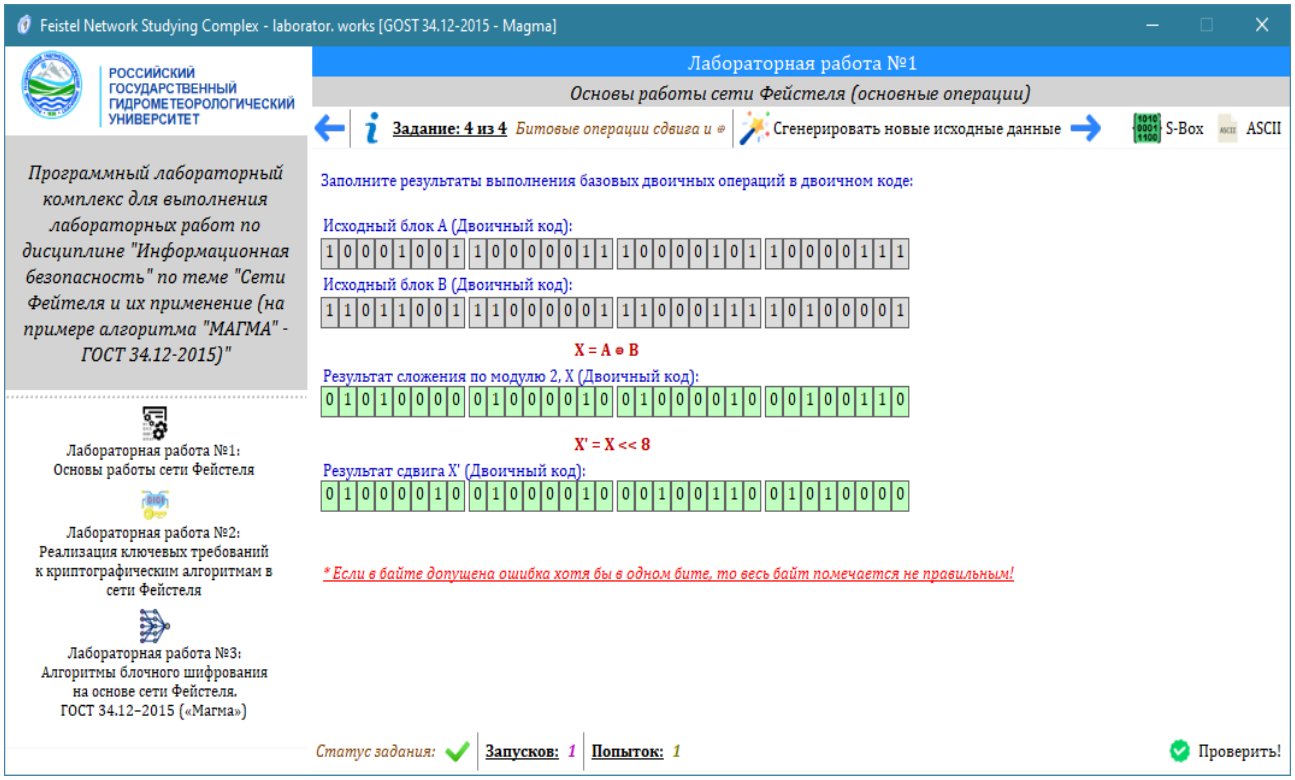


Рисунок 2.8 – Пример выполнения четвертого задания лабораторной работы №1 в приложении FNSC

Как видно из рисунка, статус представленного на нем задания помечен как «выполнено корректно» (зеленая «галочка»), при этом все поля ввода помечены зеленым цветом. Результаты легко проверить, если аналитическим способом (вручную) выполнить предложенные двоичные операции в соответствии с правилами бинарной логики.

После выполнения первой лабораторной работы выполняется лабораторная работа №2. Она содержит всего одно задание. В этой работе

предлагается реализовать визуализацию лавинного эффекта. Для двух заданных (сгенерированных) 32-битных подблоков (L1, R1) программа вычисляет и визуализирует результаты всех этапов шифрования сетью Фейстеля (для двух первых раундов): результаты сложения по модулю 2 (с оппозиционным подблоком), 32 (с ключом), перестановки, битовый сдвиг. Затем программа случайным образом меняет один байт в исходных данных (в ключе или любом из подблоков). Задача студента – пересчитать все вычисления и на практике убедиться в лавинном эффекте (при изменении даже маленькой части результат изменяется с лавинообразным характером).

Для выполнения этого задания лабораторной работы №2 программа создает следующие компоненты:

– двоичный блок (32 бита) для записи левого подблока;

– двоичный блок (32 бита) для записи правого подблока;

– двоичный блок (32 бита) для записи раундового ключа (в данном задании принимается допущение, что раундовый ключ одинаковый для обоих раундов);

– двоичный блок (32 бита) для записи двоичного представления левого подблока;

– двоичный блок (32 бита) для записи двоичного представления правого подблока;

– двоичный блок (32 бита) для записи двоичного представления ключа (для каждого раунда);

– двоичный блок (32 бита) для записи двоичного представления результатов операции сложения подблока с ключом по модулю 32 (для каждого раунда);

– двоичный блок (32 бита) для записи двоичного представления результатов операции подстановок (для каждого раунда);

– двоичный блок (32 бита) для записи двоичного представления результатов операции циклического сдвига на 11 бит (для каждого раунда);

– двоичный блок (32 бита) для записи результатов сложения подблоков по модулю 2 (для каждого раунда).

Исходные данные для задания лабораторной работы генерируются случайным образом (исходные блоки-операнды, байт изменения исходных данных в ключе или одном из подблоков). Результаты проверяются системой, при этом красным цветом подсвечиваются некорректные блоки, а зеленым – корректные. При этом для 32-битных блоков при наличии ошибки в одном бите красным будет подсвечиваться весь байт, содержащий ошибочный бит. На рисунке 2.9 приведен пример выполнения задания лабораторной работы №2 в приложении FNSC для выполнения лабораторного практикума.

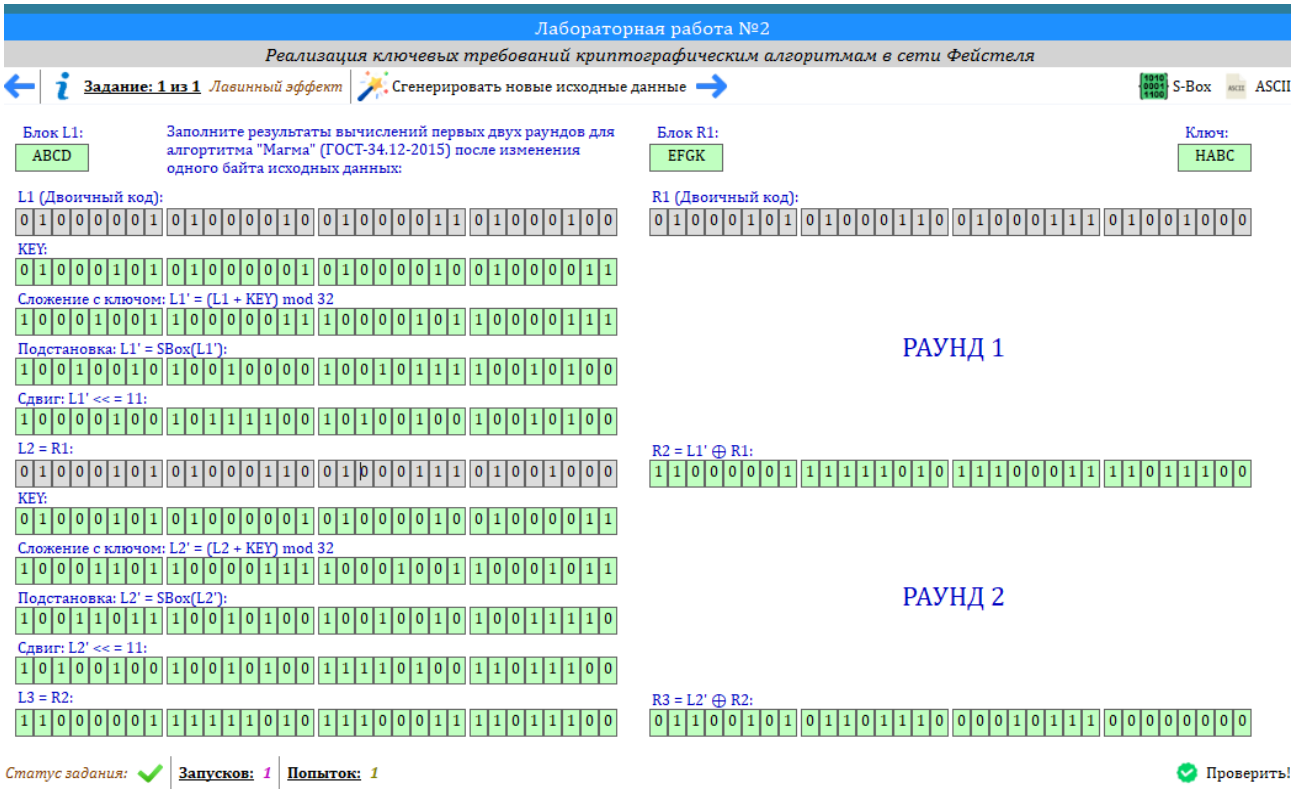


Рисунок 2.9 – Пример выполнения задания лабораторной работы №2 в приложении FNSC

Как видно из рисунка, статус представленного на нем задания помечен как «выполнено корректно» (зеленая «галочка»), при этом все поля ввода помечены зеленым цветом. Результаты легко проверить, если аналитическим способом (вручную) выполнить все операции, предусмотренные в раундах.

После выполнения второй лабораторной работы выполняется лабораторная работа №3. Она содержит всего одно задание. В этой работе предлагается реализовать студентом контроль работы алгоритма. Аналогично работе №2 для двух заданных (сгенерированных) 32-битных подблоков ($L_1$, $R_1$) программа вычисляет и визуализирует результаты всех этапов шифрования сетью Фейстеля (для двух первых раундов). Затем программа случайно добавляет в расчеты несколько ошибок, которые пользователь должен найти и исправить.

Для выполнения этого задания лабораторной работы №3 программа создает следующие компоненты:

– блок базового 32-байтного;

– двоичный блок (32 бита) для записи левого подблока;

– двоичный блок (32 бита) для записи правого подблока;

– двоичный блок (32 бита) для записи раундового ключа (в данном задании все раундовые ключи должны генерироваться по правилам алгоритма);

– двоичный блок (32 бита) для записи двоичного представления левого подблока;

– двоичный блок (32 бита) для записи двоичного представления правого подблока;

– двоичный блок (32 бита) для записи двоичного представления ключа (для каждого раунда);

– двоичный блок (32 бита) для записи двоичного представления результатов сложения подблока с ключом по модулю 32 (для каждого раунда);

– двоичный блок (32 бита) для записи двоичного представления результатов операции подстановок (для каждого раунда);

– двоичный блок (32 бита) для записи двоичного представления результатов операции циклического сдвига на 11 бит (для каждого раунда);

– двоичный блок (32 бита) для записи результатов сложения подблоков по модулю 2 (для каждого раунда).

Исходные данные для задания лабораторной работы генерируются случайным образом (исходные блоки-операнды, места ошибок). Результаты проверяются системой, при этом красным цветом подсвечиваются некорректные блоки, а зеленым – корректные. При этом для 32-битных блоков при наличии ошибки в одном бите красным будет подсвечиваться весь байт, содержащий ошибочный бит. На рисунке 2.10 приведен пример выполнения лабораторной работы №3 в приложении FNSC для выполнения лабораторного практикума.

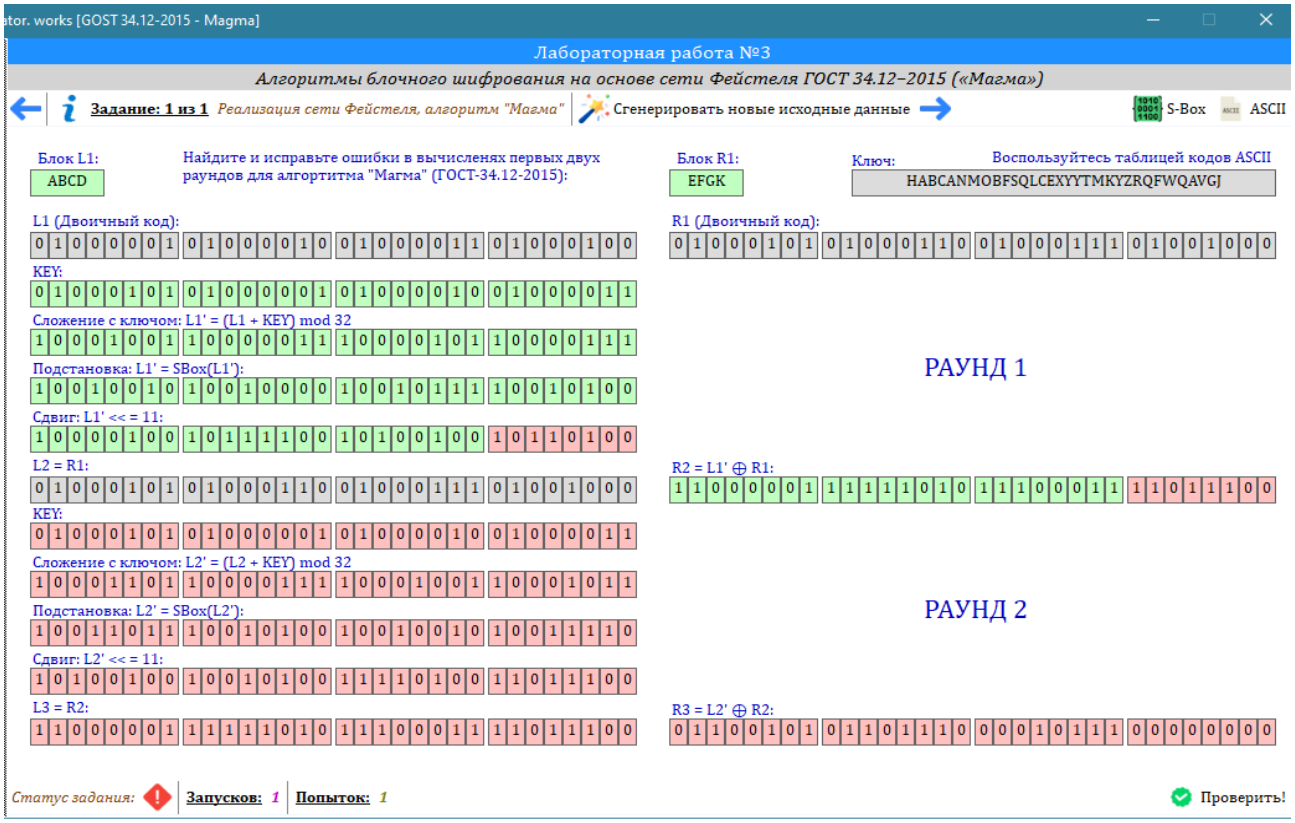


Рисунок 2.10 – Пример выполнения задания лабораторной работы №3 в приложении FNSC

Как видно из рисунка, статус представленного на нем задания помечен как «с ошибками» (красный ромб с восклицательным знаком), при этом красным цветом подсвечены места допущенных ошибок. Их легко проверить, если аналитическим способом (вручную) выполнить все операции, предусмотренные в раундах.

# 3.ГЛАВА. РАЗРАБОТКА МЕТОДИЧЕСКОГО ОБЕСПЕЧЕНИЯ ЛАБОРАТОРНОГО ПРАКТИКУМА НА ОСНОВЕ СЕТИ ФЕЙСТЕЛЯ

## 3.1 ЛАБОРАТОРНАЯ РАБОТА №1. ОСНОВЫ РАБОТЫ СЕТИ ФЕЙСТЕЛЯ

**Наименование работы:** «Основы работы сети Фейстеля».

**Цель лабораторной работы:** ознакомиться с базовыми понятиями блочного шифрования, изучить теоретический материал по работе сети Фейстеля и ее применению в алгоритмах шифрования, освоить на практике принципы работы и основные операции, производимые в рамках расчета сети Фейстеля. В результате выполнения работы у студента должны сформироваться:

– знание принципов блочного шифрования и основных криптографических алгоритмов;

– понимание принципов работы сети Фейстеля;

– умение самостоятельно выполнять базовые операции, предусмотренные в работе сети Фейстеля.

**Задание на лабораторную работу**

Лабораторная работа включает в себя выполнение четырех заданий, каждое из которых представляет собой отдельную операцию, выполняемую в алгоритме блочного шифрования «Магма», основанного на сети Фейстеля:

– на основе заданного базового ключа (256 бит) сформировать 32 раундовых ключа для сети Фейстеля в соответствии с алгоритмом «Магма» (ГОСТ 34.12–2015);

– выполнить в двоичном коде операцию сложения по модулю 32 (одна из операций функции Фейстеля) двух заданных 32-битных блоков;

– для сгенерированного 32-битного блока выполнить S-перестановку (часть функции Фейстеля, реализующая принцип конфузии) в соответствии с алгоритмом «Магма» (ГОСТ 34.12–2015);

– выполнить в двоичном коде базовые логические операции сети Фейстеля (сложения по модулю 2, сдвиг битов) для двух сгенерированных 32-битных блоков.

**Программно-аппаратное обеспечение**

Лабораторная работа выполняется на лабораторном персональном компьютере / ноутбуке. Для выполнения всех заданий используется специальное программное обеспечение лабораторного практикума – FNSC, установленное на лабораторном компьютере.

**Краткие теоретические сведения**

Приведены в п. 1.2.

**Технология выполнения работы**

Для выполнения практической части лабораторной работы запускается программа FNSC, затем в меню выбора практических работ выбирается «Лабораторная работа №1» (рис 3.1).

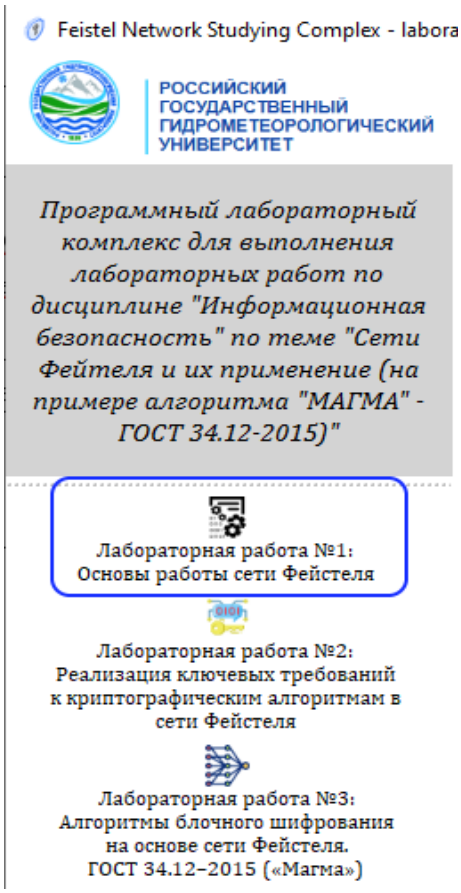


Рисунок 3.1 – Выбор первой лабораторной работы

При выполнении заданий лабораторной работы следовать указаниям в программе. Задания выполняются последовательно. В каждом задании предусматривается два счетчика: счетчик запусков и счетчик попыток (рис. 3.2).

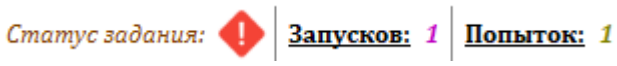


Рисунок 3.2 – Счетчики запусков и попыток

Идеальный случай – это когда для выполнения одного задания был один запуск и одна попытка. В этом случае за задание присваивается максимальные 100 баллов.

В задании можно сгенерировать новые данные (вариант) – при этом счетчик запусков будет инкрементирован на единицу. Перезапуск является нежелательным и показывает, что студент не может справиться с заданием. В этом случае в текущем задании рекомендуется снять штрафные баллы – по 20 баллов за каждый перезапуск задания после первого.

Каждое задание отправляется на проверку с помощью соответствующей команды (рис. 3.3). После успешного выполнения задания становится доступно следующее.

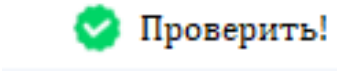


Рисунок 3.3 – Кнопка отправки задания на проверку

Результаты проверяются системой, при этом красным цветом подсвечиваются некорректные блоки, а зеленым – корректные. При этом для 32-битных блоков при наличии ошибки в одном бите красным будет подсвечиваться весь байт, содержащий ошибочный бит. Каждая неуспешная проверка нежелательна и показывает, что студент плохо усвоил материал или был невнимателен. Каждая неуспешная проверка увеличивает счетчик попыток. В этом случае в текущем задании рекомендуется снять штрафные баллы – по 5 баллов за каждую неудачную попытку после первой.

**Порядок выполнения работы**

После запуска первой лабораторной работы с помощью специального программного обеспечения лабораторного практикума для выполнения открывается первое задание. В данном задании программа генерирует 32-байтный ключ (рис. 3.4).

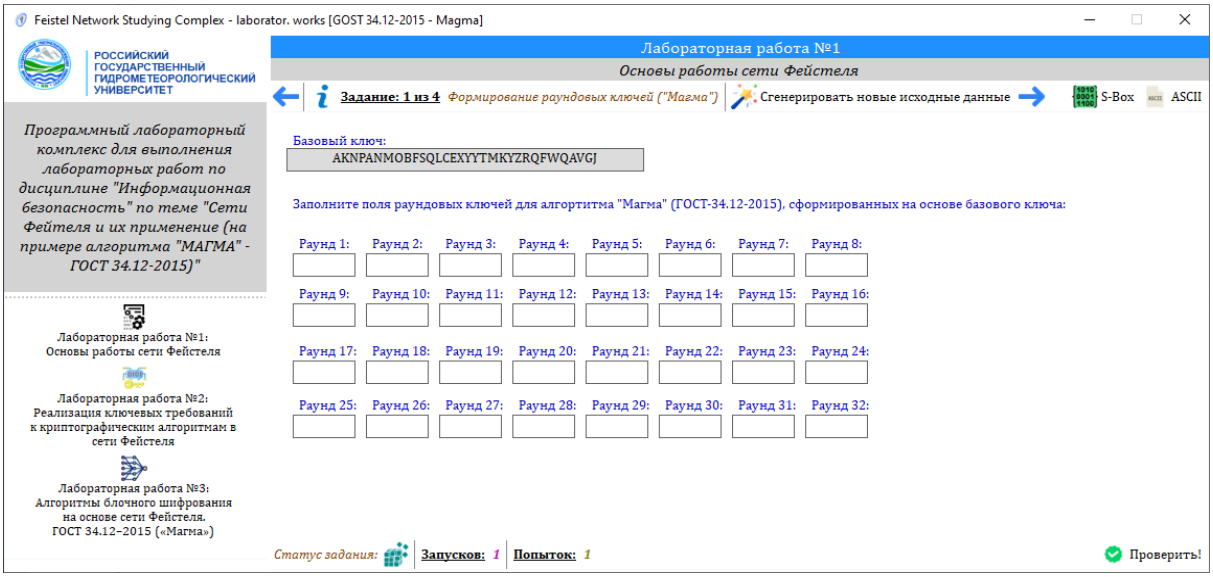


Рисунок 3.4 – Бланк выполнения первого задания лабораторной работы №1

Задача – используя правила формирования раундовых ключей, принятые в алгоритме «Магма» ГОСТ 34.12-2015, сформировать на основе базового 32 раундовых ключа и записать их в соответствующие поля. После заполнения – отправить на проверку. Сделать скриншот экрана, в котором обязательно должны быть видны счетчики и статус задания, а также поля с исходными данными и введенными значениями.

При правильном выполнении первого задания программа даст возможность перейти к следующему: стрелка вправо в верхнем меню. Бланк второго задания приведен на рис. 3.5.

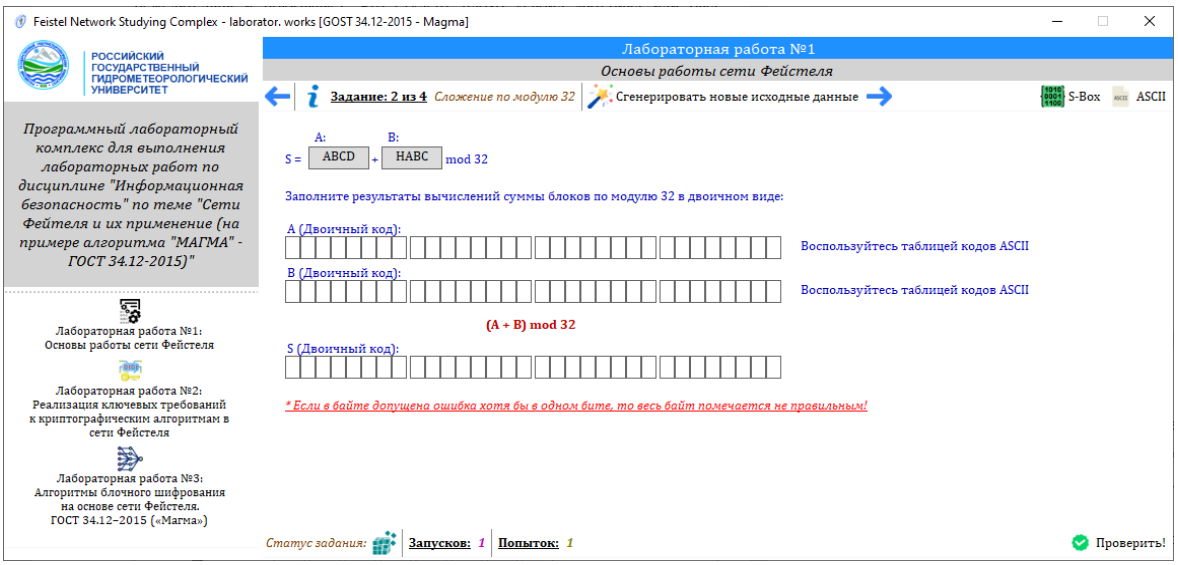


Рисунок 3.5 – Бланк выполнения второго задания лабораторной работы №1

Во втором задании программа генерирует два 4-байтных подблока – A и B. Для выполнения задания необходимо:

– заполнить поля двоичного представления подблока A;

– заполнить поля двоичного представления подблока B;

– заполнить поля двоичного представления результата выполнения операции $(A + B) \bmod^{32}$.

После заполнения всех указанных полей – отправить на проверку. Сделать скриншот экрана, в котором обязательно должны быть видны счетчики и статус задания, а также поля с исходными данными и введенными значениями.

При правильном выполнении второго задания программа даст возможность перейти к следующему: стрелка вправо в верхнем меню. Бланк третьего задания приведен на рис. 3.6.

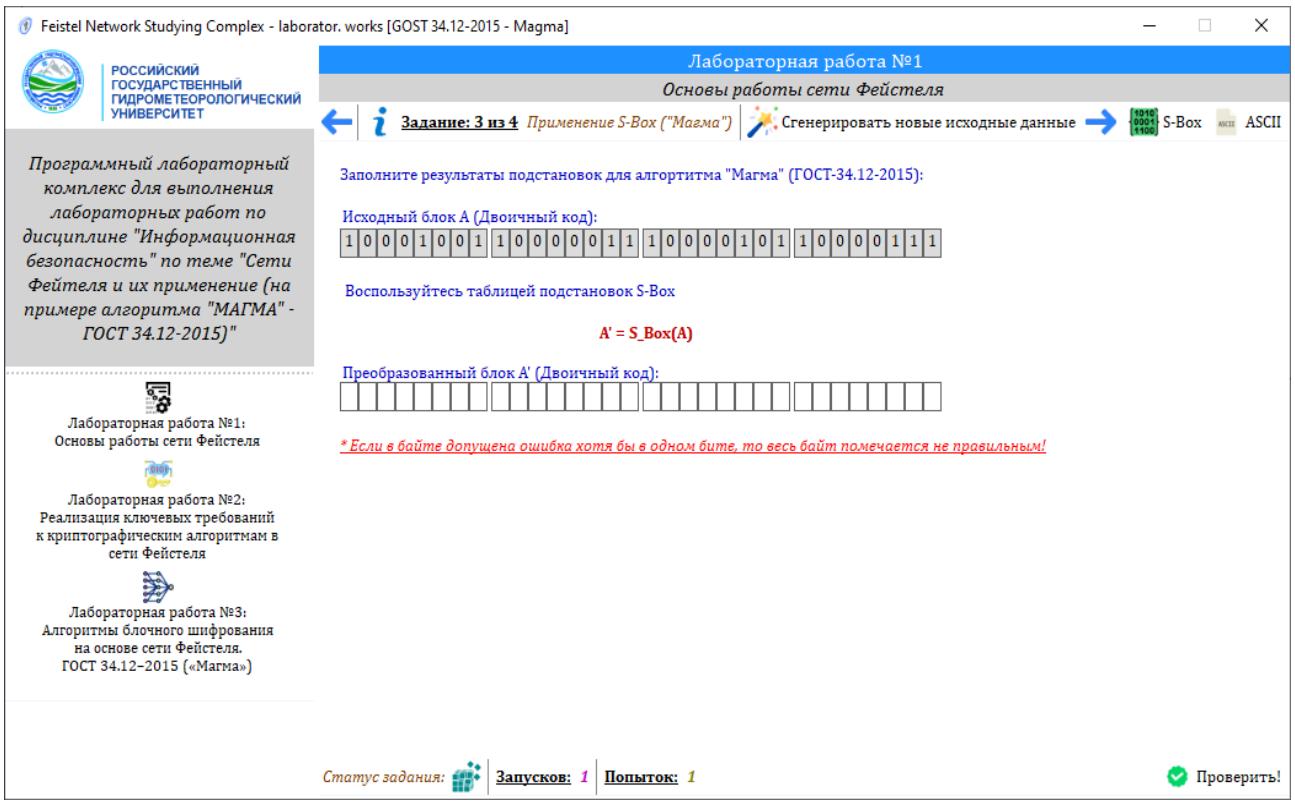


Рисунок 3.6 – Бланк выполнения третьего задания лабораторной работы №1

Во третьем задании программа генерирует один 4-байтный подблок в двоичном представлении. Для выполнения задания необходимо выполнить S-преобразование исходного блока в соответствии с правилами алгоритма «Магма» и записать полученный результат в двоичном виде в соответствующие пустые поля.

После заполнения всех указанных полей – отправить на проверку. Сделать скриншот экрана, в котором обязательно должны быть видны счетчики и статус задания, а также поля с исходными данными и введенными значениями.

При правильном выполнении третьего задания программа даст возможность перейти к следующему: стрелка вправо в верхнем меню. Бланк четвертого задания приведен на рис. 3.7.

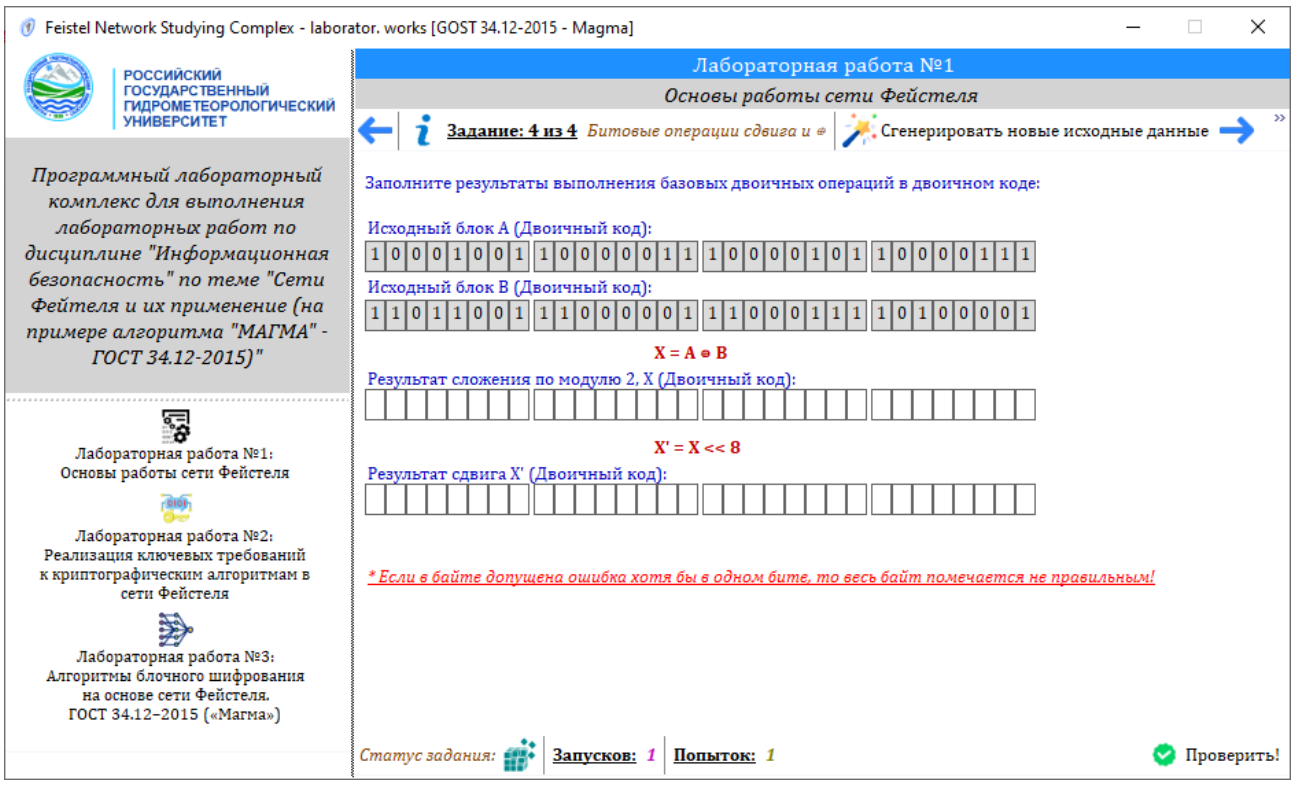


Рисунок 3.7 – Бланк выполнения четвертого задания лабораторной работы №1

Во четвертом задании программа генерирует два 4-байтных подблока (A и B) в двоичном представлении. Для выполнения задания необходимо выполнить две бинарные операции и записать результат выполнения каждой из них в двоичном виде:

– сложение A и B по модулю 2: X = A ⊕ B;

– побитовый сдвиг полученного результата в указанном направлении на указанное число бит, например: X’ = X << 8 (направление и величина сдвига задается вариантом задания).

После заполнения всех указанных полей – отправить на проверку. Сделать скриншот экрана, в котором обязательно должны быть видны счетчики и статус задания, а также поля с исходными данными и введенными значениями.

**Требования к отчету**

Отчет должен содержать следующие пункты:

– титульный лист;

– название, цель лабораторной работы;

– индивидуальное задание;

– ход выполнения работы, в котором должны быть размещены сделанные скриншоты;

– заключение и выводы.

**Варианты индивидуальных заданий для самостоятельного выполнения**

Варианты задания генерируются программой случайным образом:

– для первого задания – исходный 32-байтный ключ;

– для второго задания – два 4-байтных подблока;

– для третьего задания – 32-битный подблок;

– для четвертого задания – два 32-битный подблока, направление и величину сдвига.

**Контрольные вопросы для самопроверки**

1. Какие виды шифрования вы знаете?
2. В чем отличие блочных и поточных шифров? Какова сфера их применения?
3. Приведите примеры алгоритмов шифрования, использующих блочные и поточные шифры.
4. В чем состоит основной принцип сети Фейстеля? Почему она называется «сетью»?
5. Что такое функция Фейстеля?
6. Что такое принципы конфузии и дифузии?
7. Какие основные битовые операции используются в шифровании с помощью сети Фейстеля?
8. Что такое перестановки / подстановки в алгоритмах шифрования? Зачем они нужны?
9. Что означает «сумма по модулю»?
10. Какая простейшая логическая операция эквивалентна сложению по модулю 2?

**Рекомендуемая литература**


1. Алферов, А. П. Основы криптографии : учебное пособие / А. П. Алферов, А. Ю. Зубов, А. С. Кузьмин, А. В. Черемушкин. – 3-е изд., перераб. и доп. – Москва : Гелиос АРВ, 2005. – 480 с.
2. Бабаш, А. В. Криптографические методы защиты информации : учебник для вузов / А. В. Бабаш, Е. К. Баранова. – Москва : КноРус, 2022. – 190 с.
3. Венбо, М. Современная криптография: теория и практика / М. Венбо ; перевод с английского С. А. Панасенко. – Москва : Вильямс, 2005. – 768 с.
4. Молдовян, Н. А. Теоретический минимум и основы криптографии : учебное пособие / Н. А. Молдовян. – Санкт-Петербург : БХВ-Петербург, 2010. – 160 с.
5. Шнайер, Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер. – Москва : Триумф, 2003. – 816 с.


## 3.2. ЛАБОРАТОРНАЯ РАБОТА №2. РЕАЛИЗАЦИЯ КЛЮЧЕВЫХ ТРЕБОВАНИЙ К КРИПТОГРАФИЧЕСКИМ АЛГОРИТМАМ В СЕТИ ФЕЙСТЕЛЯ

**Наименование работы:** «Реализация ключевых требований к криптографическим алгоритмам в сети Фейстеля».

**Цель лабораторной работы:** ознакомиться с ключевыми требованиями к криптографическим алгоритмам. Изучить принципы лавинного эффекта и его реализации в сети Фейстеля. В результате выполнения работы у студента должны сформироваться:

– знание требований к криптографическим алгоритмам (секретность, криптоскойкость, лавинный эффект, эффективность, корректность);

– понимание и умение анализировать стойкость криптографических алгоритмов;

– практическое освоение способов реализации лавинного эффекта в сети Фейстеля.

**Задание на лабораторную работу**

Лабораторная работа включает в себя выполнение одного задания. Требуется реализовать визуализацию лавинного эффекта. Для двух заданных (сгенерированных) 32-битных подблоков ($L_1$, $R_1$) программа вычисляет и визуализирует результаты всех этапов шифрования сетью Фейстеля (для двух первых раундов): результаты сложения по модулю 2 (с оппозиционным подблоком), 32 (с ключом), перестановки, битовый сдвиг. Затем программа случайным образом меняет один байт в исходных данных (в ключе или любом из подблоков). Задача студента – пересчитать все вычисления и на практике убедиться в лавинном эффекте (при изменении даже маленькой части результат изменяется с лавинообразным характером)

**Программно-аппаратное обеспечение**

Лабораторная работа выполняется на лабораторном персональном компьютере / ноутбуке. Для выполнения всех заданий используется специальное программное обеспечение лабораторного практикума – FNSC, установленное на лабораторном компьютере.

**Краткие теоретические сведения**

В сети Фейстеля лавинный эффект представляет собой процесс многократного повторения простых преобразований входных данных в хоте выполнения каждой операции отдельного раунда. От операции к операции незаметные в начале изменения накладываются друг на друга – в результате первичные данные к концу раунда преобразуются так, что в них не остается ничего общего с первоначальными. Такие изменения результата вызывают даже незначительное изменение входных данных (например, одного бита в ключе).

В алгоритме «Магма» катализатором лавинного эффекта являются:

– S-преобразования (подстановки);

– циклический сдвиг на 11 бит.

**Технология выполнения работы**

Для выполнения практической части лабораторной работы запускается программа FNSC, затем в меню выбора практических работ выбирается «Лабораторная работа №2» (рис 3.8).

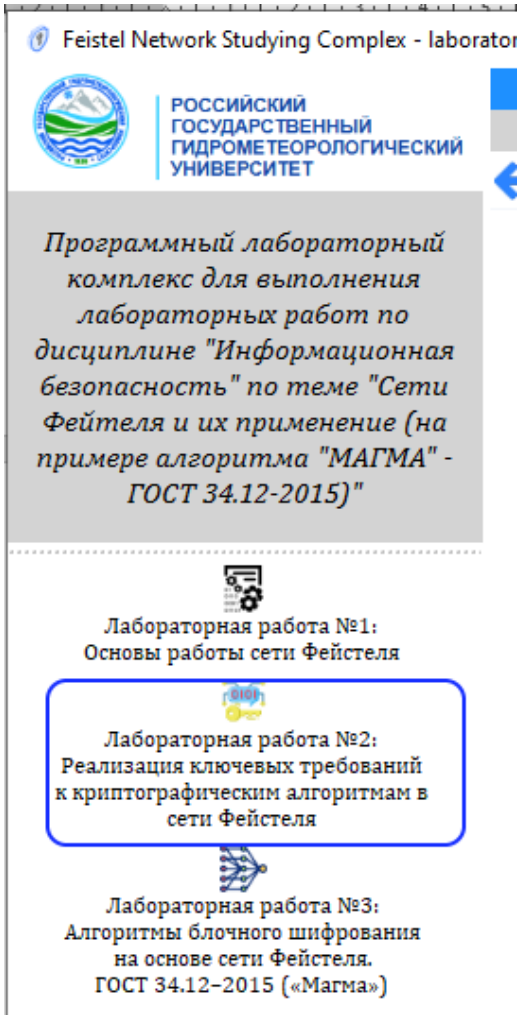


Рисунок 3.8 – Выбор второй лабораторной работы

При выполнении заданий лабораторной работы следовать указаниям в программе. В задании предусматривается два счетчика: счетчик запусков и счетчик попыток. Идеальный случай – это когда для выполнения работы был один запуск и одна попытка. В этом случае за работу присваивается максимальные 100 баллов. В работе можно сгенерировать новые данные (вариант) – при этом счетчик запусков будет инкрементирован на единицу. Перезапуск является нежелательным и показывает, что студент не может справиться с работой. В этом случае в работе рекомендуется снять штрафные

баллы – по 20 баллов за каждый перезапуск после первого. Выполненная работа отправляется на проверку с помощью соответствующей команды.

Результаты проверяются системой, при этом красным цветом подсвечиваются некорректные блоки, а зеленым – корректные. При этом для 32-битных блоков при наличии ошибки в одном бите красным будет подсвечиваться весь байт, содержащий ошибочный бит. Каждая неуспешная проверка нежелательна и показывает, что студент плохо усвоил материал или был невнимателен. Каждая неуспешная проверка увеличивает счетчик попыток. В этом случае в текущем задании рекомендуется снять штрафные баллы – по 5 баллов за каждую неудачную попытку после первой.

**Порядок выполнения работы**

После запуска второй лабораторной работы с помощью специального программного обеспечения лабораторного практикума для выполнения открывается задание (рис. 3.9).

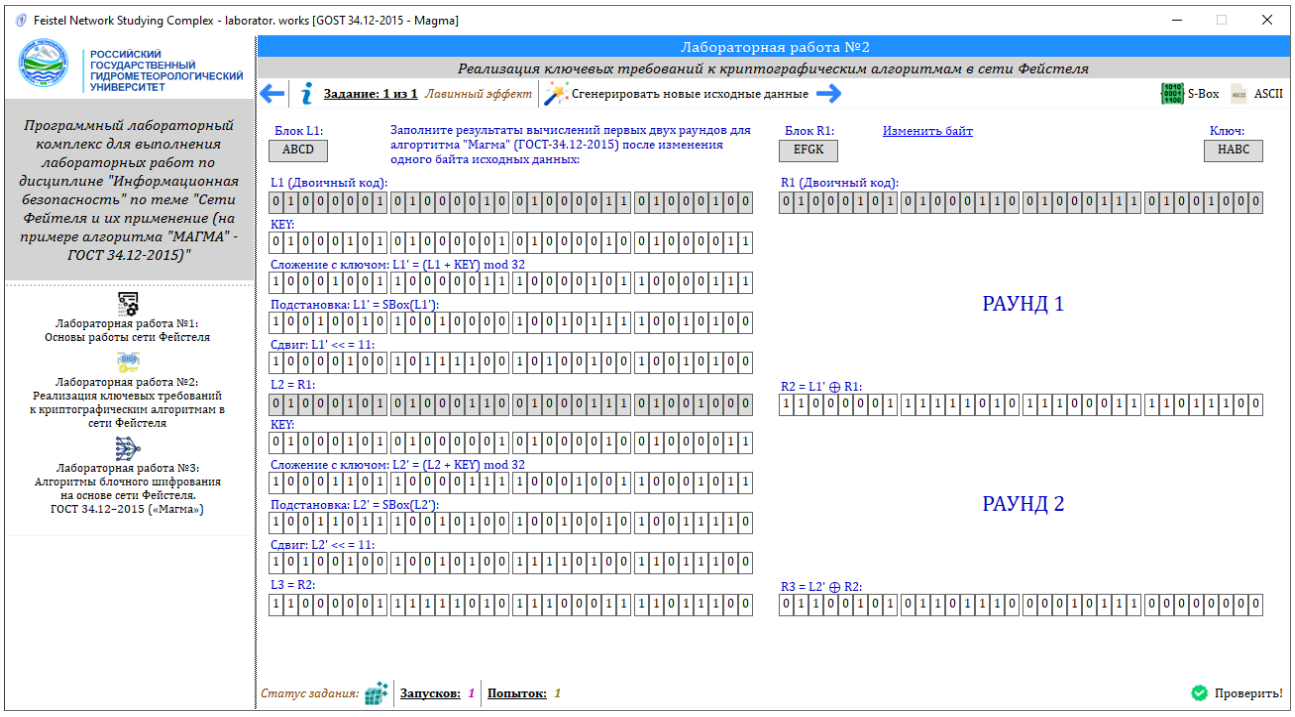


Рисунок 3.9 – Бланк выполнения лабораторной работы №2

Задача работы. Просмотреть результаты выполнения двух первых раундов алгоритма «Магма». Сделать скриншот экрана, в котором обязательно должны быть видны поля с исходными данными и рассчитанными значениями.

Затем выполнить команду «Изменить байт». Программа случайным образом изменит один из байтов в любом из блоков или ключе. Место измененного байта будет проиндицировано – рис. 3.10.


Рисунок 3.10 – Изменение данных для проверки лавинного эффекта

В соответствии с новыми исходными данными, полученными в результате изменения одного байта, необходимо пересчитать все операции (фактически заново рассчитать оба раунда). В процессе выполнения операций будет заметен «лавинный» эффект, который будет по нарастающей изменять данные в сети от операции к операции.

После пересчета всех операций – сделать скриншот экрана, в котором обязательно должны быть видны счетчики и статус задания, а также поля с исходными данными и введенными значениями.

**Требования к отчету**

Отчет должен содержать следующие пункты:

– титульный лист;

– название, цель лабораторной работы;

– индивидуальное задание;

– ход выполнения работы, в котором должны быть размещены сделанные скриншоты;

– заключение и выводы (в выводах обязательно указать, как был достигнут лавинный эффект, какие изменения он вызвал, какой количественный эффект был получен в результате изменений, как был рассчитан этот эффект).

**Варианты индивидуальных заданий для самостоятельного выполнения**

Варианты задания генерируются программой случайным образом:

– раундовый ключ (в работе допускается, что в первых двух раундах ключи идентичны);

– два 4-байтных подблока ($L_1$, $R_1$);

– место внесения изменений (левый подблок или правый подблок или ключ, номер байта для изменения).

**Контрольные вопросы для самопроверки**

1. Какие требования к криптографическим алгоритмам вы знаете?
2. В чем выражается секретность криптографического алгоритма?
3. Что определяет криптостойкость криптографического алгоритма?
4. Что такое лавинный эффект? В чем его значение для криптографического алгоритма?
5. Какое количественное значение изменения выходных бит считается идеальным при изменении одного бита входных данных?
6. Как проанализировать стойкость криптографического алгоритма и проверить ее на практике?
7. Влияет ли функция Фейстеля на криптостойкость алгоритма? Как?
8. Должна ли функция Фейстеля быть обратимой?
9. Чем реализуется лавинный эффект в криптографическом алгоритме «Магма»?
10. Какие математические операции лежат в основе криптостойкости алгоритма?

11. В чем заключается различие основных понятий, определенных К. Шенноном: «диффузия» и «конфузия»?
12. Как изменится лавинный эффект, если функция Фейстеля будет линейной?

**Рекомендуемая литература**


1. ГОСТ Р 34.12–2015. Информационная технология. Криптографическая защита информации. Блочные шифры. – Москва : Стандартинформ, 2015. – 25 с.
2. ГОСТ Р 34.13–2015. Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров. – Москва : Стандартинформ, 2015. – 26 с.
3. Ищукова, Е. А. Современные методы криптоанализа блочных шифров : учебное пособие / Е. А. Ищукова. – Таганрог : Издательство ЮФУ, 2016. – 105 с.
4. Панасенко, С. П. Алгоритмы шифрования. Специальный справочник / С. П. Панасенко. – Санкт-Петербург : БХВ-Петербург, 2009. – 576 с.
5. Фомичев, В. М. Дискретная математика и криптология : учебное пособие / В. М. Фомичев ; под редакцией профессора Н. Д. Подуфалова. – Москва : Диалог-МИФИ, 2013. – 397 с.
6. Черемушкин, А. В. Методы и алгоритмы вычислений в криптографии : учебное пособие / А. В. Черемушкин. – Москва : Гелиос АРВ, 2023. – 416 с.


## 3.3. ЛАБОРАТОРНАЯ РАБОТА №3. АЛГОРИТМЫ БЛОЧНОГО ШИФРОВАНИЯ НА ОСНОВЕ СЕТИ ФЕЙСТЕЛЯ. ГОСТ 34.12–2015 («МАГМА»)

**Наименование работы:** «Алгоритмы блочного шифрования на основе сети Фейстеля. ГОСТ 34.12–2015 («Магма»)».

**Цель лабораторной работы:** изучить основные алгоритмы блочного шифрования, основанного на применении сети Фейстеля, их характеристики, достоинства и недостатки, применимость на практике. Освоить на практике

принципы алгоритма. В результате выполнения работы у студента должны сформироваться:

– знание стандартов ГОСТ 34.12-2015, 34.13-2015 для понимания требований к современным криптографическим алгоритмам

– знание областей применения различных криптографических алгоритмов;

– умение самостоятельно рассчитывать все этапы криптографического алгоритма (на примере алгоритма «Магма»).

**Задание на лабораторную работу**

Лабораторная работа включает в себя выполнение одного задания. Требуется реализовать студентом контроль работы алгоритма. Аналогично работе №2 для двух заданных (сгенерированных) 32-битных подблоков ($L_1$, $R_1$) программа вычисляет и визуализирует результаты всех этапов шифрования сетью Фейстеля (для двух первых раундов). Затем программа случайно добавляет в расчеты несколько ошибок, которые пользователь должен найти и исправить Задача студента – найти и исправить все ошибки в исходных вычислениях.

**Программно-аппаратное обеспечение**

Лабораторная работа выполняется на лабораторном персональном компьютере / ноутбуке. Для выполнения всех заданий используется специальное программное обеспечение лабораторного практикума – FNSC, установленное на лабораторном компьютере.

**Краткие теоретические сведения**

См. п. 1.2.

**Технология выполнения работы**

Для выполнения практической части лабораторной работы запускается программа FNSC, затем в меню выбора практических работ выбирается «Лабораторная работа №3» (рис 3.11).

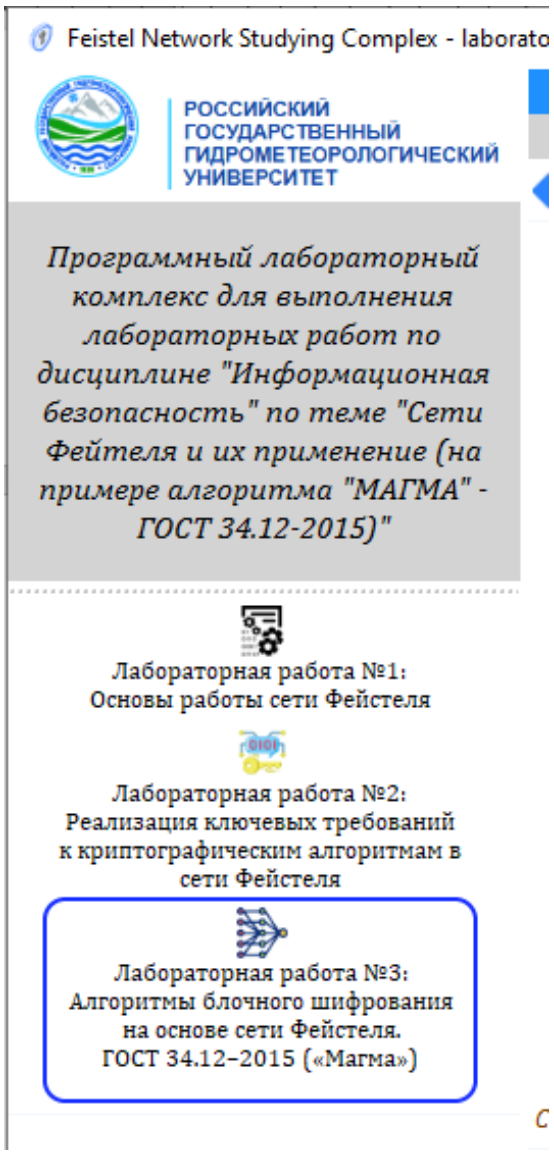


Рисунок 3.11 – Выбор третьей лабораторной работы

При выполнении заданий лабораторной работы следовать указаниям в программе. В задании предусматривается два счетчика: счетчик запусков и счетчик попыток. Идеальный случай – это когда для выполнения работы был один запуск и одна попытка. В этом случае за работу присваивается максимальные 100 баллов. В работе можно сгенерировать новые данные (вариант) – при этом счетчик запусков будет инкрементирован на единицу. Перезапуск является нежелательным и показывает, что студент не может справиться с работой. В этом случае в работе рекомендуется снять штрафные баллы – по 20 баллов за каждый перезапуск после первого. Выполненная работа отправляется на проверку с помощью соответствующей команды.

Результаты проверяются системой, при этом красным цветом подсвечиваются некорректные блоки, а зеленым – корректные. При этом для 32-битных блоков при наличии ошибки в одном бите красным будет подсвечиваться весь байт, содержащий ошибочный бит. Каждая неуспешная проверка нежелательна и показывает, что студент плохо усвоил материал или был невнимателен. Каждая неуспешная проверка увеличивает счетчик попыток. В этом случае в текущем задании рекомендуется снять штрафные баллы – по 5 баллов за каждую неудачную попытку после первой.

**Порядок выполнения работы**

После запуска третьей лабораторной работы с помощью специального программного обеспечения лабораторного практикума для выполнения открывается задание (рис. 3.12).

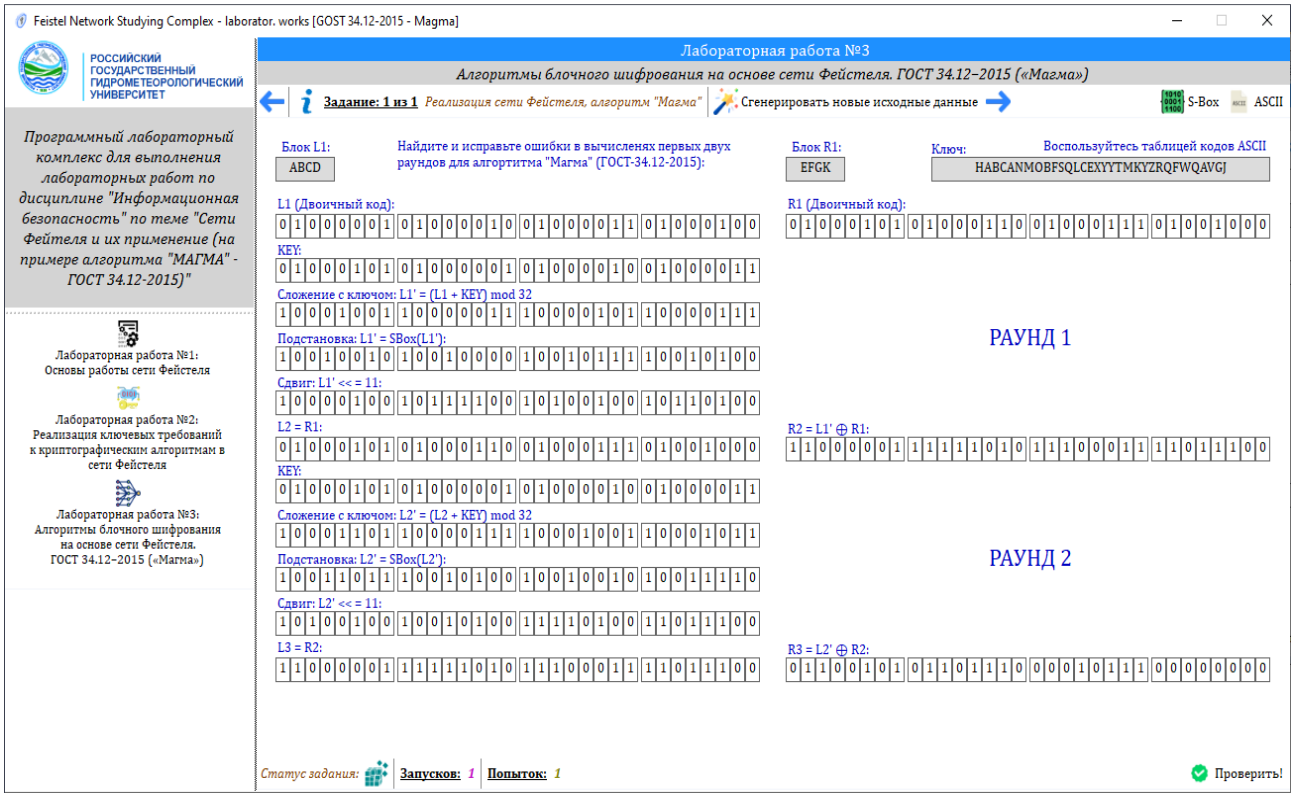


Рисунок 3.12 – Бланк выполнения лабораторной работы №3

Задача работы. Просмотреть результаты выполнения двух первых раундов алгоритма «Магма». Сделать скриншот экрана, в котором обязательно должны быть видны поля с исходными данными и рассчитанными значениями.

В приведенных вычислениях программой намеренно сделаны ошибки. Задача студента – найти все допущенные ошибки в вычислениях и исправить их.

После исправления ошибок – сделать скриншот экрана, в котором обязательно должны быть видны счетчики и статус задания, а также поля с исходными данными и введенными значениями.

**Требования к отчету**

Отчет должен содержать следующие пункты:

– титульный лист;

– название, цель лабораторной работы;

– индивидуальное задание;

– ход выполнения работы, в котором должны быть размещены сделанные скриншоты, а также перечень допущенных программой ошибок с указанием их локализации;

– заключение и выводы.

**Варианты индивидуальных заданий для самостоятельного выполнения**

Варианты задания генерируются программой случайным образом:

– базовый ключ (в отличие от лабораторной работы №2, в данной работе раундовые ключи должны быть определены на основании базового по правилам алгоритма);

– два 4-байтных подблока ($L_1$, $R_1$);

– места локализации намеренно внесённых в результаты вычислений ошибок.

**Контрольные вопросы для самопроверки**

1. Перечислите основные технические характеристики алгоритма «Магма» (ГОСТ 34.12-2015)?
2. В чем основные отличия алгоритмов «Магма» и «Кузнечик», описанных в ГОСТ 34.12-2015? Какова их сфера применения?
3. Опишите процесс получения раундовых ключей в алгоритме «Магма».

4. Чем отличается операция сложения по модулю 32 от обычного XOR?
5. Почему операция циклического сдвига в функции Фейстеля выполняется именно на 11 бит влево?

**Рекомендуемая литература**


1. ГОСТ Р 34.12–2015. Информационная технология. Криптографическая защита информации. Блочные шифры. – Москва : Стандартинформ, 2015. – 25 с.
2. ГОСТ Р 34.13–2015. Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров. – Москва : Стандартинформ, 2015. – 26 с.
3. Запечников, С. В. Криптографические методы защиты информации : учебник для академического бакалавриата / С. В. Запечников, О. В. Казарин, А. А. Тарасов. – Москва : Юрайт, 2019. – 309 с.
4. Адаменко, М. В. Основы классической криптологии: секреты шифров и кодов / М. В. Адаменко. – Москва : ДМК Пресс, 2014. – 256 с.
5. Иванов, М. А. Криптографические методы защиты информации в компьютерных системах и сетях / М. А. Иванов. – Москва : КУДИЦ-ОБРАЗ, 2001. – 368 с.
6. Харин, Ю. С. Математические и компьютерные основы криптологии : учебное пособие / Ю. С. Харин, В. И. Берник, Г. В. Матвеев, С. В. Агиевич. – Минск : Новое знание, 2003. – 382 с.
7. Молдовян, А. А. Криптография: скоростные шифры / А. А. Молдовян, Н. А. Молдовян, Н. Д. Гуц, Б. В. Изотов. – Санкт-Петербург : БХВ-Петербург, 2002. – 496 с.

# ЗАКЛЮЧЕНИЕ

В криптографии одним из наиболее надежных методов преобразования данных является шифрование. При этом выделяются как класс блочные шифры. Они высоконадежные, предназначены для безопасного хранения данных. Многие современные алгоритмы блочного шифрования (Blowfish, ГОСТ 34.12–2015 («Магма»), CAST-128, XTEA) реализованы на архитектуре преобразований сетью Фейстеля. В ходе работы был проведен сравнительный анализ разных алгоритмов блочного шифрования. При этом было установлено, что алгоритм «Магма», опубликованный в ГОСТ 34.12–2015, основан на работе сети Фейстеля и является государственным стандартом РФ, включенным в перечень обязательных требований к построению современных систем, обеспечивающих промышленную безопасность и доверенную инфраструктуру государственного сектора. Следовательно, основы и принципы работы сети Фейстеля для студентов, изучающих основы информационной безопасности и криптографии, рекомендуется проводить именно с учетом ее применения к алгоритму ГОСТ 34.12–2015: «Магма».

Основываясь на теоретическом исследовании алгоритмов блочного шифрования, архитектурно-базирующихся на применении сети Фейстеля, их многообразии и областях применения, были определены основные цели для лабораторного практикума по дисциплине «Методы и средства криптографической защиты информации». Одной из задач этого практикума было определено наличие специализированного программного обеспечения лабораторного практикума – программного комплекса для выполнения практических заданий лабораторного практикума по дисциплине. Для решения это задачи была поставлена цель разработки такого практикума.

В рамках решения задачи разработки проекта системы обеспечения проектного практикума был разработан комплекс моделей на языке UML, включающих диаграммы прецедентов и компонентов. Первая позволила формализовать функциональные требования к системе, вторая – раскрыть архитектуру ее реализации.

Разработанный технический проект системы FNSC (Feistel Network Studying Complex) позволил выполнить рабочую версию приложения. Для разработки приложения были выбран следующий стек технологий и средств: MS Visual Studio 2019 / C#.

В качестве методического обеспечения был разработан комплекс из трех лабораторных работ, охватывающих теоретические и практически знания и умения области криптографии на основе алгоритма «Магма», включенного в национальный ГОСТ 34.12–2015. Задачи предложенных работ охватывают весь спектр тех положений и способствуют формированию тех компетенций, которыми должен обладать канонический специалист-выпускник по дисциплине «Методы и средства криптографической защиты информации». Практические задания, предусмотренные в лабораторном практикуме, ориентированы на использование разработанного специального программного обеспечения – FNSC.

Контрольный пример работы приложения FNSC показал, что все заявленные в практикуме лабораторные работы отражены в методическом программном обеспечении, это программное обеспечение предлагает удобные и функциональные инструменты для выполнения специальных заданий лабораторных работ. Индивидуализация заданий выполнена посредством рандомизации исходных данных.

Таким образом, разработанный практикум обеспечивает получение студентами необходимых компетенций, а созданное программное обеспечение позволяет достичь целей, предусмотренных в практикуме. На этом основании можно сделать вывод о том, что цель выпускной квалификационной работы достигнута, а все ее задачи – решены.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Николаева, М. О. Информационная безопасность: современная картина проблемы информационной безопасности и защиты информации / М. О. Николаева // Мониторинг. Образование. Безопасность. – 2023. – № 1(1). – С. 51-57.

2. Старовойтова, М. М. DLP-системы для предотвращения утечек конфиденциальной информации / М. М. Старовойтова, А. Б. Макарец // Математика и математическое моделирование : Сборник материалов XIX Всероссийской молодёжной научно-инновационной школы, Саров, 09–11 апреля 2025 года. – Саров: ООО "Интерконтакт, 2025. – С. 56-58.

3. Фот, Ю. Д. Интеграция SIEM-систем в комплексную защиту информационных систем / Ю. Д. Фот, Д. А. Семин // Актуальные вопросы обеспечения комплексной безопасности : Материалы национальной научно-практической конференции с международным участием, посвященной 35-летию МЧС России и 95-летию Оренбургского ГАУ, Оренбург, 23 мая 2025 года. – Оренбург: Оренбургский государственный аграрный университет, 2025. – С. 1481-1485.

4. Олимов, Н. А. Криптографические методы защиты информации / Н. А. Олимов // Наука и инновация. – 2024. – № 3. – С. 74-80.

5. Лушников, Н. Д. Кодирование и шифрование как основы криптографии / Н. Д. Лушников, А. Д. Альтерман // Аллея науки. – 2018. – Т. 3, № 6(22). – С. 917-920.

6. ГОСТ Р 34.12-2015. Информационная технология. Криптографическая защита информации. Блочные шифры [Текст]. – Введ. 2016–01–01. – М. : Стандартинформ, 2015. – III, 17 с.

7. Данилова, О. Ю. Об использовании сети Фейстеля в современных криптоалгоритмах / О. Ю. Данилова // Общественная безопасность, законность и правопорядок в III тысячелетии. – 2021. – № 7-3. – С. 28-33.

8. Муравьев, Е. В. Использование сети Фейстеля в алгоритмах шифрования / Е. В. Муравьев, О. Ю. Данилова // Актуальные вопросы

эксплуатации систем охраны и защищенных телекоммуникационных систем : сборник материалов Всероссийской научно-практической конференции, Воронеж, 07 июня 2018 года. – Воронеж: Воронежский институт Министерства внутренних дел Российской Федерации, 2018. – С. 210-212.

9. Короплясов, А. В. История создания и обзор архитектуры сетей Фейстеля / А. В. Короплясов, В. В. Румбешт // Вестник магистратуры. – 2012. – № 1. – С. 20-22.

10. Куринная, Ю. С. Сравнительный анализ блочных шифров, описанных в ГОСТ 34.12-2015 / Ю. С. Куринная, Е. В. Андреева // Электронный научный журнал. – 2015. – № 3(3). – С. 73-77. – DOI 10.18534/enj.2015.03.73.

11. Огриско, Е. А. Сравнительный анализ блочных алгоритмов шифрования / Е. А. Огриско, А. В. Свирь, В. О. Антонов // Студенческая наука для развития информационного общества : сборник материалов VI Всероссийской научно-технической конференции, Ставрополь, 22–26 мая 2017 года. Том Часть 2. – Ставрополь: Северо-Кавказский федеральный университет, 2017. – С. 297-299.

12. Перевозник, Ю. Я. Современные отечественные алгоритмы симметричного шифрования, их сравнительный анализ и перспективы применения / Ю. Я. Перевозник, А. П. Фоменкова // Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2023) : Сборник научных статей XII Международной научно-технической и научно-методической конференции. В 4-х томах, Санкт-Петербург, 28 февраля – 01 2023 года / Под редакцией С.И. Макаренк, сост. В.С. Елагин, Е.А. Аникевич. Том 2. – Санкт-Петербург: Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича, 2023. – С. 837-840.

13. Рогов В.А. Средства автоматизации и управления [Текст]: учебник / В.А. Рогов, А.Д. Чудаков. – М.: Издательство Юрайт, 2017. – 404 с.Object Management Group Inc., Unified Modeling Language (UML) Ver. 2.5.1

Infrastructure Specification: OMG Document number: formal/2017-12-05 URL: https://www.omg.org/spec/UML. – December, 2017. – 796 p.

14. ISO/IEC/IEEE 29148:2018, International Standard – Systems and software engineering – Life cycle processes – Requirements engineering, 01 December 2018.

15. Благодатских, В.А. Стандартизация разработки программных средств: Учебное пособие для вузов / В. А. Благодатских, В. А. Волнин, К. Ф. Поскакалов; под ред. О. С. Разумова. - М.: Финансы и статистика, 2017 г. – 288 с.

16. SOLID (ООП). [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/SOLID_(объектно_ориентированное_программирование) (дата обращения: 17.12.2025 г).

17. Буч Г. UML. Классика CS / Буч Г., Якобсон А., Рамбо Дж.; пер. с англ.; под общей ред. проф. С. Орлова. ¬– СПб.: Питер, 2009. – [2-е изд.]. – 736 с.

18. Visual Studio IDE, Редактор кода [электронная документация] // MSDN. Комплекс технической документации по продуктам Microsoft. URL: https://docs.microsoft.com/ru-ru/windows/uwp/get-started/ (дата обращения: 20.12.2025 г).

19. Джозеф Албахари – C#. Справочник. Полное описание языка. Пер. с англ. - М: ООО «И.Д. Вильямс», 6-е изд., 2019, 1040 с.

20. Вершинин М., Иванова Е. C# Enterprise Edition. Технологии проектирования и разработки. – М.: BHV, 2023 г. – 1088 с.