Міністерство освіти і науки України
Національний університет «Одеська політехніка»
Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
Кафедра кібербезпеки та програмного забезпечення

Кошель Єлизавета Володимирівна,
студент групи РУ-181

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Розробка додатка для шифрування текстової інформації з метою забезпечення безпеки каналу передачі даних

Спеціальність:
125 Кібербезпека

Спеціалізація, освітня програма:
Управління кібербезпекою

Керівник:
Бобок Іван Ігорович,
д.т.н., доцент

Одеса – 2022

Міністерство освіти і науки України
Національний університет «Одеська Політехніка»
Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
Кафедра кібербезпеки та програмного забезпечення
Рівень вищої освіти перший (бакалаврський)
Спеціальність 125 – Кібербезпека
Освітня програма – Управління кібербезпекою

ЗАТВЕРДЖУЮ
Завідувач кафедри КБПЗ
_____
д.т.н.,проф. А.А.Кобозєва
____ _____2022р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Кошель Єлизаветі Володимирівні

1.Тема роботи: *Розробка додатка для шифрування текстової інформації з метою забезпечення безпеки каналу передачі даних,*
керівник роботи *Бобок Іван Ігорович, д. т. н., доцент,*
затверджені наказом ректора від „17_”_05. 2022 р. №168-в .
2.Зміст роботи*: аналіз проблемної області, постановка задачі, аналіз загроз інформаційної безпеки та методів захисту інформації, дослідження відомих методів шифрування, створення додатку для шифрування текстової інформації різними методами.*
3. Перелік ілюстративного матеріалу: *приклади використання середовища Sharp Developer та виконання програми.*

4. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
Охорона праці	Ярова І.В., доцент	Завдання видав	Завдання прийняв

5. Дата видачі завдання “_____”_____2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання	Примітка
1	*Аналіз джерел з теми випускної кваліфікаційної роботи*	*15.02.2022*	*виконано*
2	*Обґрунтування вибору рішення. Збір даних*	*25-03-2022*	*виконано*
3	*Аналіз основних загроз інформаційної безпеки та методи їх протидії*	*11-03-2022*	*виконано*
4	*Програмна реалізація завдання*	*30-03-2022*	*виконано*
5	*Підготовка тексту роботи*	*25-05-2022*	*виконано*
6	*Підготовка презентації та доповіді*	*30-05-2022*	*виконано*
7	*Попередній захист*	*03-06-2022*	*виконано*
8	*Нормоконтроль, рецензування*	*17.06.2022*	*виконано*

Здобувач вищої освіти _____ *Кошель Є. В.*

Керівник роботи _____ *Бобок І. І.*

ЗАВДАННЯ

на розробку розділу «Охорона праці» у кваліфікаційній роботі бакалавра

Кошель Єлизаветі Володимирівні

Інститут Інформаційної безпеки, радіоелектроніки та телекомунікацій

Кафедра Кібербезпеки та програмного забезпечення

Дата отримання завдання 02.06.2022
Консультації 02.05.2022, 14.06.2022
Дата закінчення розділу 14.06.2022

Тема роботи *Розробка додатка для шифрування текстової інформації з метою забезпечення безпеки каналу передачі даних*

Зміст розділу

1. Аналіз умов праці і вибір основних заходів виробничої безпеки
2. Аналіз пожежної безпеки і вибір заходів і засобів пожежної безпеки

Керівник дипломної роботи

_____ Бобок І..
(підпис)

« __ » _____ 2022 р.

Консультант з охорони праці

_____ Ярова І.А.
(підпис)

« __ » _____2022 р.

АНОТАЦІЯ

Кваліфікаційна робота на тему “ Розробка додатка для шифрування текстової інформації з метою забезпечення безпеки каналу передачі даних” на здобуття першого (бакалаврського) рівня вищої освіти за спеціальністю 125 – Кібербезпека, спеціалізація, освітня програма: Управління кібербезпекою, містить 8 рисунків, 2 таблиці, 1 додаток, 25 літературних джерел за переліком посилань. Робота виконана на 60 сторінках загального тексту і 50 сторінках основного тексту.

Метою роботи є захищення особистої інформації, яка зберігається або передається в текстовому вигляді, шляхом створення додатку для шифрування текстових даних.

Об’єктом дослідження є захист інформації. Предметом дослідження є методи шифрування текстової інформації за допомогою різних методів шифрування.

Аналізуючи закони України «Про інформацію», «Про доступ до публічної інформації», «Про технічний захист інформації» були зроблені висновки про властивості інформації, її цінність і необхідність її захисту.

У результаті виконання кваліфікаційної роботи був реалізований додаток для шифрування даних різними методами шифрування.

Зроблено висновок, що в сучасному світі значимість інформації набуває особливого значення. Від нього залежить соціальний добробут власника інформації.

Результати даної роботи можуть бути використані для приховання інформації, яка зберігається, або підвищення рівня безпеки каналу передачі даних.

ІНФОРМАЦІЯ, ІНФОРМАЦІЙНА БЕЗПЕКА, ШИФРУВАННЯ, ЗАГРОЗИ, МЕТОДИ ЗАХИСТУ ІНФОРМАЦІЇ.

ANNOTATION

Qualification work on "Development of the application for encryption of textual information to ensure the security of the data channel" to obtain the first (bachelor's) level of higher education in the specialty 125 – Cybersecurity, specialization, educational program: Cybersecurity Management, contains 8 figures, 2 tables, 1 appendix, 17 references. The work is performed on 58 pages of general text and 44 pages of main text.

The purpose of the work is protection personal information that is stored or transmitted in text form by creating an application for encrypting text data.

The object of research is information protection. The subject of research is the methods of encrypting textual information using various encryption methods.

By analyzing the laws of Ukraine "On Information", "On Access to Public Information", "On Technical Protection of Information", were made conclusions about the properties of information, its value and necessity for its protection.

As a result of the qualification work, an application was implemented to encrypt data with different encryption methods.

It is concluded that in the modern world the value of information becomes especially important. The economic and psychological status of the owner of the information depends on it.

The results of this work can be used to hide the information stored, or increase the security of the data channel.

INFORMATION, INFORMATION SECURITY, ENCRYPTION, THREATS, INFORMATION PROTECTION METHODS.

ЗМІСТ

ВСТУП

Дана дипломна робота присвячена темі захисту інформації. Інформаційний простір в 21 столітті займає майже всі сфери життєдіяльності людини. Важко зазначити, де не створюється, переміщається чи використовується інформація. Застосування сучасних інформаційних технологій має вплив на рівень інформаційної безпеки як і людини, так і суспільства в цілому. Інформаційно-комунікаційне насичення суспільних відносин, одночасно створює можливість настання бажаних і загрозливих наслідків як для суспільства в цілому, так і для кожного громадянина. Людина, котра володіє цінними знаннями, завжди має певну перевагу або прибуток, також інформація є предметом купівлі-продажу, а в такому випадку можна стверджувати, що вона є цінною. Цінність інформації є суб'єктивною: для користувача мережі його особиста інформація, тексти особистих повідомлень є також важливими.

Людина, яка використовує будь-який цифровий пристрій (ноутбук, планшет, смартфон), навіть не помічає кількості текстової інформації на гаджеті. Вся ця інформація підлягає загрозі порушення конфіденційності.

Часто причиною «витоку» інформації є людський фактор. Наприклад, залишений ПК з відкритим месенджером чи робочим столом, на якому розміщені важливі текстові документи.

Відповідно, якщо текстові документи та повідомлення підлягали шифруванню – це б захистило інформацію власника. Тому ми робимо висновок щодо актуальності створення додатку для шифрування текстових повідомлень. Використання такого додатку буде зручним та зрозумілим завдяки графічному інтерфейсу, а можливість вибору того чи іншого методу шифрування допоможе підвищити рівень інформаційної безпеки.

Однією з причин вибору шифрування в якості методу захисту інформаціє є простота реалізації і використання, метод не потребує особливого обладнання чи фінансування. Також даний процес не займає багато часу і є достатньо стійким для використання в побуті.

Для досягнення мети були поставлені такі задачі:

- дослідження існуючих методів шифрування;
- розробка та програмна реалізація додатка, який виконує шифрування за різними методами;
- створення комбінованого методу шифрування.

Для виконання завдання було використані такі шифри: переміщення, Віженера, Вернама та Ель-Гамаля.

До особливостей додатка належать: по-різному написаний алфавіт для кожного алгоритму, асортимент шифрів, можливість вибору бажаного, використання комбінованого шифру.

Практична цінність результатів, представлених у роботі полягає у швидкому та зручному захисту цінної інформації будь-якою особою, яка має програму.

1 ІНФОРМАЦІЙНЕ СЕРЕДОВИЩЕ ТА МЕТОДИ ЙОГО ЗАХИСТУ

1.1 Поняття інформації. Цінність інформації

З того часу, коли був винайдений комп'ютер, люди почали використовувати термін «дані» з метою позначення комп'ютерної інформації, яка була передана або збережена. Терміни «дані» та «інформація» часто використовуються як взаємозамінні, але насправді вони не є однаковими. Між цими компонентами та їх призначенням є незначні відмінності.

Дані самі по собі не мають сенсу чи мети, тобто - це необроблена форма знань. Це сукупність деталей або даних, що зберігаються у вигляді зображень, символів, описів або простих спостережень за сутностями, подіями чи речами, з яких можна зробити аналіз і висновки. Вони необроблені, і їх потрібно оброблювати, щоб отримати важливу інформацію.

Інформація вже давно є універсальним поняттям, але термін досі не має загального наукового визначення. З точки зору різних галузей науки, воно визначається різними специфічними характеристиками.

Сьогодні під інформацією розуміють сукупність корисних відомостей, які збираються, реєструються, зберігаються, передаються та перетворюються. Інформація – це ресурс, який можна накопичувати, продавати, оновлювати. Він придатний для колективного використання і не втрачає своєї якості в процесі споживання на відмінну від інших ресурсів.

Інформація – це дані, які можна порівняти, щоб отримати значущі висновки, як того вимагає контекст. Інформація підлягає структуруванню, обробці та подачі із заданим значенням. Це підвищує надійність даних.

Відповідно до статті 1 Закону України «Про інформацію» інформація - будь-які відомості та/або дані, які можуть бути збережені на матеріальних носіях або відображені в електронному вигляді [1] .

Інформація має такі властивості:

- актуальність - важливість в певний момент часу.

- вірогідність - інформація повинна відображати реальність.

- об'єктивність - неупередженість, інформація не залежить від волі і бажання людини.

- повнота - інформації повинно бути достатньо для розуміння та прийняття рішень.

- адекватність - певний рівень ймовірності створюваного образу реального об'єкта, явища чи процесу.

- інформація має бути цінною, що вимірюється користю, яку інформація може принести для досягнення цілей.

- конфіденційність - властивість захисту інформації від несанкціонованого доступу;

- інформація має бути доступною. Доступність залежить від здатності одержувача сприймати інформацію.

Також інформації притаманні такі ознаки, як: корисність, вірогідність, однозначність, періодичність, несуперечливість, надмірність. Корисність інформації зазвичай оцінюється за впливом інформації на результати управління.

На практиці інформація може виявитися абсолютно непотрібною (не змінювати ймовірність досягнення мети) і дуже цінною, підвищуючи ймовірність реалізації завдань.

В деяких випадках отримана інформація може призвести до прийняття рішень, які знижують ймовірність досягнення мети – такий тип інформації називається дезінформацією.

Важливою якісною характеристикою інформації є її вірогідність.. Важливою вважається та інформація, яка не перевищує допустимого рівня спотворення фактичного явища чи процесу та відображає те, що вона повинна відображати.

Надмірність (дублювання) може бути корисною, якщо вона підвищує надійність системи даних, і непотрібною, якщо вона містить повторювані дані, що не використовуються для прийняття рішень [2].

Завдання будь-якого менеджера - забезпечити достовірну цінність переданої інформації. Критеріями оцінки цієї вимоги є глибина, зміст і ступінь релевантності інформації для цілі.

Своєчасність інформації важлива для ефективної роботи системи управління. Часткова інформація, отримана вчасно, більш корисна для керівництва, ніж повна, якщо вона отримана із запізненням.

Відповідно до статті 11 Закону України «Про інформацію» конфіденційна інформація про фізичних осіб не може збиратися, зберігатися, використовуватися та поширюватися без згоди особи, крім випадків, передбачених законом, та лише в інтересах національної безпеки, економічної безпеки, процвітання та прав людини[1].

Інформаційні технології інтенсивно застосовуються в різних сферах людської діяльності, особливо важливим є захист інформації в розподілених або локальних інформаційно-обчислювальних системах. Водночас є й протиріччя: з одного боку, автоматизація процесів значно покращує можливості управління, з іншого — призводить до порушення таких характеристик, як доступність, конфіденційність, достовірність та цілісність.

Оскільки інформацію можна продати, купити, підробити, вкрасти тощо, її потрібно якимось чином оцінювати. Інформація, якою одна особа через машину обмінюється з іншою людиною або машиною, може бути важливою і, отже, захищеною, тобто, є цінною.

Цінною можна вважати ту інформацію, володіння якою дасть дійсному чи потенційному власнику певну користь: моральну, матеріальну, політичну тощо. Оскільки в суспільстві завжди є люди, які хочуть мати певну перевагу над іншими, у них може виникнути бажання отримати цінну інформацію нелегальним шляхом, а їх власники мають потребу захистити її.

Цінність – атрибут інформаційного відношення суб’єкта, елементами якого є інформація та суб’єкт, що переслідує свої цілі. З точки зору виникнення інформації на життєвому рівні, більш правильним є визначення, що інформація є не атрибутом автономних систем, а їх взаємозв’язок з об’єктами.

До того ж, цінність не є природною властивістю інформації, а скоріше результатом суб’єктно-практичних взаємодій між об’єктами (інформацією) та суб’єктами (користувачами). Будь-яка цінність обумовлена практикою, що

розуміється в широкому сенсі, і практика є об’єктивною детермінантою цінності. Цінність – це те, що потрібно людині для виконання практичної та пізнавальної діяльності, а практика сприяє об’єктивності оцінювання.

1.2 Розповсюдження інформації. Загрози інформації

Під розповсюдженням інформації розуміється поширення, оприлюднення, продаж письмової або публічно оприлюдненої інформації в порядку, встановленому законом.

За порядком доступу інформацію класифікують на відкриту інформацію та інформацію з обмеженим доступом. Вся інформація вважається загальнодоступною, за винятком тих, які згідно із законом мають обмежений доступ. Інформація з обмеженим доступом – це конфіденційна, секретна та конфіденційна інформація.

Конфіденційна інформація – це інформація про фізичних осіб та інформація, доступ до якої обмежено фізичними або юридичними особами, крім суб’єктів владних повноважень. Конфіденційна інформація може бути розповсюджена на вимогу сторони, на визначених стороною умовах, у порядку, визначеному стороною, та за інших обставин, передбачених законом.

Розпорядники інформації, які володіють конфіденційною інформацією, можуть поширювати цю інформацію лише за наявності згоди осіб, які обмежили доступ до інформації. В іншому випадку - лише в інтересах національної безпеки, економічного процвітання та прав людини.

Таємна інформація - інформація, доступ до якої обмежено. Розголошення таємної інформації може завдати шкоди окремим особам, суспільству та нації. Відомості, що містять державну таємницю, професійну таємницю, банківську таємницю, розвідувальну таємницю, таємницю досудового розслідування та іншу таємницю, встановлену законом, вважаються таємною[3].

Загрози інформаційної безпеки:

Загрози конфіденційності (незаконний доступ до інформації). Загроза

полягає в тому, що інформація буде відома тим, хто не має до неї доступу. Це відбувається, коли отримали доступ до інформації з обмеженим доступом, яка зберігається в комп'ютерній системі або передається з однієї системи в іншу.

Термін «витік» використовується через загрозу вторгнення в приватне життя. Джерелами таких загроз можуть бути «людський фактор» (наприклад, випадкове делегування повноважень іншому користувачеві), збої програмного та апаратного забезпечення.

Загрози цілісності (несанкціоновані зміни даних). Це загрози, пов'язані з можливістю модифікації інформації, що зберігається в інформаційній системі. Порушення цілісності можуть виникати з різних причин - від навмисних дій співробітників до збою програмного забезпечення.

Загрози доступності – це дії, які унеможливлюють або ускладнюють доступ до ресурсів інформаційної системи. Це створення умов, коли доступ до послуг чи інформації або блокується, або певні дії можуть не забезпечуватися протягом певного періоду часу[4].

Інформація, що поширюється без використання засобів масової інформації, повинна містити достовірну інформацію про її власника або іншу особу, яка поширює інформацію, у формі та в кількості, достатніх для ідентифікації цієї особи.

Забороняється розповсюдження інформації з метою розпалювання війни, розпалювання національної, расової чи релігійної ворожнечі та ворожнечі, а також іншої інформації, поширення за яку тягне за собою кримінальну чи адміністративну відповідальність[3].

Нижче наведено кілька прикладів інформаційних загроз в Інтернеті.

Атака листами вважається найстарішим методом атаки, незважаючи на його простоту та примітивну природу: наплив листів унеможливлює використання поштових скриньок, а іноді навіть цілих поштових серверів. Цієї атаки важко запобігти, оскільки постачальники можуть обмежити кількість листів від одного відправника, але адреса та тема відправника часто генеруються рандомно.

Мережева розвідка, під час якої хакер фактично не здійснює жодних

деструктивних дій, але таким чином може отримати конфіденційну інформацію про побудову та принципи роботи комп'ютерної системи жертви. Під час цього інтелектуального процесу зловмисники можуть виконувати сканування портів, запити DNS, луна-тестування відкритих портів, доступність і безпеку проксі-сервера.

Сніффінг пакетів - досить поширений тип атаки, заснований на мережевій карті в безладному режимі і режимі моніторингу мережі Wi-Fi. У цьому режимі всі пакети, отримані мережевою картою, відправляються для обробки в спеціальну програму, яка називається сніфером. У результаті зловмисники можуть отримати доступ до великої кількості службової інформації: хто надсилав пакети, звідки, куди та за якими адресами ці пакети пройшли.

IP-спуфінг - тип атаки в погано захищених мережах, коли зловмисник видає себе за авторизованого користувача всередині або за межами організації. Ця атака можлива, якщо система безпеки дозволяє ідентифікувати користувачів лише за IP-адресою і не потребує додаткового підтвердження.

Man-in-the-Middle – атака під час якої зловмисник перехоплює канал зв'язку між двома системами, у результаті чого може отримати доступ до всієї переданої інформації. Метою цього типу атаки є крадіжка або підробка переданої інформації або отримання доступу до ресурсів мережі. Таким чином, технічно захистити себе можна, лише зашифрувавши передану інформацію.

Відмова в обслуговуванні – це атака, призначена для того, щоб змусити сервер не відповідати на запити[5].

1.3 Методи та засоби захисту інформації

Захист інформації – це діяльність, спрямована на забезпечення захисту конфіденційності, цілісності та доступності важливої для нації, суспільства та окремих осіб інформації.

Запобігання або істотне перешкоджання загрозам державної інформації, просування законних інтересів громадян, юридичних осіб, державних органів у

виконанні ними завдань і функцій, а також можливих загроз державі, соціальній діяльності. чи політичних діячів, економічні, моральні та інші втрати – мета захисту інформації.

Вирізняють основні засоби боротьби із загрозами безпеки комп’ютерної системи: праві, морально-етичні, організаційні, фізичні та технічні.

Засоби правового захисту включають законодавчо-правову базу, на якій гуртуються інші засоби правового захисту. В Україні, зокрема, це закони «Про інформацію», «Про технічний захист інформації», «Про державну таємницю», «Про публічну інформацію», Державна служба спецзв’язку, які регулюють правила обробки інформації.

Дотримання норм поведінки, які традиційно склалися або розвиваються в інформаційних суспільствах належать до морально-етичних засобів протистояння.

Організаційні засоби захисту представляють собою засоби організаційного характеру для регулювання обробки даних, використання її ресурсів, діяльності співробітників, взаємодії користувачів із системами для запобігання або ускладнення реалізації загроз інформаційної безпеки.

Фізичні засоби захисту базуються на застосуванні різноманітних механічних, електронних або електромеханічних пристроїв, спеціально призначених для створення фізичних бар'єрів для можливого проникнення потенційних зловмисників, їх доступу до компонентів системи та захищеної інформації, а також візуальних засобів спостереження, мови. та охоронна сигналізація.

Технічні засоби захисту ґрунтуються на застосуванні різноманітних електронних пристроїв та спеціального програмного забезпечення, що є частиною автоматизованої системи і виконують функцію захисту інформації окремо або в поєднанні з іншими засобами[5].

З метою ідентифікації користувачів, контролю доступу, шифрування інформації, видалення залишкової інформації, наприклад, тимчасових файлів, а також тестового контролю системи захисту використовують програмні засоби. Переваги програмного забезпечення - універсальність, гнучкість, надійність,

простота встановлення, можливість модифікації та розвитку. Недоліки - обмежені можливості мережі, використовує частину ресурсів файлових серверів і робочих станцій, дуже чутливі до випадкових або навмисних змін, можуть залежати від типу комп'ютера (їх апаратного забезпечення)[6].

Отже, інформація застосовується у всіх сферах діяльності. Цінна та конфіденційна інформація мають підлягати захисту. Захист відбувається на законодавчому рівні, тобто регулювання правил збереження, передачі та обробки даних в законах та на технічному рівні, використовуючи паролі, методи шифрування, тощо.

2 ШИФРУВАННЯ ДАНИХ

2.1 Поняття шифрування даних

Завдання захисту інформації в комп'ютерних системах сьогодні є одним із найважливіших у зв'язку з широким використанням локальних і глобальних комп'ютерних мереж, які передають величезні обсяги публічної, військової, комерційної та приватної інформації.

Галузь науки, яка вивчає методи шифрування та дешифрування інформації, називається криптологією. Вона поділяється на дві частини: криптографія та криптоаналіз.

Побудова сучасної криптографії як науки базується на низці базових знань і фактів з математики, фізики, теорії інформації та обчислювальної складності. Однак, незважаючи на притаманну їй складність, багато теоретичних досягнень у криптографії зараз широко використовується в нашому насиченому інформацією житті, як-от: пластикові смарт-картки, електронна пошта, банківські платіжні системи, Інтернет-комерція, системи електронних документів та ведення баз даних. Ця загальна внутрішня складність і практична застосовність до теоретичної науки можуть бути унікальними.

Криптографія — розділ прикладної математики, об'єктом вивчення якого є моделі, методи, алгоритми, програмне та апаратні засоби, що використовуються для перетворення інформації з метою приховування її вмісту, запобігання зміні чи несанкціонованому використанню. Іншими словами, криптографи намагаються знайти способи забезпечення конфіденційності та/або автентичності повідомлень.

Зворотні процеси називаються криптоаналітичними методами, а відповідна область дослідження – криптоаналізом. Отже, криптоаналіз — це наука, покликана подолати криптографічний захист.

Процес перетворення відкритих даних у зашифровані і навпаки називається шифрування даних. Перша частина процесу називається зашифруванням, а друга частина називається дешифруванням. Термін шифрування також використовується як синонім зашифрування. Проте некоректно використовувати термін «кодування»

як синонім шифрування (і «код», а не «шифр»), оскільки, як правило, кодування - представлення даних у належному наборі символів.

Ключ – набір символів, що використовується в криптосистемі, необхідний для безперебійного шифрування та дешифрування тексту. Секретність ключа забезпечує стійкість шифрування всієї системи.

Алгоритми шифрування — це математичні правила створення та зламу секретних паролів. Основною особливістю алгоритмів шифрування є криптостійкість.

Криптостійкість — це ознака шифру, яка визначає його стійкість до дешифрування без знання ключа. Як правило, вона визначається інтервалом часу, необхідним для розкриття шифру.

Криптоаналіз - набір методів і алгоритмів для розшифровки криптографічно захищених повідомлень, аналізу криптосистеми. Він поєднує в собі математичні методи порушення конфіденційності та автентичності, а також розшифровки повідомлення, не знаючи ключа.

Криптоаналізатор або криптоаналітик – це людина, яка займається дешифруванням. Криптоаналіз заснований на теорії ймовірностей і математичних науках, таких як: математична статистика, алгебра, теорія чисел і теорія алгоритмів.

Абсолютно стійким вважають шифр, який неможливо зламати. При використанні таких шифрів є необхідність в ключі для шифрування, розмір якого рівний або більший довжини вхідного повідомлення.

У більшості випадків шифрувальники вважають цей нюанс незручним, окрім одиничних випадків. Перевага надається шифрам, які не є абсолютно стійкими. У всіх випадках, за винятком деяких особливих, ця ціна непомірна, тому на практиці використовуються переважно менш стійкі шифри., найпоширеніші шифри можуть бути зламані за певний проміжок часу, у період якого буде зроблено певну кількість кроків. Кожен крок є певною маніпуляцією з числами.

Важливим є поняття фактичної стійкості, яке визначає реальну складність дешифрування. Одиницею виміру труднощів може бути кількість кроків,

основних математичних і логічних перетворень, які необхідні для дешифрування інформації, тобто визначити відповідний відкритий текст даного шифротексту з імовірністю не меншою за задане значення [7].

Велика кількість людей, які активно використовують новітні технології, не знають, що велика кількість даних підлягає захисту, так як використовується шифрування. Для прикладу, онлайн-покупки та онлайн-банкінг не можуть працювати без справного шифрування. Також шифрування використовується для захисту інтелектуальної власності та інновацій компанії та інших цінних даних.

2.2 Принципи створення криптографічних алгоритмів

Протягом довгого часу криптографія була більше ремеслом, аніж наукою. Люди, які створювали шифри спиралися на вдачу, вони не знали ніяких математичних теорій, на основі яких могли б реалізувати відповідні перетворення.

Роботою, яка змінила вектор розвитку криптографії, вважається теорія комунікації в секретних системах Клода Шеннона 1949 року. У даній теорії Шеннон представляє основні принципи створення криптографічних алгоритмів: розсіювання та змішування.

Поширення впливу одного символу вхідного повідомлення на кілька символів зашифрованого тексту називається розсіюванням. Підсумком цього принципу є поширення впливу ключового символу на кілька символів зашифрованого тексту, запобігаючи таким чином відновлення ключа по частинам. Перемішування — це використання криптографічних перетворень, що створюють перешкоди в відновленні узгодження символів відкритого тексту і шифротексту, ключа і шифротексту.

Криптографічний алгоритм повинен легко шифруватись і дешифруватися, при умові, що ключ відомий. Поширеним методом створення алгоритмів шифрування є виконання послідовності простих перетворень, які зазвичай використовують операції додавання, множення, заміни та перестановки. Перестановка змінює порядок символів відкритого тексту. У процесі заміни набір

символів відкритого тексту замінюється на таку ж кількість символів, а конкретний тип заміни визначається ключем.

Якщо міксувати використання підстановки і перестановки n-кількість разів, можна отримати стійкий шифр [8.9].

2.3 Види шифрування

На сьогоднішній день існує два методи шифрування: симетричний і асиметричний.

Симетричне шифрування використовує секретний ключ для шифрування та дешифрування, він єдиний для обох процесів. Алгоритми симетричного шифрування можна розділити на потокові та блочні. Алгоритми потокового шифрування поступово обробляють текстові повідомлення. Алгоритм блочного типу працює з порціями інформації фіксованого розміру. У більшості випадків розмір блоку становить 64 біта.

Завдяки особливій конструкції симетричне шифрування має деякі переваги, такі як висока пропускна здатність; невеликі ключі; ці шифри можуть бути використані як основа для побудови різноманітних криптографічних механізмів, включаючи генератори випадкових чисел, тощо.

Слід зазначити, що одним із недоліків цього шифрування є те, що в кожній великій мережі необхідно використовувати велику кількість ключів; при спілкуванні між кількома людьми ключі потрібно часто міняти; коли є зв’язок між двома людьми, ключ необхідно засекречувати на обох кінцях.

Прикладами симетричного шифрування є: Twofish, Serpent, AES (або Rainday), Blowfish, CAST5, RC4, TDES (3DES) і IDEA.

Введений у 1976 році, DES є найстарішим симетричним методом шифрування. Шифр розроблений IBM для захисту секретних державних даних і був офіційно прийнятий федеральними агентствами США в 1977 році. Алгоритм шифрування DES є одним з алгоритмів, включених до TLS (Transport Layer Security) версій 1.0 і 1.1.

DES перетворює блоки розміром 64-біт даних простого тексту в шифротекст, розбиваючи його на два окремих 32-розрядних блоки, шифруючи кожен блок окремо. Метод складаються з 16 різних циклів - таких як розширення, перестановка, заміна або інші операції - дані будуть проходити через ці цикли в зашифрованому вигляді. В результаті отримуємо 64-розрядні зашифровані блоки.

AES є одним із найпоширеніших методів шифрування. Він був розроблений як заміна DES і став новим стандартом шифрування після схвалення NIST у 2001 році. AES — це набір блочних шифрів з різною довжиною ключа та різними розмірами блоків.

Даний метод шифрування працює з методами підстановки та перестановки. Спершу вхідні дані перетворюються на блоки, які потім шифруються за допомогою ключа. У процесі шифрування над текстом виконуються різні процеси, такі як переміщення рядків, змішування стовпців і додавання ключів. Залежно від довжини ключа виконується певна кількість таких перетворювань. Важливо зазначити, що фінальний раунд, на відміну від попередніх, не включає процес мікшування..

Методи, які передбачають використання пари двох різних ключів, секретного та відкритого - асиметричне шифрування. Як випливає з назви, відкритий ключ вільно розміщується в мережі і аналогічно - секретний ключ завжди зберігається в секреті. При асиметричному шифруванні ключі працюють парами, коли інформація шифрується за допомогою відкритого ключа, для її дешифрування використовується лише відповідний секретний ключ, і навпаки. Ви не можете використовувати відкрити відкритиі закритий ключ з різних пар. Усі асиметричні пари ключів пов’язані математичними залежностями.

Важливо відзначити, що певні проблеми виникають, коли потрібно передати ключ для розшифрування інформації. Ключ може бути перехоплений хакерами в процесі передачі. Даний тип шифрування допомагає адресату контролювати цілісність інформації, що передається. Прикладами асиметричного шифрування є RCA і ESS.

Алгоритм асиметричного шифрування RSA був винайдений в 1977 році трьома вченими з Массачусетського технологічного інституту Роном Рівестом, Аді Шаміром і Леонардом Адлеманом. Сьогодні це найпоширеніший алгоритм асиметричного шифрування. Його ефективність полягає в підході «первинної факторизації». Основна ідея, два різних випадкових простих числа певного розміру вибираються і перемножуються, щоб створити ще одне велике число. Постає завдання: визначити вихідні прості числа гіганта множника. Завдання виявилося практично неможливим для сучасних комп'ютерів, не кажучи вже про людей.

Асиметричний алгоритм шифрування ECC був винайдений в 1985 році. Математики Ніл Кобліц і Віктор Міллер запропонували використання еліптичних кривих для створення шифру. В 2004-2005 роках почав використовуватися алгоритм ECC.

У процесі шифрування еліптична крива представляє собою набір точок, які задовольняють математичне рівняння $(y\ 2 = x\ 3 + ax + b)$.

RSA і ECC дотримуються принципу незворотності. Простіше кажучи, помножте число ECC, що представляє одну точку на кривій, на інше число, щоб отримати іншу точку на кривій. Щоб зламати шифр, потрібно знайти нову точку на кривій. Математика ECC побудована таким чином, що навіть якщо ви знаєте початкову точку, знайти нові точки майже неможливо [10].

Таким чином, будь-який алгоритм шифрування має свої переваги та недоліки. Тобто, симетричні методи шифрування чудово підходять для швидкого шифрування великої кількості інформації. Але даний алгоритм не забезпечує аутентифікацію, яка необхідна для безпеки в Інтернеті. З іншого боку, асиметричне шифрування забезпечує доступ до даних передбачуваного одержувача. Однак ця перевірка робить процес шифрування дуже повільним. Новітні криптосистеми використовують комбінацію симетричних і асиметричних алгоритмів, щоб скористатися перевагами обох схем.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ДОДАТКА ДЛЯ ШИФРУВАННЯ ТЕКСТОВОЇ ІНФОРМАЦІЇ

3.1 Обґрунтування вибору програмних засобів

Для реалізації практичного завдання використовувалося середовище Sharp Develop та мова програмування C#.

Sharp Develop представляє собою безкоштовне середовище програмування для проєктів на платформі Microsoft .NET. Середовище дозволяє програмувати на C#, VB.NET, Boo, Iron Python, Iron Ruby, F#. Це IDE з відкритим вихідним кодом, є можливість легко завантажити вихідний код і файли з певного джерела.

Нижче наведені особливості використання даного середовища та відмінності з Visual Studio.

Середовище розробки надає користувачам різноманітні можливості для підвищення продуктивності і багато в чому таке ж багатофункціональне, як Visual Studio .NET 2003, але не як Visual Studio 2005. До основних переваг SharpDevelop належить:

- компілятор C# від Microsoft і Mono;
- функціональність IntelliSense та розширення програмного коду;
- використання діалогового вікна Add Referenc для зв’язку із зовнішніми блоками збірки, включаючи блоки макета, встановлені в GAG ;
- наявність інструментів візуального дизайну Windows Forms;
- використання різних вікон, які називаються scouts, для перегляду структури проєкту та його складових:
- наявність утиліта Integrated Object Browser - Assembly Scout;
- використання утиліти для роботи з базами даних;
- утиліти для перетворення коду C# у VB .NET та зворотній поцес:
- інтеграція утиліти для тестування модулів .NET - NUnit і NAnt - для розмітки .NET:
- інтеграція з документацією SDK .NET Framework [11].

Sharp Develop сумісна з багатьма версіями проєктів VS, підтримує достатню кількість поширених мов програмування, зображених на рис. 3.1:

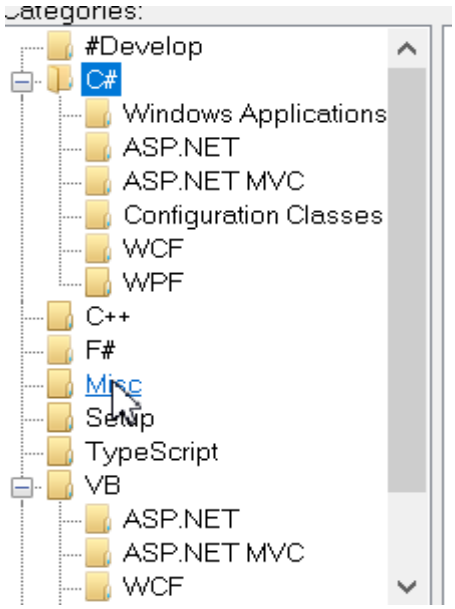


Рисунок 3.1 – Мови програмування Sharp Develop

Щоб використовувати SharpDevelop, потрібно створити нове об'єднання (Combain), куди буде доданий проєкт. Ви можете використовувати вікно Project Scout для перегляду файлів і посилань у цьому об'єднанні. У цьому вікні або у вікні Classes Scout ви можете відкривати файли у вікні коду, які за замовчуванням відображаються на вкладках. Редагування коду майже ідентично редагування Visual Studio, включаючи аналог автозавершення в SharpDevelop.

У тому випадку, якщо об'єднання використовується для реалізації графічної програми Windows, легко створити форму, де можна розмістити елементи керування аналогічно Visual Studio. Окремий виняток становлять особливості конструктори форм у новішому пакеті Visual Studio, і деякі з найновіших елементів керування Windows. Можливість встановлення властивостей елементу керування реалізована у вікні Properties Scout. Контрольний код міститься у файлі форми з розширенням .CS, як у Visual Studio.

Найбільш суттєву різницю ми можемо спостерігати при налагоджуванні помилкової логіки програми. На даний момент Sharp Develop не підтримує такого потужного налаштовувача, як Visual Studio.

Робота над Sharp Develop триває, пакет має багато чудових функцій, незважаючи на деякі недоліки та дуже слабку документацію. Наприклад, у документації немає інформації про використання команди Debugger меню Tools в Sharp Develop.

Однією з сучасних, об’єктно-орієнтованих мов програмування є C#, яка дає можливість програмістам створювати різні типи безпечних і надійних програм, які працюють у .NET. C#, належить до широковідомого сімейства мов C і буде знайома кожному, хто використовує C, C++, Java або JavaScript.

C# — це об'єктно-орієнтована, компонентно-орієнтована мова програмування. Вона надає мовні конструкції, які безпосередньо підтримують цю концепцію. Це робить C# придатним для створення та використання програмних компонентів. З моменту появи C# був збагачений функціями для підтримки нових робочих навантажень і сучасних рекомендацій щодо розробки програмного забезпечення.

Нижче описані деякі особливості мови C#, котрі дозволяють користувачу створити надійні та стабільні програми. Збір сміття – автоматичний процес, який звільняє пам’ять, зайняту недоступними об’єктами, котрі не використовуються. Тип Null забезпечує захист змінних, які посилаються на вибраний об’єкт.

Обробка винятків забезпечує структурований і масштабований підхід до виявлення та відновлення помилок. Лямбда-вирази підтримують методи функціонального програмування.

Загальний шаблон, створений синтаксисисом LINQ, дозволяє працювати з даними з будь-якого джерела. Синтаксис для створення розподілених систем забезпечує підтримку мови для асинхронних операцій.

C# — єдина система типів. Усі типи C#, включаючи самі прості, типу int і double, успадковуються від одного кореневого типу object. Вони використовують загальний набір операцій, у яких значення будь-якого типу можна зберігати, передавати й маніпулювати. Крім того, C# підтримує визначені користувачем типи посилань і типи значень.

C# дає можливість програмісту динамічно вибирати об'єкти та зберігати спрощені структури в стеку. C# надає ітератори, що дають можливість розробникам класів колекцій визначати поведінку клієнтського коду.

C# акцентує увагу на управлінні версіями, щоб гарантувати, що програми та бібліотеки залишаються сумісними з часом. На проблеми версій значно впливають аспекти розробки C#, такі як роздільні модифікатори virtual і override, правила дозволу на навантаження методів та підтримка явних декларацій членів інтерфейсу[12].

Для виконання поставленого завдання були використані наступні бібліотеки:

System – містить базові класи, що визначають типи значень і посилань, які часто використовуються, події та обробники подій, інтерфейси, атрибути та винятки обробки. Наприклад:

```
DateTime ddt2=DateTime.Now;
System.TimeSpan t=ddt2-ddt1;
test=t.Milliseconds;
int t2=t.Seconds;
MessageBox.Show(Convert.ToString(t2*1000+test));
```

System.Windows.Forms – використовується для візуальної обгортки програми — вікон, кнопок, текстових полів, тощо. Наприклад:

```
string start1=textBox1.Text;
start1=start1.ToLower();
string finish1="";
if(radioButton1.Checked==true){
finish1=Rotate.ToCode(start1);
}
else {
finish1=Rotate.ToDecode(start1);
```

System.IO – призначена для роботи з файлами, їх читання, запис. априклад:

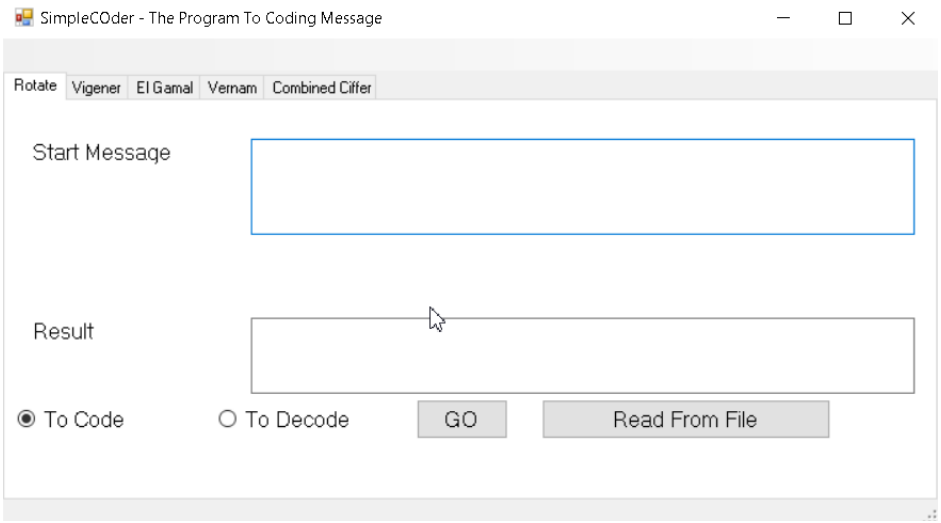
```
var sr = new StreamReader(openFileDialog1.FileName);
var sw=new StreamWriter("output.txt");
```

System.Security – забезпечує базову структуру загальномовної системи безпеки під час виконання, включаючи базові класи для дозволів. Наприклад:

```
catch (SecurityException ex)
{
MessageBox.Show("Error");
}
```

3.2 Опис розробленого продукту

Після відкриття Sharp Develop та запуску потрібного проєкту, на екрані



з’явиться вікно як показано на рисунку 3.2

Рисунок 3.2 – Вікно програми

У верхній частині вікна розташовані вкладки. Вкладка — це елемент інтерфейсу користувача, який дозволяє зручно перемикатися між кількома відкритими документами або певним набором елементів інтерфейсу в одному вікні програми, коли їх доступно більше одного, і може показувати лише один з них, котрій обрав користувач.

Вкладка представляє собою значущий напис. Частіше всього вкладки розташовані одна за одною горизонтально, рідше вертикально. Клік мишею по вкладці робить її активною, на екрані відображається відповідний вміст.

У залежності від обраної вкладки користувач обирає котрим шифром бажає зашифрувати повідомлення. У програмі реалізовані шифр перестановки, шифр Віженера, шифр Вернама, комбінований шифр.

Шифр перестановки в даному випадку працює наступним чином: вхідне повідомлення читається зліва направо, розбивається на дві частини і вже потім зчитування йде зверху вниз. Для ускладнення шифру реалізований метод Noise.Add(), котрий у якості вхідного параметра має текст, який підлягає шифруванню. Метод додає кожен другий символ із рядка:

```
Adder="plmkoiujnbhygvftrdcxsewazqїхзщшгнекуцйфівапролджєюбьтимсчя 1234567890".
```

Аналогічно для видалення зайвих літер при дешифруванні використовується метод Noise.From().

Вперше система Віжинера була опублікована в 1586 році. І є однією з найстаріших і найвідоміших багатобуквенних систем. Він названий на честь Блеза Вігінера, французького дипломата 16-го століття, який розвивав і вдосконалював криптологію.

Шифр Віженера дуже схожий на шифр Цезаря, можна сказати, шо це послідовність шифрів Цезаря з різними значеннями зсуву. Суть алгоритму шифрування є простою, тобто до першої літери тексту застосовується наприклад зсув на 7 символів, до другої на 14, і так далі. Значення зсуву визначається ключовою фразою, у якій кожна літера слова позначає необхідний зсув. Ключова фраза повторюється до тих пір, поки не буде рівна довжині вхідного повідомлення, тобто кожен символ отримає своє значення зсуву. Спочатку знаходимо положення символу ключової фрази в алфавіті шифрування - це і є значення зсуву.

Для шифрування використовується метод Vigener.ToCode(), який в якості вхідних параметрів має тестове повідомлення та ключ. Для маніпулювання символами використовується алфавіт:

```
string VigABC="йцукенрпавіфячсмитьбю.єждлогшщзхї:;.,asdfghjklmnbvcxzqwertyuiop1234567890\"-";.
```

Якщо користувач не ввів ключ та існує файл key.txt, використовується перевантажений метод, який у якості вхідного параметра має лише введений текст. У такому разі ключ читається з файлу. В іншому випадку, якщо ключ не введений та файл не існує, у якості ключа використовується ключ за замовчуванням - "turbulent". Аналогічно відбувається дешифрування при виклику Vigener.DeCode().

Схема Ель-Гамаля (Elgamal) - криптосистема з відкритим ключем, заснована на складності обчислення дискретних логарифмів у кінцевому полі.

Схему було запропоновано Тахером Эль-Гамалем в 1985 році. Ель-Гамаль розробив один із варіантів алгоритму Діффі-Хеллмана. На відміну від RSA алгоритм Ель-Гамаля не був запатентований і тому став дешевшою альтернативою, тому що не була потрібна оплата внесків за ліцензію. Вважається, що алгоритм підпадає під дію патенту Діффі-Хеллмана.

Першим кроком є генерація ключів: генерується випадкове просте число p, обирається ціле число g, що є первісним коренем p. Далі вибирається число x таке, що ($1<x<p-1$). Обчислюється $y = g^x \bmod p$.

Відкритим ключем є (y,g,p), закритим ключем - число x.

Процес шифрування:

Обирається випадкове ціле число – сессійний ключ k. Вхідне повідомлення приймаємо за H. Обчислюємо: $a = g^k \bmod p$, $b = y^k H \bmod p$.

Результат a,b – наш зашифрований текст. Для розшифрування використовуємо формулу: $H = ba^{(p-1-x)} (\bmod p)$.

Для шифрування та дешифрування використовується алфавіт:

```
ABC="1234567890abcdefghijklmnopqrstuvwxyzйцукенгшщзхїфівапролджєячсмитьбю\"-";
```

Для генерації ключів взято значення: $p = 71$, $g = 2$, $x = 30$, $k = 3$. У результаті шифрування на екрані буде виведено результат у вигляді $a = b$ (string res=res1+"="+res2;). При виконанні методів шифрування ElGamal.ToCode() і дешифрування ElGamal.ToDecode() вхідним параметром є лише повідомлення.

Шифр Вернама вважається абсолютно стійким шифром. Його умовою є використання постійно змінного ключа. Також ключ повинен бути більшим або дорівнювати довжині вхідного повідомлення, символи ключа повинні бути розташовані випадковим чином. Шифрування представляє собою додавання по модулю n, де n – потужність алфавіту кодування, символу відкритого тексту та символу ключового слова. Для даного алгоритму використовується алфавіт:

```
ABC="qazxswedcvfrtgbnhyujm,kiol./;ярфйцічсвукамипенртьогшлбюдщзжєхї1234567890,?: \"-";.
```

Ключ прописаний у файлі vernam.txt. У якості вхідного параметра метод шифрування і дешифрування приймають змінну d та вхідне повідомлення. Змінна

d – день місяця і відповідно номер рядку у файлу, визначається за допомогою структури DateTime:

```
DateTime ddt=DateTime.Now;
int dd=ddt.Day;
```

3.3 Опис роботи програми

Даний проєкт розроблений на Sharp Develop, створений для роботи на персональному комп'ютері під керуванням операційної системи Windows.

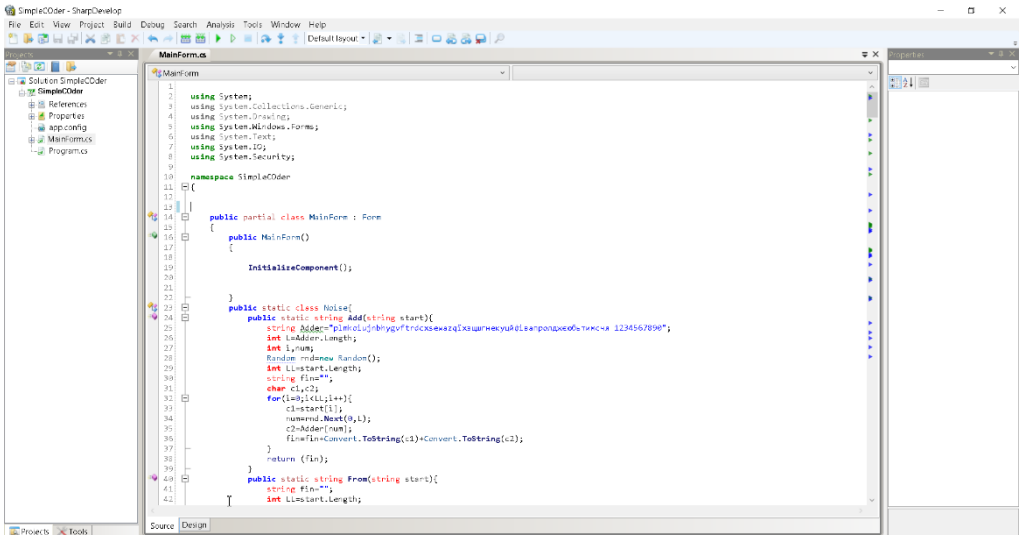


Рисунок 3.2 – Середовище програмування Sharp Developer

При запуску програми з’являється вікно, у якому потрібно обрати метод шифрування, дію (шифрування чи дешифрування) та ввести вхідний текст. Усі методи шифрування підтримують кирилицю та латиницю, числа та інші символи, що дозволяє не обмежувався при вводі тексту. Для програми розроблений зручний графічний інтерфейс, який буде зрозумілий будь-кому.

На рисунку 3. 3 наведений приклад роботи шифра Віженера, при використанні кодового слова «життя».

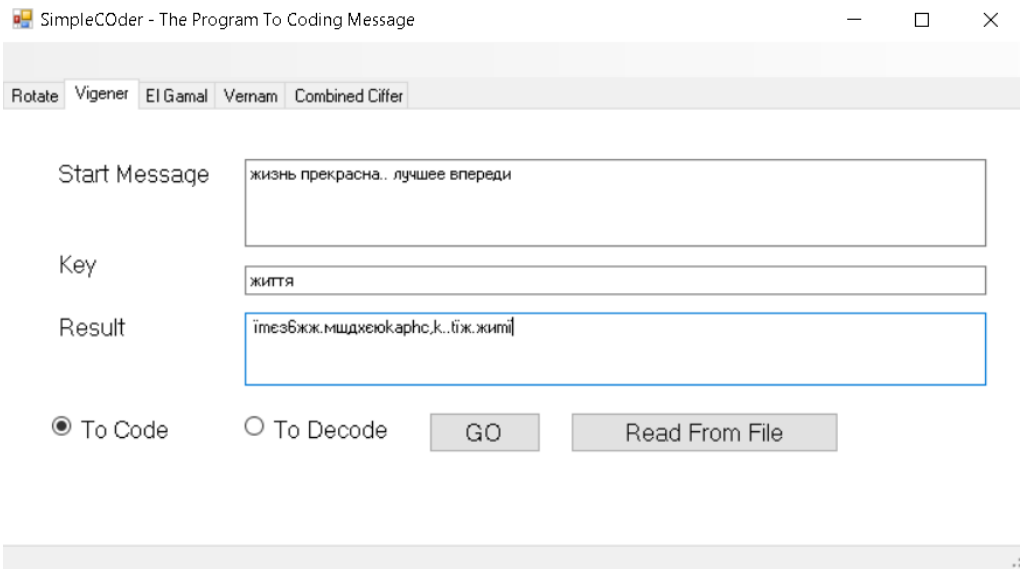


Рисунок 3.3 – Результат шифра Віженера

Для перевірки введемо отриманий зашифрований текст та позначимо «To Decode» . Результат виконання дій наведений на рисунку 3.4.

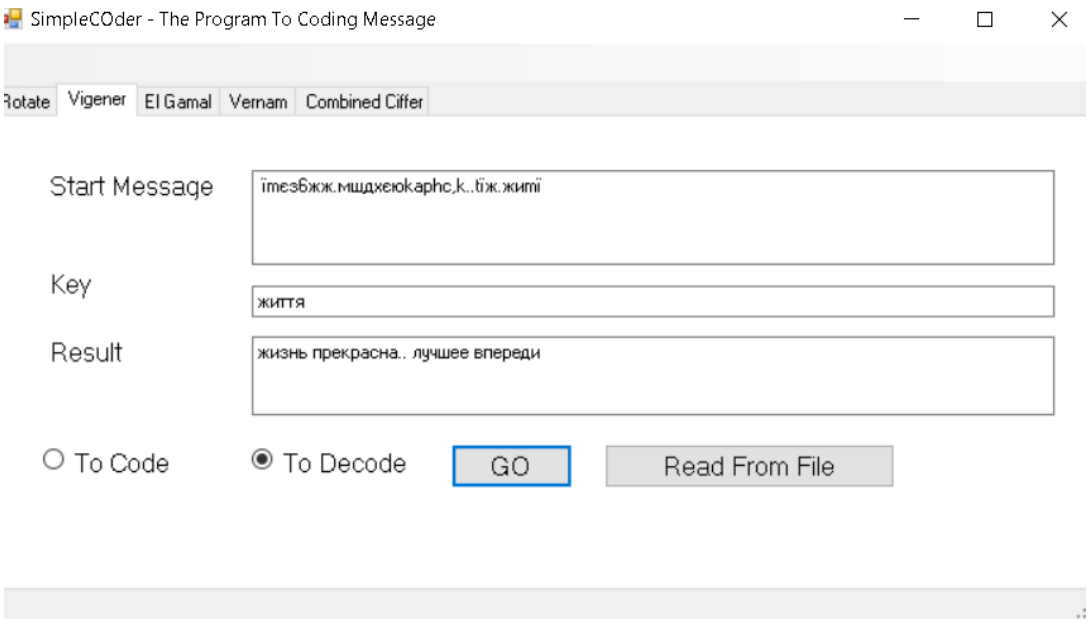


Рисунок 3.4 – Дешифрування шифру Віженера

Кожен з запропонованих шифрів підтримує шифрування тексту з файлу. При натисканні кнопки «Read From File» відкривається діалогове вікно, у якому

користувач може обрати файл для шифрування. У результаті отримуємо зашифрований текст у файлі output.txt та час, за який програма виконала необхідні дії. Приклад вибору файлу наведений на рисунку 3.5.

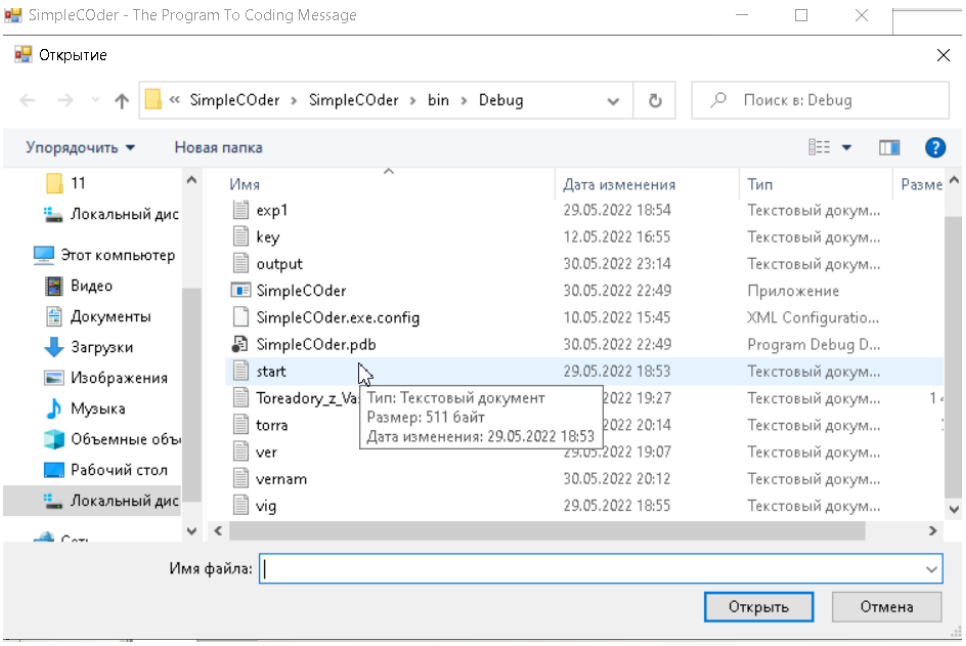


Рисунок 3.5 – Вибір файлу

Для шифру Вернама був створений файл з 31 рядком рандомних символів, кожен рядок різної довжини. При використанні даного методу шифрування програма знаходить, який день місяця у момент запуску, та використовує відповідний рядок з ключем. В інший день отримувач не зможе дешифрувати повідомлення, що надає стійкості даному методу.

Остання вкладка «Combined Ciffer» виконує шифрування, використовуючи всі попередні шифри. За принципом: перший рядок повідомлення – шифр переміщення, другий – Віженера і так далі:

```
switch(num) {
case 0:fin=Rotate.ToCode(st); break;
case 1:fin=Vigener.ToCode(st); break;
case 2:fin=ElGamal.ToCode(st); break;
case 3:fin=Vernam.ToCode(st,dd); break;
}
```

В аналогічному порядку відбувається дешифрування. Результат виконання цього шифру зображений на рисунку 3.7.

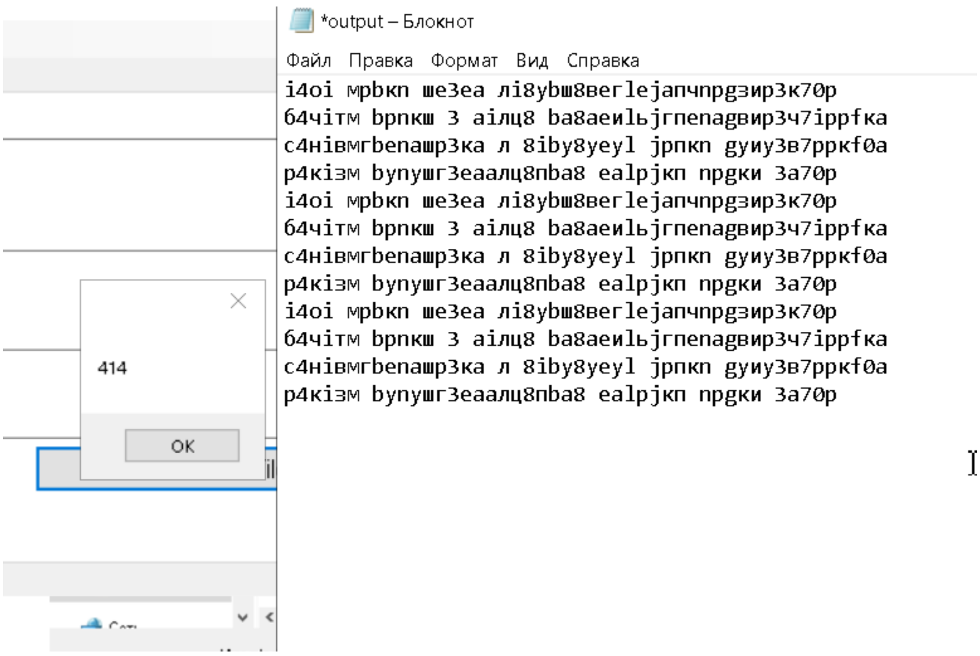


Рисунок 3.6 – Результат роботи комбінованого шифру

Отже, програма свої завдання виконує. Варто зазначити, програма не забезпечує абсолютної криптостійкості, але цілком підходить для користування в

побуті, для приховування таємної інформації або переписки з товаришем у месенджері, тощо.

4 ОХОРОНА ПРАЦІ

Найвища цінність людини – це життя і здоров’я, захист яких потребує значних зусиль. Тому охорона праці є важливою складовою будь-якого підприємства, установи, організації, адже безпека і здоров'я працівника дозволяють зробити виробничий процес чіткішим, що підвищить прибутковість самої компанії.

У даному розділі розглядається аналіз охорони праці для спеціаліста інформаційної безпеки, робоче місце якого знаходиться в приміщенні офісного типу.

Процес планування робочого місця передбачає чіткий порядок і сталість розміщення предметів, засобів праці і документації. Раціонально організоване робоче місце спеціаліста з інформаційної безпеки допомагає збільшити ефективність праці і запобігти професійним захворюванням.

Робоче місце становить собою стіл, робоче крісло та розміщений на ньому персональний комп’ютер з необхідною периферією (принтер, клавіатура, мишка).

Згідно з ГОСТ 12.0.003-74, у якому описана сукупність факторів виробничого середовища, що впливають на здоров’я і працездатність людини, були визначені шкідливі й небезпечні фактори, які присутні на робочому місці спеціаліста з інформаційної безпеки:

- відсутність або нестача природного світла;
- недостатня освітленість робочої зони;
- підвищений рівень шуму на робочому місці;
- підвищена чи знижена температура повітря робочої зони;
- підвищене значення напруги в електричному ланцюзі, замикання якого може статися через тіло людини;
- підвищений рівень статичної електрики;
- підвищений рівень іонізуючих випромінювань у робочій зоні.

Належне освітлення робочого місця – один із найважливіших факторів безпеки праці. Освітленість робочого місця спеціаліста з інформаційної безпеки

повинна бути такою, щоб працівник міг без напруги зору виконувати свою роботу.

За недостатньої освітленості візуальне сприйняття знижується. Через постійне зорове напруження настає зорова втома. Окрім захворювань зору та головного болю, при недостатньому освітленні працюючий близько нахиляється до обладнання, у результаті чого виникає небезпека нещасного випадку.

Аналогічно, робота з надлишковою освітленістю протягом значного проміжку часу може призвести до підвищення чутливості очей до світла з характерним виділенням сліз, запаленням слизової оболонки або роговиці ока.

Фактична освітленість на робочому місці складає 300 лк. Згідно з ДБН В.2.5-28-2006 освітленість поверхні під час роботи за комп'ютером або роботи з документами повинна складати не менше ніж 500лк. Виходячи з цього, недостатність природного освітлення компенсується штучним[16].

Ще одним з факторів, які несприятливо впливають на здоров'я людини і може спричинити хворобливі наслідки, є шум. Шум – це хаотична сукупність різних за силою і частотою звуків, що заважають сприйняттю корисних сигналів. Це усі неприємні та небажані звуки, які заважають працювати належним чином, сприймати потрібні звуки, відпочивати. У працівника можуть з'явитися симптоми перевтоми, послаблюється увага, підвищується нервова збудливість, знижується працездатність.

Відповідно до документа ДСН 43.3.6 037-99 рівень шуму на робочому місці спеціаліста з інформаційної безпеки не повинен перевищувати 50 дБА. Підвищення допустимих норм шуму може бути викликано роботою принтера та кондиціонера. Для зниження рівня шуму стіни приміщення повинні бути облицьовані звукопоглинальними матеріалами – звукоізоляційними плитами з пінополіуретану.

Одним із найважливіших фізичних факторів є мікроклімат. У приміщеннях на робочих місцях повинні забезпечуватися оптимальні значення параметрів мікроклімату: температури, відносної вологості і рухливості повітря відповідно до ДСН 3.3.6.042-99 (Таблиця 4.1) [14].

Таблиця 4.1 – Оптимальні значення параметрів мікроклімату

Пора року	Категорія робіт	Температура повітря, град. С	Відносна вологість повітря, %	Швидкість руху повітря, м/с
Холодна	легка-1 а	22 – 24	40 — 60	0,1
Тепла	легка-1 а	23 — 25	40 — 60	0,1

Фактичні показники мікроклімату для теплої пори року:

- температура повітря 24-26 С;
- вологість повітря – 30%;
- швидкість руху повітря - 0,1.

Рівні позитивних і негативних іонів в повітрі повинні відповідати санітарно-гігієнічним нормам № 2152-80 (Таблиця 4.2).

Таблиця 4.2 – Рівні позитивних і негативних іонів

Рівні	Кількість іонів на 1 см куб. повітря	
	n +	n —
Мінімально необхідні	400	600
Оптимальні	1500 — 3000	3000 — 5000
Максимально припустимі	50000	50000

Для підтримки допустимих значень мікроклімату та концентрації позитивних і негативних іонів у приміщенні необхідно передбачати установки або прилади зволоження та / або штучної іонізації, кондиціювання повітря.

Положення тіла, у якому працівники знаходиться більшу частину часу і виконує свою роботу, називається робоча поза. Дуже важливо правильно підібрати робочу позу, це сприяє збереженню працездатності працівника та

зменшенню втоми. Зручна робоча поза має забезпечувати стійкість положення корпуса, ніг, рук, голови працівника під час роботи, достатній огляд робочого місця, свободу дій і швидку зміну робочих рухів, мінімальні затрати енергії та максимальну продуктивність праці.

Тривале перебування працівника в неправильній і напруженій позі може призвести до таких захворювань, як сколіоз, варикозне розширення вен, плоскостопість тощо. Доведено, що робота в зігнутому положенні збільшує затрати енергії на 20%, а при значному нахиленні — на 45% у порівнянні з прямим положенням корпуса. Зручною вважається робоча поза, яка відповідає характеру роботи, яку виконує працівник, і вимогам гігієни та фізіології праці. Площа робочого місця має бути такою, щоб відповідала нормам технологічного проєктування, і складала не менше 4,5 $м^2$ на одного працівника, відповідно до ГН 3.3.5-8-6.6.1 [15].

Оскільки сучасну роботу важко уявити без використання електричних приладів на будь-якому підприємстві, приділяють значну увагу комплексу організаційних і технічних заходів з метою захисту працівників від впливу електричного струму. Одним з головних завдань електробезпеки є унеможливлення загорання внаслідок короткого замикання та перевантаження проводів.

Заходи електробезпеки, на які необхідно звернути особливу увагу:

- раціональне розподілення навантаження на всі приміщення офісу;
- раціональне розподілення електромережі за призначенням;
- якість комплектуючих електромережі;
- наявність потенціалу для збільшення навантаження;
- усі пристрої повинні підключатися до електромережі за допомогою справних штепсельних з’єднань і електророзеток заводського виготовлення.

Отже, під час аналізу робочого місця спеціаліста з інформаційної безпеки було виявлено ряд шкідливих факторів, які шкодять здоров’ю працівника: підвищений рівень шуму, недостатня освітленість робочої поверхні, неправильна

робоча поза, електромагнітне випромінювання, неоптимальні показники мікроклімату.

Відповідно до цих факторів можна надати наступні рекомендації:

1) використовувати кондиціонери, охолоджувачі, обігрівачі в залежності від пори року;
2) зменшувати кількість приладів, які створюють шум, правильно їх розміщувати. Використовувати звукоізоляційні та звукопоглинаючі засоби;
3) використовувати сумісне освітлення. Недостачу природного освітлення компенсувати штучним;
4) використовувати зволожувачі повітря та вологопоглиначі для регуляції рівня вологи в приміщенні;
5) дотримуватися норм щодо положення тіла при роботі за комп’ютером, описаних у документі ДСанПіН 3.3.2.007-98 [16];
6) дотримуватися правил електробезпеки.

Запропоновані рекомендації зниження шкідливого впливу виробничих чинників та раціональне облаштування робочого місця дає змогу покращити умови праці, знизити рівень травматизму та загальної захворюваності, підвищити працездатність та якість праці.

Неорганізоване і неконтрольоване горіння, внаслідок якого знищуються матеріальні цінності, називається пожежею.

Забезпечення пожежної безпеки є невід'ємною частиною діяльності підприємства щодо охорони життя та здоров'я працівників. Метою пожежної безпеки є запобігання пожежі на визначеному чинними нормативами рівні, а в разі виникнення пожежі – обмеження її розповсюдження, своєчасне виявлення, гасіння пожежі, захист людей і матеріальних цінностей.

Причини виникнення пожежі:

- необережне поводження з вогнем у побуті;
- порушення пожежних норм і правил у технологічних процесах виробництва;

- неправильне облаштування систем опалення, вентиляції, електроустаткування;
- порушення норм і правил зберігання пожежонебезпечних несумісних матеріалів;
- порушення правил користування електрообладнанням;
- невиконання протипожежних заходів щодо обладнання пожежного водозабезпечення, улаштування пожежної сигналізації, забезпечення первинними засобами пожежогасіння;
- використання відкритого вогню, паління у заборонених місцях;
- погане володіння персоналом основ пожежної безпеки;
- порушення вимог протипожежного інструктажу під час виконання робіт.

Відповідно до НАПБ Б.03.002-2007 визначили категорію приміщення, як категорія В. Оскільки приміщення оснащене електроприладами, мають бути присутні вуглекислотні або порошкові вогнегасники.

Одним з найбільш надійних засобів боротьби з пожежею є вогнегасники. У деяких випадках саме за допомогою цих засобів можна ліквідувати вогнище загоряння власними силами ще на перших незначних його проявах.

Вуглекислотні вогнегасники застосовуються для гасіння легкозаймистих та горючих рідин, твердих речовин, електроустановок, що перебувають під напругою. У вуглекислотних вогнегасниках в якості вогнегасної речовини застосовується діоксид вуглецю (вуглекислота), що практично виключає можливість доступу кисню і призводить до згасання вогнища загоряння.

Переваги вуглекислотних вогнегасників:

- універсальність застосування;
- висока надійність і відмовостійкість;
- застосування не призводить до забруднення об’єкта, на якому проводиться гасіння пожежі;
- широкий діапазоні температур використання від –40 до +50 градусів.

Не дивлячись на переваги вуглекислотного вогнегасника, його використання є небезпечним:

- токсичний вплив на людину при гасінні загоряння в замкнутому просторі;
- при попаданні на відкриті ділянки шкіри викликає сильне обмороження;
- концентрація CO_2 в навколишньому людини просторі викликає параліч дихальних шляхів.

Порошкові вогнегасники є одними з найбільш поширених засобів первинного пожежогасіння, основна складова яких є порошкоподібна суміш мінеральних солей з додаванням різних хімічних елементів, які перешкоджають злежуванню і грудкуванню, щоб у відповідальний момент реалізувати гасячу здатність вогнегасника. Причина поширеності і попиту полягає в оптимальному співвідношенні вартості пристрою до площі вогню, яку можна ліквідувати за його допомогою.

Переваги порошкових вогнегасників:

- працездатні при температурі до –50С;
- вогнегасний порошок є діелектриком;
- не виділяють токсичних газів під впливом високої температури і відкритого полум’я при гасінні;
- низька вартість;
- легкі в експлуатації (немає необхідності в частому технічному обслуговуванні);
- можливість гасіння електроустановок під напругою до 1000 В.

До недоліків і технічних обмежень потрібно згадати, що склад може вступати в реакцію з деякими матеріалами і утворювати корозійно-активні склади. Використовуючи порошкові моделі вогнегасників в закритому приміщенні, необхідно пам’ятати про те, що розпорошена суміш мінеральних солей ускладнює видимість, може створити проблеми при евакуації людей і впливати на їх органи дихання і зору. За допомогою порошкового вогнегасника можуть бути ліквідовані такі класи загорянь:

A — тверді речовини;

B — рідкі речовини;

Е — електроустановки під напругою до 1000 В.

Порошкоподібний вогнегасник так само може бути використаний для гасіння лужних і лужноземельних матеріалів, які горять без доступу кисню. Але при цьому на корпусі пристрою повинно бути відповідне маркування про наповнення його спеціальною вогнегасною речовиною [17].

Виходячи з розміру приміщення, яке складає 120 м^2 та з факту наявності у приміщенні електроприладів, можна зробити висновок про необхідність облаштування такого приміщення 5 порошковими вогнегасниками, кожен з яких повинен мати заряд вогнегасної речовини у розмірі 3 кг.

ВИСНОВКИ

Сьогодні захист інформації розглядається не тільки для захищення інформації, яка становить державну цінність, чи цінність для підприємства, а й для кожної людини.

Працюючи над даною роботою, було досліджено сучасний стан проблеми захисту інформації, досліджено властивості інформації. Наведено методи та засоби захисту інформації. Під час дослідження основну увагу було приділено криптографічним методам.

Був проведений аналіз існуючих видів алгоритмів шифрування, наведені їх головні переваги та недоліки. При створенні продукту було запропоновано використання простих алгоритмів шифрування для зберігання або передачі даних в захищеному вигляді.

Для забезпечення захисту даних було програмно реалізовано додаток, для шифрування текстової інформації, який виконує шифрування за різними шифрами. Програма була реалізована в середовищі Sharp Developer мовою програмування C#.

Програма виконує шифрування та дешифрування справно, використовується графічний інтерфейс.

Можна зробити висновок, що додаток працює добре, але може бути покращеним завдяки розширенню методів шифрування даних.

ПЕРЕЛІК ПОСИЛАНЬ

1. Про інформацію: Закон України від 02.10.1992р. № 2657 - XII. Дата оновлення 01.01.2022р. № 1089-IX. URL:
https://zakon.rada.gov.ua/laws/show/2657-12#Text

2. Верлань А.Ф., Апатова Н.В. Інформатика. Основи інформатики та обчислювальної техніки. Київ: Форум, 2003.

3. Про доступ до публічної інформації: Закон України від 13.01.2011р. №2939-VI. Дата оновлення 19.02.2022р. № 2024-IX. URL: https://zakon.rada.gov.ua/laws/show/2939-17#Text

4. Смірнова О.Ю. Загрози безпеці та пошкодження даних у комп'ютерних системах. URL:
https://sites.google.com/view/smirnovaseu/інформатика/9-клас/тема-3-урок-1

5. Слінько Т. Сучасні загрози інформаційній безпеці країни й шляхи їх подолання. URL:
https://www.constjournal.com/wp-content/uploads/issues/2021-4/pdfs/6-tetiana-slinko-suchasni-zahrozy-informatsiyniy-bezpetsi-krainy-shliakhy-ikh-podolannia.pdf

6. Гребенюк А.М., Рибальченко Л.В. Основи управління інформаційною безпекою. Дніпро: Дніпропетровський державний університет внутрішніх справ, 2020.

7. Дудикевич В.Б., Хорошко В.О., Яремчук Ю.Є. Основи інформаційної безпеки. Вінниця: ВНТУ, 2018

8. Франчук В.М. Захист інформаційних ресурсів: криптографічні та стеганографічні методи захисту даних. Київ: НПУ ім. М.П.Драгоманова, 2012р.

9. Beklemysheva A. Шифрування: типи і алгоритми. URL:
https://hostpro.ua/wiki/ua/security/encryption-types-algorithms

10. Філіп'єва М.В. Порівняння симетричного і асиметричного шифрування. URL:
http://www.konferenciaonline.org.ua/us/article/id-272/

11. Троелсен Эндрю. Возможности SharpDevelop. URL:

https://it.wikireading.ru/30685

12. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. СПб.: Питер, 2013.

13. ДБН В.2.5-28. Інженерне обладнання будинків і споруд. ПРИРОДНЕ І ШТУЧНЕ ОСВІТЛЕННЯ. [Чинний від 2006.05.15]. Київ, 2006.

14. ДСН 3.3.6.042-99. Санітарні норми мікроклімату виробничих приміщень. [Чинний від 1999.01.05]. Київ, 1999.

15. ГН 3.3.5-8-6.6.1 «Гігієнічна класифікація праці за показниками шкідливості та небезпечності факторів виробничого середовища, важкості та напруженості трудового процесу». [Чинний від 2001.12.27]. Київ, 2001.

16. ДСанПіН 3.3.2.007-98 Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. [Чинний від 1998.12.10]. Київ, 1998.

17. Лідньов А.О. Види вогнегасників. URL:

https://www.sop.com.ua/article/343-tipi-vognegasnikv

Додаток А. Лістинг програмного коду

```
using System;
using System.Windows.Forms;
using System.Text;
using System.IO;
using System.Security;

namespace SimpleCOder
{

    public partial class MainForm : Form
    {
        public MainForm()
        {

            InitializeComponent();


        }
        public static class Noise{
            public static string Add(string start){
                string
Adder="plmkoiujnbhygvftrdcxsewazqїхзщшгнекуцйфівапролджєюбьтимс
чя 1234567890";
                int L=Adder.Length;
                int i,num;
                Random rnd=new Random();
                int LL=start.Length;
                string fin="";
                char c1,c2;
                for(i=0;i<LL;i++){
                    c1=start[i];
                    num=rnd.Next(0,L);
                    c2=Adder[num];
                    fin=fin+Convert.ToString(c1)+Convert.ToStri
ng(c2);
                }
                return (fin);
            }
            public static string From(string start){
                string fin="";
                int LL=start.Length;
                int i;
                char c;
                fin=start.Substring(0,1);
                for(i=1;i<LL;i++){
                    c=start[i];
                    if(i%2==0){
                        fin=fin+Convert.ToString(c);
                    }
```

```
                }
                return(fin);

            }


        }

        public static class Vigener{
            public static string ToCode(string mess,string
key){
                string res="";
                string
VigABC="йцукенрпавіфячсмитьбю.єждлогшщзхї:;.,asdfghjkl
mnbvcxzqwertyuiop1234567890\"-";
                int Labc=VigABC.Length;
                int L=mess.Length;
                string kkey=key;
                while (kkey.Length<L){
                    kkey+=key;
                }
                int i,num,keyNum;
                char sym1=' ',sym2=' ';
                for(i=0;i<L;i++){
                    sym1=mess[i];
                    sym2=kkey[i];
                    keyNum=VigABC.IndexOf(sym2);
                    num=(VigABC.IndexOf(sym1)+keyNum)%Labc;
                    sym1=VigABC[num];
                    res+=Convert.ToString(sym1);
                }
                return(res);
            }
            public static string ToCode(string mess){
                string key="";

                if(File.Exists("key.txt")){
                       StreamReader sr=new
StreamReader("key.txt");
                       key=sr.ReadLine();
                       sr.Close();
                          }
                else{
                    key="turbulent";
                }
                string res="";
                string
VigABC="йцукенрпавіфячсмитьбю.єждлогшщзхї:;.,asdfghjkl
mnbvcxzqwertyuiop1234567890\"-";
                int Labc=VigABC.Length;
                int L=mess.Length;
                string kkey=key;
                while (kkey.Length<L){
```

```
                    kkey+=key;
                }
                int i,num,keyNum;
                char sym1=' ',sym2=' ';
                for(i=0;i<L;i++){
                    sym1=mess[i];
                    sym2=kkey[i];
                    keyNum=VigABC.IndexOf(sym2);
                    num=(VigABC.IndexOf(sym1)+keyNum)%Labc;
                    sym1=VigABC[num];
                    res+=Convert.ToString(sym1);
                }
                return(res);
            }
            public static string ToDeCode(string mess,string
key){
                string res="";
                string
VigABC="йцукенрпавіфячсмитьбю.єждлогшщзхї:;.,asdfghjkl
mnbvcxzqwertyuiop1234567890\"-";
                int Labc=VigABC.Length;
                int L=mess.Length;
                string kkey=key;
                while (kkey.Length<L){
                    kkey+=key;
                }
                int i,num,keyNum;
                char sym1=' ',sym2=' ';
                for(i=0;i<L;i++){
                    sym1=mess[i];
                    sym2=kkey[i];
                    keyNum=VigABC.IndexOf(sym2);
                    num=(VigABC.IndexOf(sym1)-keyNum);
                    if(num<0){
                        num+=Labc;
                    }
                    sym1=VigABC[num];
                    res+=Convert.ToString(sym1);
                }
                return(res);
            }
            public static string ToDeCode(string mess){
                string key="";
                if(File.Exists("key.txt")){
                       StreamReader sr=new
StreamReader("key.txt");
                       key=sr.ReadLine();
                       sr.Close();
                          }
                else{
                    key="turbulent";
                }
                string res="";
```

```
                string
VigABC="йцукенрпавіфячсмитьбю.єждлогшщзхї:;.,asdfghjkl
mnbvcxzqwertyuiop1234567890\"-";
                int Labc=VigABC.Length;
                int L=mess.Length;
                string kkey=key;
                while (kkey.Length<L){
                    kkey+=key;
                }
                int i,num,keyNum;
                char sym1=' ',sym2=' ';
                for(i=0;i<L;i++){
                    sym1=mess[i];
                    sym2=kkey[i];
                    keyNum=VigABC.IndexOf(sym2);
                    num=(VigABC.IndexOf(sym1)-keyNum);
                    if(num<0){
                        num+=Labc;
                    }
                    sym1=VigABC[num];
                    res+=Convert.ToString(sym1);
                }
                return(res);
            }
        }
        public static class Rotate{
            public static string ToCode(string start){
                //start=Noise.Add(start);
                int L=start.Length;
                if(L%2!=0){
                    start=start+"0";
                    L+=1;
                }
                int Lx=Convert.ToInt32(L/2);
                string [,]mas=new string[2,Lx];
                int i,j,k;
                for(k=0;k<L;k++){
                    i=k%2;
                    j=Convert.ToInt32(k/2);
                    mas[i,j]=Convert.ToString(start[k]);
                }
                string res="";
                for(j=0;j<Lx;j++){
                    res=res+mas[0,j];
                }
                for(j=0;j<Lx;j++){
                    res=res+mas[1,j];
                }
                res=Noise.Add(res);
                return(res);
            }
            public static string ToDecode(string start){
                start=Noise.From(start);
```

```
                int L=start.Length;
                if(L%2!=0){
                    start=start+"0";
                    L+=1;
                }
                int Lx=Convert.ToInt32(L/2);
                string [,]mas=new string[2,Lx];
                int i,j;
                for (j=0;j<Lx;j++){
                    mas[0,j]=Convert.ToString(start[j]);
                }
                for(j=Lx;j<L;j++){
                    mas[1,j-Lx]=Convert.ToString(start[j]);
                    }
                string res="";
                for(j=0;j<Lx;j++){
                    for(i=0;i<2;i++){
                        res=res+mas[i,j];
                    }
                }

                return(res);
            }
        }

        public static class ElGamal{
            public static string ToCode(string Start) {
                string ABC="1234567890
abcdefghijklmnopqrstuvwxyzйцукенгшщзхїфівапролджєячсмитьбю\"-";
                int p=71;
                int g=2;
                int x=30;
                int y=Convert.ToInt32(Math.Pow(g,x))%p;
                int k=3;
                int M=1;
                int i,j1,j2;
                int LL=Start.Length;
                int []oldNum=new int[512];
                char cc;
                int [] a=new int[512];
                int[]b=new int[512];
                string s1="",s2="";
                for(i=0;i<LL;i++){
                    cc=Start[i];
                    oldNum[i]=Math.Abs(ABC.IndexOf(cc));
                }
                for (i=0;i<LL;i++){
                    M=oldNum[i];
                    a[i]=Convert.ToInt32(Math.Pow(g,k))%p;
                    s1=Convert.ToString(a[i])+Environment.NewLi
ne;
                    b[i]=Convert.ToInt32(Math.Pow(y,k))*M %p;
                    s2=Convert.ToString(b[i])+Environment.NewLi
```

```
ne;
                }
                string res1="", res2="";
                for (i=0;i<LL;i++){
                    j1=a[i];
                    res1=res1+Convert.ToString(ABC[j1]);
                    j2=b[i];
                    if(j2<0) {
                        j2=0;
                    }
                    res2=res2+Convert.ToString(ABC[j2]);

                }
                string res=res1+"="+res2;
                return(res);
            }
            public static string ToDecode(string Start) {
                string ABC="1234567890
abcdefghijklmnopqrstuvwxyzйцукенгшщзхїфівапролджєячсмитьбю\"-
";
                int p=71;
                int g=2;
                int x=30;
                int y=Convert.ToInt32(Math.Pow(g,x))%p;
                long M=1;
                int i,j;
                int LL=Start.Length;
                int []newNum=new int[512];
                char cc;
                int [] a=new int[512];
                int[]b=new int[512];
                string s2="";
                int delta=Start.IndexOf('=');
                cc=Start[0];
                a[0]=ABC.IndexOf(cc);
                s2=Start.Substring(delta+1);
                LL=s2.Length;
                for(i=0;i<LL;i++){
                    cc=s2[i];
                    b[i]=ABC.IndexOf(cc);
                };
                int pok=p-1-x;
                int aa=a[0];
                long[] resNum=new long[255];//значення можуть
бути більшими 2^32

                for (i=0;i<LL;i++) {
                    M=Convert.ToInt64(b[i]*Math.Pow(aa,pok)%p);
                    if(M>p) {
                        M=M%p;
                    }
                    resNum[i]=M;
                }
```

```
                string res2="";
                for (i=0;i<LL;i++) {
                    j=Convert.ToInt32(resNum[i]);
                    cc=ABC[j];
                    res2=res2+Convert.ToString(cc);

                }
                string res=res2;
                return(res);
            }
        }

        public static class Vernam{
            public static string ToCode(string Start, int
d){
                string []keys=new string[31];
                string
ABC="qazxswedcvfrtgbnhyujm,kiol./;ярфйцічсвукамипенртьогшлбюдщз
жєхї1234567890,?: \"-";
                int i=0;
                StreamReader SR=new StreamReader("vernam.txt");
                for (i=0;i<31;i++) {
                    keys[i]=SR.ReadLine();
                }
                SR.Close();
                string key=keys[d];
                int L=ABC.Length;
                int LL=Start.Length;
                char c;
                int num=0,delta=0;
                string fin="";
                for (i=0;i<LL;i++){
                    c=Start[i];
                    num=ABC.IndexOf(c);
                    c=key[i];
                    delta=ABC.IndexOf(c);
                    num=(num+delta)%L;
                    if(num<0){
                        num=0;
                    }
                    c=ABC[num];
                    fin=fin+Convert.ToString(c);
                }
                string res=fin;
                return(res);
            }
            public static string ToDecode(string Start, int d){
                string []keys=new string[31];
                string
ABC="qazxswedcvfrtgbnhyujm,kiol./;ярфйцічсвукамипенртьогшлбюдщз
жєхї1234567890,?: \"-";
                int i=0;
                StreamReader SR=new StreamReader("vernam.txt");
```

```
            for (i=0;i<31;i++) {
                keys[i]=SR.ReadLine();
            }
            SR.Close();
            string key=keys[d];
            int L=ABC.Length;
            int LL=Start.Length;
            char c;
            int num=0,delta=0;
            string fin="";
            for (i=0;i<LL;i++){
                c=Start[i];
                num=ABC.IndexOf(c);
                c=key[i];
                delta=ABC.IndexOf(c);
                num=(num-delta)%L;
                if(num<0) {
                    num=num+L;
                }
                c=ABC[num];
                fin=fin+Convert.ToString(c);
            }
            string res=fin;
            return(res);
        }

    }

    void Button1Click(object sender, EventArgs e)
    {
        string start1=textBox1.Text;
        start1=start1.ToLower();
        string finish1="";
        if(radioButton1.Checked==true){
        finish1=Rotate.ToCode(start1);
        }
        else {
            finish1=Rotate.ToDecode(start1);
        }
        textBox2.Text=finish1;
    }
    void Button2Click(object sender, EventArgs e)
    {
        string start2=textBox3.Text;
        start2=start2.ToLower();
        string finish2="";
        string key2=textBox5.Text;
        if(key2.Length!=0){
        if(radioButton3.Checked==true){
            finish2=Vigener.ToCode(start2,key2);
        }
        else{
            finish2=Vigener.ToDeCode(start2,key2);
```

```
            }

            }
            else{
                if(radioButton3.Checked==true){
                finish2=Vigener.ToCode(start2);
            }
            else{
                finish2=Vigener.ToDeCode(start2);
            }
            }
            textBox4.Text=finish2;
        }
        void Button3Click(object sender, EventArgs e)
        {
            string start3=textBox6.Text;
            start3=start3.ToLower();
            string finish3="";
            if(radioButton5.Checked==true){
                finish3=ElGamal.ToCode(start3);
            }
            else{
                finish3=ElGamal.ToDecode(start3);
            }
            textBox7.Text=finish3;
        }
        void Button4Click(object sender, EventArgs e)
        {
            DateTime ddt=DateTime.Now;
            int dd=ddt.Day;
            string start4=textBox8.Text;
            start4=start4.ToLower();
            string fin4="";
            if(radioButton7.Checked==true){
            fin4=Vernam.ToCode(start4,dd);
            }
            else{
                fin4=Vernam.ToDecode(start4,dd);
            }
            textBox9.Text=fin4;
        }

        void Button5Click(object sender, EventArgs e)
        {
            DateTime ddt1=DateTime.Now;
             openFileDialog1 = new OpenFileDialog();
             if (openFileDialog1.ShowDialog() ==
DialogResult.OK)
        {
            try
            {
                var sr = new
StreamReader(openFileDialog1.FileName);
```

```
                var sw=new StreamWriter("output.txt");
                string st, fin=" ";

                    st=sr.ReadLine();
                    while(st!=null) {
                    if(radioButton1.Checked==true){
                    fin=Rotate.ToCode(st);
                    }
                    else {
                        if (radioButton2.Checked==true){
                            fin=Rotate.ToDecode(st);
                        }
                    }
                    sw.WriteLine(fin);
                    st=sr.ReadLine();
                }
                sr.Close();
                sw.Close();
            }
            catch (SecurityException ex)
            {
                MessageBox.Show("Error");
            }
        }
            DateTime ddt2=DateTime.Now;
            int test=0;

             System.TimeSpan t=ddt2-ddt1;

            test=t.Milliseconds;
            MessageBox.Show(Convert.ToString(test));
        }
        void Button6Click(object sender, EventArgs e)
        {
            DateTime ddt1=DateTime.Now;
    openFileDialog1 = new OpenFileDialog();
             if (openFileDialog1.ShowDialog() ==
DialogResult.OK)
        {
            try
            {
                var sr = new
StreamReader(openFileDialog1.FileName);
                var sw=new StreamWriter("output.txt");
                string st, fin=" ";
                     st=sr.ReadLine();
                     while(st!=null) {
                    if(radioButton3.Checked==true){
                    fin=Vigener.ToCode(st);
                    }
                    else {
                        if (radioButton4.Checked==true){
                            fin=Vigener.ToDeCode(st);
```

```
                    }
                }
                sw.WriteLine(fin);
                st=sr.ReadLine();
            }
            sr.Close();
            sw.Close();
        }
        catch (SecurityException ex)
        {
            MessageBox.Show("Error");
        }
    }
        DateTime ddt2=DateTime.Now;
         int test=ddt2.Millisecond-ddt1.Millisecond;

         System.TimeSpan t=ddt2-ddt1;
        test=t.Milliseconds;
        int t2=t.Seconds;

        MessageBox.Show(Convert.ToString(t2*1000+test));
    }
    void Button7Click(object sender, EventArgs e)
    {
        DateTime ddt1=DateTime.Now;
openFileDialog1 = new OpenFileDialog();
         if (openFileDialog1.ShowDialog() ==
DialogResult.OK)
    {
        try
        {
            var sr = new
StreamReader(openFileDialog1.FileName);
            var sw=new StreamWriter("output.txt");
            string st, fin=" ";
               st=sr.ReadLine();
                while(st!=null)  {
                if(radioButton5.Checked==true){
                fin=ElGamal.ToCode(st);
                }
                else {
                    if (radioButton6.Checked==true){
                        fin=ElGamal.ToDecode(st);
                    }
                }
                sw.WriteLine(fin);
                st=sr.ReadLine();
            }
            sr.Close();
            sw.Close();
        }
        catch (SecurityException ex)
        {
```

```
                MessageBox.Show("Error");
            }
        }
            DateTime ddt2=DateTime.Now;
             int test=0;

             System.TimeSpan t=ddt2-ddt1;
            test=t.Milliseconds;
            int t2=t.Seconds;
            MessageBox.Show(Convert.ToString(t2*1000+test));
        }
        void Button8Click(object sender, EventArgs e)
        {
            DateTime ddt1=DateTime.Now;
                int dd=ddt1.Day;
    openFileDialog1 = new OpenFileDialog();
             if (openFileDialog1.ShowDialog() ==
DialogResult.OK)
        {
            try
            {

                var sr = new
StreamReader(openFileDialog1.FileName);
                var sw=new StreamWriter("output.txt");
                string st, fin=" ";
                       st=sr.ReadLine();
                    while(st!=null)  {
                    if(radioButton7.Checked==true){
                    fin=Vernam.ToCode(st,dd);
                    }
                    else {
                        if (radioButton8.Checked==true){
                            fin=Vernam.ToDecode(st,dd);
                        }
                    }
                    sw.WriteLine(fin);
                    st=sr.ReadLine();
                }
                sr.Close();
                sw.Close();
            }
            catch (SecurityException ex)
            {
                MessageBox.Show("Error");
            }
        }
             DateTime ddt2=DateTime.Now;
             int test=0;
             System.TimeSpan t=ddt2-ddt1;
            test=t.Milliseconds;
            int t2=t.Seconds;
            MessageBox.Show(Convert.ToString(t2*1000+test));
```

```
        }


        void Button9Click(object sender, EventArgs e)
        {
    DateTime ddt1=DateTime.Now;
                int dd=ddt1.Day;
    openFileDialog1 = new OpenFileDialog();
    int i=0,num=0;
             if (openFileDialog1.ShowDialog() ==
DialogResult.OK)
        {
            try
            {
                var sr = new
StreamReader(openFileDialog1.FileName);
                var sw=new StreamWriter("output.txt");

                string st, fin=" ";
                       st=sr.ReadLine();
                    while(st!=null)  {
                    if(radioButton9.Checked==true){
                               num=i%4;
                               switch(num) {
                                       case
0:fin=Rotate.ToCode(st); break;
                                       case
1:fin=Vigener.ToCode(st); break;
                                       case
2:fin=ElGamal.ToCode(st); break;
                                       case
3:fin=Vernam.ToCode(st,dd); break
                               }

                    }
                    else {
                        if (radioButton10.Checked==true){
                                          num=i%4;//
обчислюємо номер рядка від 0 до 3
                               switch(num) {
                                       case
0:fin=Rotate.ToDecode(st); break;
                                       case
1:fin=Vigener.ToDeCode(st); break;
                                       case
2:fin=ElGamal.ToDecode(st);
break;                                      case
3:fin=Vernam.ToDecode(st,dd); break;
                               }

                        }
                    }
                    sw.WriteLine(fin);
```

```
                    st=sr.ReadLine();
                    i++;
                }
                sr.Close();
                sw.Close();
            }
            catch (SecurityException ex)
            {
                MessageBox.Show("Error");
            }
        }
             DateTime ddt2=DateTime.Now;
             int test=0;
             System.TimeSpan t=ddt2-ddt1;
            test=t.Milliseconds;
            int t2=t.Seconds;
            MessageBox.Show(Convert.ToString(t2*1000+test));
        }
    }
}
```