**Муниципальное бюджетное общеобразовательное учреждение**

**«Школа № 16» города Сарова**

Проект на тему:

**«Шифрование информации»**

Информационный проект

Работу выполнила

ученица 10 «Б» класса

Бликова Анна Антоновна

Руководитель:

Галкина Инна Владимировна,

учитель информатики

МБОУ Школы №16

Город Саров

# Оглавление

[Введение 4](#_Toc133188426)

[Глава 1. Теоритическая часть 5](#_Toc133188427)

[1.1История криптографии 5](#_Toc133188428)

[1.2Виды шифрования 7](#_Toc133188429)

[1.3 Обоснование выбора вида шифрования 9](#_Toc133188442)

[1.5Языки программирования 10](#_Toc133188443)

[Глава 2. 12](#_Toc133188444)

[Практическая часть. Создание программы и презентации 12](#_Toc133188445)

[3.Заключение 18](#_Toc133188446)

[Список литературы 19](#_Toc133188447)

[Приложение 1 20](#_Toc133188448)

# Введение

В наше время вопрос безопасности данных в интернете стоит особенно остро. Многие люди сталкиваются с проблемой утечки данных и их дальнейшим использованием в нежелательных целях мошенниками.

Однажды, используя для переписки с подругой электронную почту, я столкнулась с проблемой утечки данных. Впоследствии мошенники использовали мои личные данные для финансового вымогательства. Впредь я захотела повысить безопасность личной переписки.

Я предполагаю, что смогу создать такой метод шифрования, позволяющий составить сообщение, которое не будет понятным посторонним людям.

Перед собой я поставила цель: разработать такой метод шифрования, который не будут понимать посторонние людям.

Мне было необходимо получить базовые знания по теме шифрование, особенности и характеристики шифров, среди изученных методов выбрать лучший. Далее с использованием выбранного метода реализовать собственный метод шифрования. В заключение осуществить сначала внутреннее, а потом и внешнее тестирование, передав зашифрованное сообщение адресату(подруге).

Перед собой я поставила задачи:

1. Изучить методы шифрования
2. Проанализировать полученную информацию
3. Обобщить полученную информацию
4. Разработать собственный метод шифрования

Для реализации поставленных задач я использовала такие методы как: анализ

синтез, тестирование, работа с источниками интернета и литературой, эксперимент, сравнение и систематизацию. Все поставленные задачи были выполнены в полном объеме и в срок.

# Глава 1. Теоретическая часть

## 1.1История криптографии

Криптография — наука о методах обеспечения [конфиденциальности](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%84%D0%B8%D0%B4%D0%B5%D0%BD%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C), [целостности данных](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D0%BE%D1%81%D1%82%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%B8), [аутентификации](https://ru.wikipedia.org/wiki/%D0%90%D1%83%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F) и [шифрования](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5). Имеются свидетельства, что криптография как техника защиты текста возникла вместе с письменностью, и способы тайного письма были известны уже древним цивилизациям [Индии](https://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B5%D0%B2%D0%BD%D1%8F%D1%8F_%D0%98%D0%BD%D0%B4%D0%B8%D1%8F), [Египта](https://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B5%D0%B2%D0%BD%D0%B8%D0%B9_%D0%95%D0%B3%D0%B8%D0%BF%D0%B5%D1%82) и [Месопотамии](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%81%D0%BE%D0%BF%D0%BE%D1%82%D0%B0%D0%BC%D0%B8%D1%8F). Криптография появилась около 4 тысяч лет назад. Историю криптографии можно поделить на несколько периодов.

Первый период развития начался приблизительно с 3-го тысячелетия до н. э. и характеризовался господством моноалфавитных шифров, смысл которых заключался в замене алфавита исходного текста другим алфавитом через замену букв другими буквами или символами. Например, в [древнеиндийских текстах](https://ru.wikipedia.org/wiki/%D0%9A%D0%B0%D0%BC%D0%B0%D1%81%D1%83%D1%82%D1%80%D0%B0) среди 64 искусств названы способы изменения текста, некоторые из них можно отнести к криптографическим. Автор таблички с рецептом для изготовления [глазури](https://ru.wikipedia.org/wiki/%D0%93%D0%BB%D0%B0%D0%B7%D1%83%D1%80%D1%8C) для гончарных изделий из Месопотамии использовал редкие обозначения, пропускал буквы, а имена заменял на цифры, чтобы скрыть написанное. В дальнейшем встречаются различные упоминания об использовании криптографии, большая часть относится к использованию в военном деле.

Второй период (с [IX века](https://ru.m.wikipedia.org/wiki/IX_%D0%B2%D0%B5%D0%BA) на Ближнем Востоке и с [XV века](https://ru.m.wikipedia.org/wiki/XV_%D0%B2%D0%B5%D0%BA) в Европе — до начала [XX века](https://ru.m.wikipedia.org/wiki/XX_%D0%B2%D0%B5%D0%BA)) ознаменовался введением в обиход полиалфавитных шифров. В шестидесятых годах XV века итальянский учёный и архитектор [Леон Баттиста Альберти](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D1%8C%D0%B1%D0%B5%D1%80%D1%82%D0%B8,_%D0%9B%D0%B5%D0%BE%D0%BD_%D0%91%D0%B0%D1%82%D1%82%D0%B8%D1%81%D1%82%D0%B0), стремясь получить устойчивый к частотному криптоанализу шифр, одним из первых начал применять вместо одного секретного алфавита два или более, переходя от одного к другому по некоторому правилу. Полиалфавитные шифры долгое время не получали широкого распространения в связи со сложностью их использования. Для повседневных целей обычных людей было достаточным использование моноалфавитных шифров, а для военных и дипломатических целей государство большое внимание уделяло скорости и простоте шифрования. Со временем стали появляться Чёрные кабинеты — органы, занимающийся [перлюстрацией](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%BB%D1%8E%D1%81%D1%82%D1%80%D0%B0%D1%86%D0%B8%D1%8F) и [дешифрованием](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F#Терминология) корреспонденции; помещением, служащеим для этих целей, обычно была тайная комната в почтовом отделении. С развитием [«Черных кабинетов»](https://ru.wikipedia.org/wiki/%D0%A7%D1%91%D1%80%D0%BD%D1%8B%D0%B9_%D0%BA%D0%B0%D0%B1%D0%B8%D0%BD%D0%B5%D1%82) в странах [Европы](https://ru.wikipedia.org/wiki/%D0%95%D0%B2%D1%80%D0%BE%D0%BF%D0%B0) в [XVIII веке](https://ru.wikipedia.org/wiki/XVIII_%D0%B2%D0%B5%D0%BA) все шифры моноалфавитной замены потеряли всякую надежность. Этот факт способствовал вынужденному переходу к использованию полиалфавитных шифров. Другой причиной популяризации более сложного вида шифрования стало развитие [телеграфа](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D0%B3%D1%80%D0%B0%D1%84) и возникшая необходимость в защите сообщений от перехвата. Шифрование начинает широко использоваться дипломатами, купцами и даже простыми гражданами.

С началом внедрения электромеханических устройств в шифрование начался третий период, при этом использование полиалфавитных шифров продолжалось. К началу 1930-х годов окончательно сформировались разделы математики, являющиеся основой для будущей науки — общая [алгебра](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%B5%D0%B1%D1%80%D0%B0), [теория чисел](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D1%87%D0%B8%D1%81%D0%B5%D0%BB), [теория вероятностей](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B2%D0%B5%D1%80%D0%BE%D1%8F%D1%82%D0%BD%D0%BE%D1%81%D1%82%D0%B5%D0%B9) и [математическая статистика](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%81%D1%82%D0%B0%D1%82%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0). К концу 1940-х годов построены первые программируемые счётные машины, заложены основы [теории алгоритмов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%BE%D0%B2), [кибернетики](https://ru.wikipedia.org/wiki/%D0%9A%D0%B8%D0%B1%D0%B5%D1%80%D0%BD%D0%B5%D1%82%D0%B8%D0%BA%D0%B0). Тем не менее в период после Первой мировой войны и до конца 1940-х годов в открытой печати было опубликовано совсем немного работ и монографий, но и те отражали далеко не самое актуальное состояние дел. Наибольший прогресс в криптографии достигается в военных ведомствах.

Ключевой вехой в развитии криптографии является фундаментальный труд [Клода Шеннона](https://ru.wikipedia.org/wiki/%D0%A8%D0%B5%D0%BD%D0%BD%D0%BE%D0%BD,_%D0%9A%D0%BB%D0%BE%D0%B4_%D0%AD%D0%BB%D0%B2%D1%83%D0%B4) «[Теория связи в секретных системах](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D1%81%D0%B2%D1%8F%D0%B7%D0%B8_%D0%B2_%D1%81%D0%B5%D0%BA%D1%80%D0%B5%D1%82%D0%BD%D1%8B%D1%85_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%D1%85)» — секретный доклад, представленный автором в [1945 году](https://ru.wikipedia.org/wiki/1945_%D0%B3%D0%BE%D0%B4) и опубликованный им в «[Bell System Technical Journal](https://ru.wikipedia.org/w/index.php?title=Bell_System_Technical_Journal&action=edit&redlink=1)» в [1949 году](https://ru.wikipedia.org/wiki/1949_%D0%B3%D0%BE%D0%B4). В этой работе, по мнению многих современных криптографов, был впервые показан подход к криптографии в целом как к [математической](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0) науке. Были сформулированы её теоретические основы и введены понятия, с объяснения которых сегодня начинается изучение криптографии студентами.

Четвёртый период — с середины до 70-х годов XX века — период перехода к математической криптографии. Появляются строгие математические определения [количества информации](https://ru.m.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B1%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D1%8F), передачи данных, функций шифрования. До [1975 года](https://ru.m.wikipedia.org/wiki/1975_%D0%B3%D0%BE%D0%B4) в криптографии использовался секретный ключ. В криптографии с секретным ключом один [ключ](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Key) используется для выполнения функций как [шифрования](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Encryption), так и [дешифрования](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Decryption). [Зашифрованное](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Encrypted) [сообщение](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Message) может быть свободно отправлено из одного места в другое через небезопасный носитель. Как следует из названия, криптография с секретным ключом зависит от того, что обе стороны хранят [ключ](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Key) в [секрете](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Secret). Если этот [ключ](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Key) рассекречен, [безопасность](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Security), обеспечиваемая процессом [шифрования](https://translated.turbopages.org/proxy_u/en-ru.ru.ece43adb-642dafe3-49b8cc0f-74722d776562/https/itlaw.fandom.com/wiki/Encryption), устраняется.

Современный, то есть пятый, период развития криптографии (с конца 1970-х годов по настоящее время). В [1976 году](https://ru.wikipedia.org/wiki/1976_%D0%B3%D0%BE%D0%B4) публикуется работа [Уитфилда Диффи](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%84%D1%84%D0%B8,_%D0%A3%D0%B8%D1%82%D1%84%D0%B8%D0%BB%D0%B4) и [Мартина Хеллмана](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D0%BB%D0%BB%D0%BC%D0%B0%D0%BD,_%D0%9C%D0%B0%D1%80%D1%82%D0%B8%D0%BD) «Новые направления в криптографии». Данная работа открыла новую область в криптографии, теперь известную как [криптография с открытым ключом](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F_%D1%81_%D0%BE%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%BC_%D0%BA%D0%BB%D1%8E%D1%87%D0%BE%D0%BC). Криптографическая система с открытым ключом— система [электронной подписи](https://ru.m.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%BF%D0%BE%D0%B4%D0%BF%D0%B8%D1%81%D1%8C), при которой открытый ключ передаётся по незащищённому, доступному для наблюдения каналу и используется для проверки электронной подписи и для шифрования сообщения. Одним из результатов публикации стал значительный рост числа людей, занимающихся криптографией. Совсем недавно криптографические методы также использовались для создания криптовалют. Криптовалюты используют несколько передовых криптографических методов, включая хэш-функции, криптографию с открытым ключом и цифровые подписи. Эти методы используются главным образом для обеспечения безопасности данных, хранящихся в блокчейне, а также для аутентификации транзакций. Специализированная форма криптографии, известная как ECDSA - алгоритм для создания цифровой подписи, определенной в группе точек эллиптической кривой, используется в Биткоине и других криптовалютных системах как средство обеспечения дополнительной безопасности и гарантии того, что средствами могут пользоваться только их законные владельцы.

Криптография прошла долгий путь за последние 4000 лет, и вряд ли он скоро прекратится. Пока конфиденциальные данные требуют защиты, криптография будет продолжать развиваться. Криптографические системы, используемые в блокчейне криптовалют сегодня, представляют одну из наиболее продвинутых форм этой науки. Они также являются частью традиционной истории человечества.

По мере развития криптографии появлялось все больше и больше различных видов шифрования, имеющих значение и применение по сей день. Для достижения цели проекта и выполнения всех задач мне необходимо подробнее ознакомится с практичными видами шифрования, которые я смогу применить в переписке с подругой.

## 1.2Виды шифрования

Для начала я бы хотела показать вам шифры из двух классов шифрования: моноалфавитный и полиалфавитный.

1. Моноалфавитный шифр – класс методов шифрования, в которых текст шифруется по особому правилу. В основном используется замена положений букв в алфавите или в таблице символов. Затем каждую букву текста смотрят по положению в алфавите и меняют на другую букву шифрованного алфавита в том же положении. Таким простым способом шифруется весь текст. К такому принципу работы относится программа ROT13, шифр Цезаря, Атбаш. Данный способ шифрования является очень простым и отлично реализуемым.
2. Полиалфавитный представляет класс шифров, в которых шифрование символа чередуется последовательностью моноалфавитных шифров. На каждый символ в тексте приходится свой простой метод замены текста. Так перечисленные способы моноалфавитного шифрования можно чередовать в тексте.

### Шифр Атбаша

### «Атба́ш» – один из самых древних методов шифрования. Шифрование заключается в замене каждой буквы исходного текста на «симметричную» ей букву алфавита, то есть первая алфавита заменялась на последнюю и наоборот, вторая буква – на предпоследнюю и наоборот и т.д. Простой в использовании, но и такой же простой в дешифровке. Все, что вам нужно сделать для расшифровки, это создать таблицу перевода с буквами алфавита, написанными от А до Я вверху и перевернутыми внизу. Найти букву в вашем зашифрованном тексте в нижней строке и посмотреть над ней, чтобы увидеть, как она расшифрована.

### Шифр кодового слова

### Принцип шифрования примерно такой же, как у шифра цезаря. Только в этом случае мы сдвигаем алфавит не на определенное число позиций, а на кодовое слово.

ридумаем кодовое слово. Например, «Яблоко». Выдернем из него все повторяющиеся символы. На выходе получаем слово «Яблок».  
Теперь вписываем данное слово в начале алфавита, а остальные символы оставляем без изменений.

Исходный: абвгдеёжзийклмнопрстуфхцчшщъыьэюя

### Шифрованный:яблокавгдеёжзиймнпрстуфхцчшщъыьэю

### Шифр Цезаря

### Метод шифрования назван носит имя римского императора Юлия Цезаря (I век до н. э.), который если и не сам изобрел его, то активно им пользовался. Этот метод основан на замене каждой буквы шифруемого текста на другую путем смещения в алфавите от исходной буквы на фиксированное количество символов, причем алфавит читается по кругу, то есть после буквы я рассматривается а. Тут добавляется еще один параметр — примитивный ключ в виде числа от 1 до 25 (для латиницы). На практике, ключ будет от 4 до 10.

Например, ниже представлено смещение алфавита на 3 символа:

Иcходный: абвгдеёжзийклмнопрстуфхцчшщъыьэюя

Шифрованный: гдеёжзийклмнопрстуфхцчшщъыьэюяабв

Малое пространство ключей (всего 25 вариантов) делает брут-форс самым эффективным и простым вариантом атаки.

### Для вскрытия необходимо каждую букву шифр текста заменить буквой, стоящей на один знак левее в алфавите. Если в результате этого не удалось получить читаемое сообщение, то необходимо повторить действие, но уже сместив буквы на два знака левее. И так далее, пока в результате не получится читаемый текст

### Шифр Вернама (XOR-шифр)

### Простейший шифр на основе бинарной логики, который обладает абсолютной криптографической стойкостью. Без знания ключа, расшифровать его невозможно (доказано Клодом Шенноном). Особенность шифрования методом Виженера (шифром Виженера) заключается в том, что каждая буква «сдвигается» не на одно и то же количество позиций, как в шифре Цезаря. Таких сдвигов — несколько. Размер этих сдвигов определяется некоторым ключом шифрования — словом.

### Шифр кодового слова

Принцип шифрования примерно такой же, как у шифра цезаря.. Идея заключается в том, что выбирается [кодовое слово](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D0%BE%D0%B2%D0%BE%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2%D0%BE), которое пишется впереди, затем выписываются остальные буквы алфавита в своем порядке. Пространство ключей шифра простой замены огромно и равно количеству перестановок используемого алфавита. Так для английского языка это число составляет 26! = 288. Разумеется наивный перебор всех возможных ключей дело безнадежное и для взлома потребуется более утонченная техника, такая как поиск восхождением к вершине:

1. Выбирается случайная последовательность букв — основной ключ. Шифртекст расшифровывается с помощью основного ключа. Для получившегося текста вычисляется коэффициент, характеризующий вероятность принадлежности к естественному языку.
2. Основной ключ подвергается небольшим изменениям (перестановка двух произвольно выбранных букв). Производится расшифровка и вычисляется коэффициент полученного текста.
3. Если коэффициент выше сохраненного значения, то основной ключ заменяется на модифицированный вариант.
4. Шаги 2-3 повторяются пока коэффициент не станет постоянным.

### Шифр Плейфера

Шифр Плейфера использует матрицу 5х5 (для латинского алфавита, для кириллического алфавита необходимо увеличить размер матрицы до 4х8), содержащую ключевое слово или фразу. Для создания матрицы и использования шифра достаточно запомнить ключевое слово и четыре простых правила. Чтобы составить ключевую матрицу, в первую очередь нужно заполнить пустые ячейки матрицы буквами ключевого слова (не записывая повторяющиеся символы), потом заполнить оставшиеся ячейки матрицы символами алфавита, не встречающимися в ключевом слове, по порядку (в английских текстах обычно опускается символ «Q», чтобы уменьшить алфавит, в других версиях «I» и «J» объединяются в одну ячейку). Ключевое слово может быть записано в верхней строке матрицы слева направо, либо по спирали из левого верхнего угла к центру. Ключевое слово, дополненное алфавитом, составляет матрицу 5х5 и является ключом шифра.

Для того чтобы зашифровать сообщение, необходимо разбить его на биграммы (группы из двух символов), например «Hello World» становится «HE LL OW OR LD», и отыскать эти биграммы в таблице. Два символа биграммы соответствуют углам прямоугольника в ключевой матрице. Определяем положения углов этого прямоугольника относительно друг друга. Затем, руководствуясь следующими 4 правилами, зашифровываем пары символов исходного текста:

1. Если два символа биграммы совпадают (или если остался один символ), добавляем после первого символа «Х», зашифровываем новую пару символов и продолжаем. В некоторых вариантах шифра Плейфера вместо «Х» используется «Q».
2. Если символы биграммы исходного текста встречаются в одной строке, то эти символы замещаются на символы, расположенные в ближайших столбцах справа от соответствующих символов. Если символ является последним в строке, то он заменяется на первый символ этой же строки.
3. Если символы биграммы исходного текста встречаются в одном столбце, то они преобразуются в символы того же столбца, находящиеся непосредственно под ними. Если символ является нижним в столбце, то он заменяется на первый символ этого же столбца.
4. Если символы биграммы исходного текста находятся в разных столбцах и разных строках, то они заменяются на символы, находящиеся в тех же строках, но соответствующие другим углам прямоугольника.

Для расшифровки необходимо использовать инверсию этих четырёх правил, откидывая символы «Х» (или «Q»), если они не несут смысла в исходном сообщении.

**Шифр Бэкона**

В этом шифре каждая буква текста заменяется группой из пяти букв «А» или «В»  
(двоичный шифр). Замена осуществляется с помощью специального алфавита шифра  
Бэкона. Автор должен использовать два разных [шрифта](https://translated.turbopages.org/proxy_u/en-ru.ru.2931a765-642f3762-d4af3670-74722d776562/https/en.wikipedia.org/wiki/Typeface) для этого шифра. После подготовки ложного сообщения с тем же количеством букв, что и все *As* и *Bs* в настоящем секретном сообщении, выбираются два шрифта: один для представления *As*, а другой *Bs*. Затем каждая буква ложного сообщения должна быть представлена соответствующим шрифтом, в зависимости от того, означает ли она *A* или *B*. Для расшифровки сообщения применяется обратный метод. Каждая буква "шрифт 1" в ложном сообщении заменяется на *A*, а каждая буква "шрифт 2" заменяется на *B*. Затем для восстановления исходного сообщения используется алфавит Бэкона.

**По**дводя итоги, можно сделать вывод, что уже более трех тысяч лет, с момента появления криптографии, появилось множество методов шифрования. Криптография и шифрование имеют огромное значение и по сей день, шифрование используется как профессионалами, так и обычными людьми. Криптография развивается, появляется все больше возможностей для сохранения конфиденциальности своих личных данных, однако при этом растет и опасность их утечки в руки мошенников.

## 1.3 Обоснование выбора вида шифрования

Изучив самые известные виды шифрования, я решила, что буду создавать метод шифрования, комбинирующий два вида шифрования: шифр Цезаря и шифр с использованием кодового слова. Приняла я такое решение по двум причинам:

1. Мой метод шифрования должен быть прост в написании. Я новичок в программировании, поэтому буду писать код, исходя из своих возможностей. Я выбрала одни из самых простых для написания шифров: шифр Цезаря и шифр с использованием кодового слова.
2. Объединяя два шифра в один, я обеспечиваю большую безопасность своему шифру, таким образом усложняю его расшифровку.

## 1.5Языки программирования

Следующим этапом моей работы является выбор самого удобного для реализации моей программы языка программирования. Моя осведомленность о различных языках программирования и опыт работы с ними отличаются, поэтому я решила подробнее изучить достоинства и недостатки уже известных мне языков программирования, и выбрать наиболее подходящий лично мне.

**С++**

C++— [компилируемый](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D0%B8%D1%80%D1%83%D0%B5%D0%BC%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), [статически типизированный](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) [язык программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) общего назначения. Широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также компьютерных игр. Данный язык программирования имеет достаточное количество плюсов:

* высокая скорость.
* Возможности для работы с данными на низком уровне — то есть на уровне, близком к аппаратному. Благодаря этому на С++ можно писать драйвера, микроконтроллеры.
* Для С++ создано много библиотек и компиляторов.
* С++ используется практически везде *(несколько примеров мы уже привели выше).*
* Синтаксис С++ похож на синтаксис C, С# и Java, так что переключаться между этими языками достаточно легко.
* Совместимость с C благодаря тому, что С++ создавался на его основе.
* Возможность работать с указателями дает c++ преимущество. С другой стороны, при программировании на **С++** работа с указателями неизбежна, то есть разбираться с ними придётся волей-неволей. Неправильное использование указателей может привести к неприятным последствиям, как, например, крах программы или искажение памяти.

**Pascal**

Pascal – язык программирования, используемый новичками. Он подходит для изучения основ алгоритмизации. Практическое применение языка Паскаль в обыденной жизни сильно ограничено. В своем «чистом» виде он используется крайне редко, в основном, для обучения. Более опытным разработчикам Паскаль не нужен. Так как язык создавался еще в 1960-х, сейчас он устарел и имеет такие недостатки:

1. Сильная ограниченность взаимодействия с компонентами, написанными на других языках программирования.
2. Меньшая приспособленность для решения некоторых узкоспециализированных задач.
3. Ограниченные возможности работы с динамической памятью, а также массивами с переменной длиной.
4. Относительно небольшая функциональность – для большинства современных приложений и игр встроенных возможностей Паскаля будет не хватать.

**Python**

Python— [высокоуровневый язык программирования](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) общего назначения с [динамической](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) [строгой](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%BE%D0%B3%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости [кода](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) и его качества, а также на обеспечение переносимости написанных на нём программ. Python — это высокоуровневый язык программирования, с помощью которого создают сайты, разрабатывают приложения, автоматизируют процессы анализа или визуализации данных. Python не был разработан для конкретных целей, поэтому подходит как для создания алгоритма рекомендаций видеосервиса, так и для разработки программного обеспечения для самоуправляемых автомобилей или управления космическими аппаратами на других планетах. К плюсам языка относится легкое изучение и простой минималистичный синтаксис, а также большая стандартная библиотека. Однако он имеет маленькую скорость, из-за динамической типизации вероятность ошибки при запуске становится выше, а также не подходит для работы с памятью на низком уровне.

**Java**

Java — [строго](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%B8_%D1%81%D0%BB%D0%B0%D0%B1%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) [типизированный](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) [объектно-ориентированный язык программирования](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) общего назначения. У всех качеств Java, будь то строгая типизация или объектная ориентированность, есть свои плюсы и минусы, а ещё они есть у самой Java как у языка. Плюсами языка являются:

1. Независимость — ваш код будет работать на любой платформе, которая поддерживает Java.
2. Надёжность — в немалой мере достигается благодаря строгой статической типизации.
3. Мультифункциональность.

Синтаксис Java похож на синтаксисы C и С++. У всех этих языков достаточно строгий и «многословный» синтаксис, и для новичков это минус: вместо того чтобы сосредоточиться на том, **что писать,** приходится больше думать о том, **как писать.**

**1**.6Обоснование выбора языка программирования

Таким образом, я решила, что буду использовать уже известный мне язык программирования С++. Он имеет множество плюсов, среди которых для меня наиболее важны высокая скорость и большое количество библиотек, а значит большое количество путей воплощения моей идеи. Также свою роль сыграло то, что я уже пробовала программировать на этом языке, а поэтому мне будет легче работать со сложным синтаксисом C++. Мне интересно попробовать написать код на популярном и дающим множество возможностей языке.

# Глава 2.

## Практическая часть. Создание программы и презентации

После изучения и анализа информации про методы шифрования и пути их реализации я приступила к созданию собственного шифра. Подытожив изученный материал, я сделала вывод, что буду создать свой метод шифрования на основе шифра Цезаря, то есть сдвига на некоторое количество символов, и шифра с использованием кодового слова. Мой код будет написан на языке программирования C++.

Первым этапом на пути к созданию шифра стала разработка алгоритма для кодировки. Заранее написанный алгоритм помогает структурировать идеи и проще воплотить их в самой программе.

Таким образом, мой метод шифрования будет проходить в несколько этапов:

1. Шифрование по алфавиту с кодовым словом, вводимым в ручную. Кодовое слово будет сообщаться адресату лично для дальнейшей расшифровки. В моем случае было принято решение договариваться с подругой, то есть адресатом, о использовании определенного кодового слова в течении недели.
2. Уже зашифрованное с помощью кодового слова сообщение будет зашифровано с помощью сдвига алфавита на 3 знака, то есть третий символ зашифрованного алфавита станет первым, а первый — тридцать первым.

Следующим этапом моей работы стала поэтапная разработка алгоритма самого кода. Проанализировав цели и задачи своего метода шифрования, я создала следующий алгоритм работы моего кода:

1. Ввод кодового слова
2. Подготовка алфавита, зашифрованного кодовым словом
3. Ввод сообщения для шифровки
4. Сдвиг каждой буквы сообщения на три значения влево в соответствии с зашифрованным алфавитом
5. Вывод зашифрованного сообщения

Такой же алгоритм я написала для программы дешифровки(расшифровки):

1. Ввод кодового слова
2. Подготовка алфавита, зашифрованного кодовым словом
3. Ввод сообщения для расшифровки
4. Сдвиг каждой буквы на три значения вправо в соответствии с зашифрованным алфавитом
5. Вывод расшифрованного сообщения

Далее я приступила к установке консольного приложения на свой компьютер. Консольное приложение или программа командной строки - это [компьютерная программа](https://translated.turbopages.org/proxy_u/en-ru.ru.92f4db91-64417b5c-e6d01e84-74722d776562/https/en.wikipedia.org/wiki/Computer_program), предназначенная для использования через [текстовый](https://translated.turbopages.org/proxy_u/en-ru.ru.92f4db91-64417b5c-e6d01e84-74722d776562/https/en.wikipedia.org/wiki/Text-only) [пользовательский интерфейс](https://translated.turbopages.org/proxy_u/en-ru.ru.92f4db91-64417b5c-e6d01e84-74722d776562/https/en.wikipedia.org/wiki/User_interface), такой как [текстовый терминал](https://translated.turbopages.org/proxy_u/en-ru.ru.92f4db91-64417b5c-e6d01e84-74722d776562/https/en.wikipedia.org/wiki/Text_terminal), [интерфейс командной строки](https://translated.turbopages.org/proxy_u/en-ru.ru.92f4db91-64417b5c-e6d01e84-74722d776562/https/en.wikipedia.org/wiki/Command-line_interface) некоторых [операционных систем](https://translated.turbopages.org/proxy_u/en-ru.ru.92f4db91-64417b5c-e6d01e84-74722d776562/https/en.wikipedia.org/wiki/Operating_system) или текстовый интерфейс. Для установки я выбрала одно из самый популярных консольных приложений —Microsoft Visual Studio.

Наконец, я приступила к самому написанию кода моей программы. Первым этапом стало написание так называемой “верхушки” программы. Здесь определяется функция с именем main, которая не принимает никаких аргументов (внутри круглых скобок ничего нет) и не выполняет никаких содержательных команд. В каждой программе на C++ должна быть ровно одна функция main — с неё начинается выполнение программы. У функции указан тип возвращаемого значения int (целое число), и она возвращает 0 — в данном случае это сообщение для операционной системы, что программа выполнилась успешно. И наоборот, ненулевой код возврата означает, что при выполнении возникла ошибка (например, программа получила некорректные входные данные). Также здесь пишутся все необходимые далее библиотеки и команда, отвечающая за использование области имен переменных std.

#include <windows.h> //*ввод необходимых библиотек*

#include <conio.h>

#include <iostream>

#include <array>

#include <iostream>

#include <cctype>

#include <algorithm>

#include <locale>

#include <Windows.h>

#include <vector>

using namespace std; // *команда, отвечающая за использование имен переменных std*

int main() // *функция с именем main*

{

Для дальнейшей работы с шифром на русском языке я включила в свой код функцию, позволяющую работать на русском языке:

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Далее я создала необходимые для цикла массивы, обозначающие русский алфавит в разных регистрах: заглавные и обычные буквы. Я присвоила массиву заглавные буквы алфавита в соответствии с таблицей ASCII. Однако буква ‘ё’ находится в таблице не в свойственном ей месте. Поэтому для присваивания элементу массива значения ‘Ё’ было создано отдельное условие.

Char alphabet[33]={'а','б','в','г','д','е','ё','ж','з','и','й','к','л','м','н','о','п','р','с','т','у','ф','х','ц','ч','ш','щ','ъ','ы','ь','э','ю','я'}; //*создание массива с алфавитом русского языка*

char BBB[33] = {}; *//создание пустого массива для заглавных букв русского языка*

for (int v = 0; v < 33; v++) //*начало цикла по присваиванию элементам массива BBB значений элементов массива alphabet увеличенного регистра*

{

if (alphabet[v] != 'ё') //*действие при элементе не равном ‘ё’*

{

BBB[v] = alphabet[v] - 32; //*присваивание значения согласно таблице ASCII*

}

if (alphabet[v] == 'ё') //*действие при элементе равном ‘ё’*

{

BBB[v] = alphabet[v] - 16; // *присваивание значения согласно таблице ASCII*

}

}

После этого я создала массив для кодового слова, ввела его с клавиатуры и разделив по элементам с помощью данных операций ввела в необходимый массив:

char A[33] = { }; //*массив, содержащий кодовое слово*

int k = 0; //*создание локальных переменных*

int z = 0;

cout << "Введите ключевое слово "<<endl; //*вывод на экран содержания ковычек*

cin.getline(A, 34); // *ввод данных из консольного окна в массив для кодового слова*

for (int i = 0; i < 33; i++) //*цикл по элементам массива A*

{

z = k;

for (int j = 0; j < 33; j++) *//цикл по элементам массива alphabet*

{

if (A[i] == alphabet[j]) *//если элементы равны*

{k++;}

}

if (z == k)

{break; } *// досрочного выхода из цикла*

}

Следующим этапом послужило кодирование алфавита с помощью кодового слова. Для этого я присвоила элементам массива алфавита, повторяющимся с элементами кодового слова значение ‘0’ . Далее нули были удален, и я получила алфавит, закодированный кодовым словом.

char alph1[33] = {}; *//массив для алфавита с кодовым словом*

char alphabet1[33] = { }; //*алфавит с ‘0’*

for (int i = 0; i < 33; i++)

{

alphabet1[i] = alphabet[i]; //*присваивание элементам массива alphabet1 соответственных значений alphabet*

}

for (int i = 0; i < 33; i++)

{

for (int j = 0; j < 33; j++)

{

if (A[i] == alphabet1[j])

{alphabet1[j] = '0';} //*присваивание повторяющимся элементам значения ‘0’*

}

}

for (int i = 0; i < k; i++)

{alph1[i] = A[i];}// *присваивание первым элементам массива значений кодового слова*

int b = 0; //создание локальной переменной

for (int i = k; i < 33; i++)

{

if (alphabet1[b] != '0')

{alph1[i] = alphabet1[b];} //*присваивание следующим за кодовым словом элементам значения алфавита без повторов элементов*

if (alphabet1[b] == '0'*)// если значение алфавита с нулями равно ‘0’, то элемент не записывается в массив, а его порядковому номеру в массиве alph1 присваивается следующий*

{i--;}

b++;

}

Как я писала ранее, вторым этапом моего метода шифрования является сдвиг элементов на три значения влево в соответствии с заготовленным алфавитом. Поэтому далее я приступила к созданию конечного алфавита с сдвигом

char B[33] = { }; //*создание массива, имеющего значение alph1 со сдвигом*

int m = 3; //*значение сдвига*

for (int i = 0; i < 33; i++)

{

if (i + m < 33) *//действие для элементов меньше 33*

{

B[i] = alph1[i + m];

}

if (i + m >= 33) *//действие для элементов больших или равных 33*

{

B[i] = alph1[i + m - 33];

}

}

В сообщении для шифровки могут присутствовать не только строчные буквы, но и заглавные. Поэтому я создала новый массив, элементы которого равны заглавным значениям массива B. Из-за особенностей таблицы ASCII пришлось сделать отдельное действие для присваивания буквы ‘Ё’.

char BB[33] = { }; //создания массива заглавных букв массива B

for (int v = 0; v < 33; v++)

{

if (B[v] != 'ё')

{

BB[v] = B[v] - 32;

}

if (B[v] == 'ё') //*действие для присваивания ‘Ё’*

{

BB[v] = B[v] - 16;

}

}

Далее моя программа делится на “кодировку” и “декодировку”. Необходимо написать программу так, чтобы при нажатии клавиши ‘1’ производилась кодировка, а при нажатии ’2’ — декодировка. Мною было принято решение осуществлять задумку через сочетание while и do:

char XCAPE[1] = {'3'};

cout << "Введите номер операции:" << endl; //*вывод на экран предлагаемых опций*

cout << "1 - Кодировка" << endl;

cout << "2 - Декодировка" << endl;

do

{

XCAPE[0] = \_getch(); *//считывание символа без повторения*

if ((XCAPE[0] != '1') && (XCAPE[0] != '2'))

{XCAPE[0]='3'; }

}

while (XCAPE[0] == '3'); //*цикл не заканчивается пока значение не будет равно 1 или 2*

При выборе ‘1’ (кодировки) производитcя следующая операция:

if (XCAPE[0] == '1')

{

cout <<endl<< "Вы выбрали операцию кодировка" << endl;

char sh[1000] = { }; //*создание массива для вводимого сообщения*

cout << "Введите сообщение для кодировки: " << endl;

cin.getline(sh, 1001);

int help = 0; //*создание вспомогательной переменной*

cout << "Вот готовое сообщение: " << endl;

for (int i = 0; i < 1000; i++) //*цикл для введенного сообщения*

{

help = 0;

for (int j = 0; j < 33; j++)

{

if (sh[i] == alphabet[j]) //*если элемент sh равен алфавиту alphabet*

{

cout << B[j]; //*вывод элемента зашифрованного массива в соответствии с порядковым номером*

help = 1;

}

if (sh[i] == BBB[j]) //*если элемент sh равен алфавиту BBB*

{

cout << BB[j]; //*вывод элемента зашифрованного массива в соответствии с порядковым номером*

help = 1;

}

}

if (help == 0) //*если вспомогательная переменная равна нулю(то есть элемент массива не равен ни одному из алфавитов)*

{cout << sh[i];} //*вывести элемент без изменений*

}

}

При выборе ‘2’ (декодировки) производитcя следующая операция:

if (XCAPE[0] == '2')

{

cout << endl<<"Вы выбрали операцию декодировка" << endl;

//\_getch();

char ch[1000] = { }; //создание массива для декодировки сообщения

cout << "Введите сообщение для декодировки: " << endl;

cin.getline(ch, 1001);

int help = 0; ; //*создание вспомогательной переменной*

cout << "Вот готовое сообщение: " << endl;

for (int i = 0; i < 1000; i++) //*цикл для введенного сообщения*

{

help = 0;

for (int j = 0; j < 33; j++)

{

if (ch[i] == B[j]) //*если элемент ch равен алфавиту B*

{

cout << alphabet[j]; ]; //*вывод элемента расшифрованного массива в соответствии с порядковым номером*

help = 1;

}

if (ch[i] == BB[j]) //*если элемент ch равен алфавиту BB*

{

cout << BBB[j]; //*вывод элемента расшифрованного массива в соответствии с порядковым номером*

help = 1;

}

}

if (help == 0) //*если вспомогательная переменная равна нулю(то есть элемент массива не равен ни одному из алфавитов)*

{

cout << ch[i]; //*вывести элемент без изменений*

}

}

}

Таким образом, я написала программу, производящую кодировку и декодировку по созданному мной методу шифрования. Последним этапом практической части моего проекта является создание презентации — памятки для пользования и ознакомления с моей программой. Предлагаю вам ознакомится с текстом моей программы в полном объеме в приложении 1.

# 3.Заключение

В наше время вопрос безопасности данных в интернете стоит особенно остро. Многие люди сталкиваются с проблемой утечки данных и их дальнейшим использованием в нежелательных целях мошенниками. Столкнувшись с данной проблемой, я нашла и реализовала ее решение—создание метода шифрования.

Я смогла создать метод шифрования, позволяющий составить сообщение, которое не будет понятным посторонним людям, достигла своей цели: разработать такой метод шифрования, который не будут понимать посторонние людям.

Изучив самые известные виды шифрования, я решила, что буду создавать метод шифрования, комбинирующий два вида шифрования: шифр Цезаря и шифр с использованием кодового слова.

Также решила, что буду использовать уже известный мне язык программирования С++. Он имеет множество плюсов, среди которых высокая скорость и большое количество библиотек, а значит большое количество путей воплощения моей идеи. Также свою роль сыграло то, что я уже пробовала программировать на этом языке, а поэтому мне будет легче работать на нем.

С использованием выбранных методов шифрования на языке программирования C++ я реализовала свой собственный метод шифрования. В заключение осуществила сначала внутреннее, а потом и внешнее тестирование, передав зашифрованное сообщение адресату(подруге).

Теперь, используя свой метод шифрования на постоянной основе, я не переживаю о том, что кто-то узнает содержание моей личной переписки. И могу сказать, что мои данные точно в безопасности.

С уверенностью могу сказать, что я выполнила все поставленные задачи в полном объеме и в срок.

# Список литературы

1. Полиалфавитный шифр Википедия URL : https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B8%D0%B0%D0%BB%D1%84%D0%B0%D0%B2%D0%B8%D1%82%D0%BD%D1%8B%D0%B9\_%D1%88%D0%B8%D1%84%D1%80 (дата обращения: 19.02.2022)
2. С++ для начинающих URL : <https://www.cyberforum.ru/cpp-beginners>(дата обращения: 19.02.2022)
3. Таблица символов ASCII + Windows 1251 URL: <https://snipp.ru/handbk/table-ascii>(дата обращения: 19.02.2022)
4. Java URL: https://ru.wikipedia.org/wiki/Java(дата обращения: 19.02.2022)
5. Python URL: https://ru.wikipedia.org/wiki/Python(дата обращения: 19.02.2022)
6. C++ URL: https://ru.wikipedia.org/wiki/C%2B%2B (дата обращения: 19.02.2022)
7. Выбираем язык программирования: что нужно знать о С++ URL: https://skillbox.ru/media/code/vybiraem\_yazyk\_programmirovaniya\_chto\_nuzhno\_znat\_o\_s\_/(дата обращения: 19.02.2022)
8. Страуструп Б. Программирование: принципы и практика использования C++, испр. изд. : Пер. с англ. — М. : ООО “И.Д. Вильямс”, 2011. — 1248 с.
9. Страуструп Б. Дизайн и эволюция С++: Пер. с англ. – М.: ДМК Пресс. 2018– 448с .

# Приложение 1

#include <windows.h> //*ввод необходимых библиотек*

#include <conio.h>

#include <iostream>

#include <array>

#include <iostream>

#include <cctype>

#include <algorithm>

#include <locale>

#include <Windows.h>

#include <vector>

using namespace std; // *команда, отвечающая за использование имен переменных std*

int main() // *функция с именем main*

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Char alphabet[33]={'а','б','в','г','д','е','ё','ж','з','и','й','к','л','м','н','о','п','р','с','т','у','ф','х','ц','ч','ш','щ','ъ','ы','ь','э','ю','я'}; //*создание массива с алфавитом русского языка*

char BBB[33] = {}; *//создание пустого массива для заглавных букв русского языка*

for (int v = 0; v < 33; v++) //*начало цикла по присваиванию элементам массива BBB значений элементов массива alphabet увеличенного регистра*

{

if (alphabet[v] != 'ё') //*действие при элементе не равном ‘ё’*

{

BBB[v] = alphabet[v] - 32; //*присваивание значения согласно таблице ASCII*

}

if (alphabet[v] == 'ё') //*действие при элементе равном ‘ё’*

{

BBB[v] = alphabet[v] - 16; // *присваивание значения согласно таблице ASCII*

}

}

char A[33] = { }; //*массив, содержащий кодовое слово*

int k = 0; //*создание локальных переменных*

int z = 0;

cout << "Введите ключевое слово "<<endl; //*вывод на экран содержания ковычек*

cin.getline(A, 34); // *ввод данных из консольного окна в массив для кодового слова*

for (int i = 0; i < 33; i++) //*цикл по элементам массива A*

{

z = k;

for (int j = 0; j < 33; j++) *//цикл по элементам массива alphabet*

{

if (A[i] == alphabet[j]) *//если элементы равны*

{k++;}

}

if (z == k)

{break; } *// досрочного выхода из цикла*

}

char alph1[33] = {}; *//массив для алфавита с кодовым словом*

char alphabet1[33] = { }; //*алфавит с ‘0’*

for (int i = 0; i < 33; i++)

{

alphabet1[i] = alphabet[i]; //*присваивание элементам массива alphabet1 соответственных значений alphabet*

}

for (int i = 0; i < 33; i++)

{

for (int j = 0; j < 33; j++)

{

if (A[i] == alphabet1[j])

{alphabet1[j] = '0';} //*присваивание повторяющимся элементам значения ‘0’*

}

}

for (int i = 0; i < k; i++)

{alph1[i] = A[i];}// *присваивание первым элементам массива значений кодового слова*

int b = 0; //создание локальной переменной

for (int i = k; i < 33; i++)

{

if (alphabet1[b] != '0')

{alph1[i] = alphabet1[b];} //*присваивание следующим за кодовым словом элементам значения алфавита без повторов элементов*

if (alphabet1[b] == '0'*)// если значение алфавита с нулями равно ‘0’, то элемент не записывается в массив, а его порядковому номеру в массиве alph1 присваивается следующий*

{i--;}

b++;

}

char B[33] = { }; //*создание массива, имеющего значение alph1 со сдвигом*

int m = 3; //*значение сдвига*

for (int i = 0; i < 33; i++)

{

if (i + m < 33) *//действие для элементов меньше 33*

{

B[i] = alph1[i + m];

}

if (i + m >= 33) *//действие для элементов больших или равных 33*

{

B[i] = alph1[i + m - 33];

}

}

char BB[33] = { }; //создания массива заглавных букв массива B

for (int v = 0; v < 33; v++)

{

if (B[v] != 'ё')

{

BB[v] = B[v] - 32;

}

if (B[v] == 'ё') //*действие для присваивания ‘Ё’*

{

BB[v] = B[v] - 16;

}

}

char XCAPE[1] = {'3'};

cout << "Введите номер операции:" << endl; //*вывод на экран предлагаемых опций*

cout << "1 - Кодировка" << endl;

cout << "2 - Декодировка" << endl;

do

{

XCAPE[0] = \_getch(); *//считывание символа без повторения*

if ((XCAPE[0] != '1') && (XCAPE[0] != '2'))

{XCAPE[0]='3'; }

}

while (XCAPE[0] == '3'); //*цикл не заканчивается пока значение не будет равно 1 или 2*

if (XCAPE[0] == '1')

{

cout <<endl<< "Вы выбрали операцию кодировка" << endl;

char sh[1000] = { }; //*создание массива для вводимого сообщения*

cout << "Введите сообщение для кодировки: " << endl;

cin.getline(sh, 1001);

int help = 0; //*создание вспомогательной переменной*

cout << "Вот готовое сообщение: " << endl;

for (int i = 0; i < 1000; i++) //*цикл для введенного сообщения*

{

help = 0;

for (int j = 0; j < 33; j++)

{

if (sh[i] == alphabet[j]) //*если элемент sh равен алфавиту alphabet*

{

cout << B[j]; //*вывод элемента зашифрованного массива в соответствии с порядковым номером*

help = 1;

}

if (sh[i] == BBB[j]) //*если элемент sh равен алфавиту BBB*

{

cout << BB[j]; //*вывод элемента зашифрованного массива в соответствии с порядковым номером*

help = 1;

}

}

if (help == 0) //*если вспомогательная переменная равна нулю(то есть элемент массива не равен ни одному из алфавитов)*

{cout << sh[i];} //*вывести элемент без изменений*

}

}

if (XCAPE[0] == '2')

{

cout << endl<<"Вы выбрали операцию декодировка" << endl;

//\_getch();

char ch[1000] = { }; //создание массива для декодировки сообщения

cout << "Введите сообщение для декодировки: " << endl;

cin.getline(ch, 1001);

int help = 0; ; //*создание вспомогательной переменной*

cout << "Вот готовое сообщение: " << endl;

for (int i = 0; i < 1000; i++) //*цикл для введенного сообщения*

{

help = 0;

for (int j = 0; j < 33; j++)

{

if (ch[i] == B[j]) //*если элемент ch равен алфавиту B*

{

cout << alphabet[j]; ]; //*вывод элемента расшифрованного массива в соответствии с порядковым номером*

help = 1;

}

if (ch[i] == BB[j]) //*если элемент ch равен алфавиту BB*

{

cout << BBB[j]; //*вывод элемента расшифрованного массива в соответствии с порядковым номером*

help = 1;

}

}

if (help == 0) //*если вспомогательная переменная равна нулю(то есть элемент массива не равен ни одному из алфавитов)*

{

cout << ch[i]; //*вывести элемент без изменений*

}

}

}