Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 20__ р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“ Програмне забезпечення захищеності даних методами шифрування ”

Виконав здобувач вищої освіти
IV курсу, групи КМ-21-3ск
ОПП «Комп'ютерна інженерія»
спеціальності 123 «Комп'ютерна інженерія»
_____ Соколенко В.О.
«____» _____ 20 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Пархоменко Ю. М.
«____» _____ 2024р.
Рецензент _____
_____

м. Кропивницький

Міністерство освіти і науки України
Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*
Кафедра *Кібербезпеки та програмного забезпечення*
Освітній ступінь *бакалавр*
Спеціальність *123 Комп'ютерна інженерія*

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____О.А.Смірнов
« __ » _____ 2024року

ЗАВДАННЯ
НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ
за першим (бакалаврським) рівнем вищої освіти

Соколенко Владислав Олександрович
(прізвище, ім'я, по батькові)

1. Тема роботи *Програмне забезпечення захищеності даних методами шифрування*

керівник роботи *Пархоменко Юрій Михайлович, канд. техн. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу *№___ від ___. .2024 року*

2. Строк подання студентом роботи до захисту *. .2024 р.*

3. Мета та завдання кваліфікаційної бакалаврської роботи*: Метою розробки є програмне забезпечення захищеності даних методами шифрування*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.
2.Перегляд аналогічних існуючих систем.
3. Опис і обґрунтування проектних рішень.
4. Етапи програмування системи.
5. Впровадження системи в промислову експлуатацію.
6. Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*

6. Дата видачі завдання « » 20 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної бакалаврської роботи	Строк виконання етапів кваліфікаційної бакалаврської роботи	Примітка
1.	Аналіз існуючих систем	10.03.2024 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2024 р.	
3.	Розробка моделі компонента	20.03.2024 р.	
4.	Розробка структур даних	25.03.2024 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2024 р.	
6.	Програмування алгоритмів	10.04.2024 р.	
7.	Оформлення ПЗ	17.04.2024 р.	
8.	Попередній захист роботи	22.05.2024 р.	

Студент _____ _____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Соколенко В.О. Програмне забезпечення захищеності даних методами шифрування. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2024.

В даній бакалаврській роботі розроблено програмне забезпечення для захисту даних методами шифрування.

Метою роботи є дослідження та програмна реалізація системи захисту даних шифрувальними методами

Об'єктом дослідження є процес захисту даних шифрувальними методами.

Предметом дослідження є методи захисту даних шифруванням.

Методи дослідження базуються на методах захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи захисту даних шифрувальними методами.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма призначена для виконання під управлінням операційної системи сімейства Windows.

Програму розроблено в середовищі Delphi 7.

Ключові слова: комп'ютерна інженерія, інформація, захист даних, захист даних від несанкціонованого доступу.

ABSTRACT

Sokolenko V.O. Software for data security by encryption methods. 123 Computer engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2024.

In this bachelor's work, software for data protection using encryption methods was developed.

The purpose of the work is the research and software implementation of the data protection system using encryption methods

The object of research is the process of data protection using encryption methods.

The subject of research is data protection methods by encryption.

Research methods are based on information protection methods, mathematical statistics methods, and software development methods.

The result of the work is the software implementation of the data protection system using encryption methods.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software tools are provided.

The program is intended for execution under the control of the operating system of the Windows family.

The software was developed in Delphi 7.

Keywords: computer engineering, information, data protection, data protection from unauthorised access

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

АС	-	Автоматизована система
IT	-	Інформаційні технології
AES	-	Алгоритм шифрування Advanced Encryption Standard
Blowfish	-	Криптографічний алгоритм блочносеметричного шифрування
CAST	-	Алгоритм криптозахисту Carlisle Adams & Stafford Tavares
NTFS	-	Файлова система «New Technology File System»
RAD	-	Концепція створення засобів розробки rapid application development
RSA	-	Rivest, Shamir & Adleman
SERPENT	-	Симетричний блочний алгоритм шифрування
SHA	-	Secure Hash Algorithm
TwoFish	-	Симетричний блочний алгоритм шифрування

ВСТУП

Актуальність теми. З ростом обсягу інформації, що зберігається в електронному вигляді, все ширше постає проблема збереження її конфіденційності. Одним з шляхів обмеження доступу до інформації є шифрування, тож і постає завдання розробки програмного забезпечення системи захисту даних шифрувальними методами, а ростучі вимоги до інформаційної безпеки роблять це завдання актуальним.

Мета й завдання дослідження. Метою роботи є дослідження та розробка програмного забезпечення для захисту даних методами шифрування.

Для досягнення поставленої мети визначена програма дослідження, що складається з таких завдань:

- огляд існуючих програмних засобів захисту даних;
- дослідження програмних засобів захисту даних;
- програмна реалізація системи захисту даних.

Об'єктом дослідження є процес захисту даних шляхом шифрування.

Предметом дослідження є методи шифрування.

Методи дослідження базуються на методах захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі захисту даних шифрувальними методами.

Таким чином, виходячи з вище перерахованого, дослідження та програмна реалізація системи захисту даних шифрувальними методами, є актуальним завданням, яке потребує вирішення у даній бакалаврській роботі.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Інформаційною безпекою називають заходи щодо захисту інформації від несанкціонованого доступу, руйнування, модифікації, розкриття і затримок у доступі. Інформаційна безпека включає в себе заходи по захисту процесів створення даних, їх введення, обробки і виведення. Мета інформаційної безпеки - убезпечити цінності системи, захистити і гарантувати точність і цілісність інформації та мінімізувати руйнування, які можуть мати місце, якщо інформація буде модифікована або зруйнована. «Захист інформації - сукупність організаційно-технічних заходів і правових норм для запобігання заподіянню шкоди інтересам власника інформації чи АС та осіб, які користуються інформацією». Під АС розуміється «система, що здійснює автоматизовану обробку даних і до складу якої входять технічні засоби їх обробки (засоби обчислювальної техніки і зв'язку), а також методи і процедури, програмне забезпечення». Таким чином, об'єктами захисту є: інформація, що обробляється в АС, права власників цієї інформації та власників АС, права користувача. Захист інформації полягає не лише в захисті засобів обробки інформації, а і в організації засобів захисту для підтримки певних властивостей інформації. Основними фундаментальними властивостями є конфіденційність, цілісність та доступність, адже захист інформації в більшості випадків пов'язаний з комплексним рішенням трьох завдань: забезпеченням конфіденційності інформації, забезпеченням цілісності інформації, забезпеченням доступності інформації [1].

Визначення понять конфіденційність, цілісність та доступність дається в Положенні про технічний захист інформації в Україні:

«конфіденційність» - властивість інформації бути захищеною від несанкціонованого ознайомлення»;

«цілісність» - властивість інформації бути захищеною від несанкціонованого спотворення, руйнування або знищення»;

«доступність» - властивість інформації бути захищеною від несанкціонованого блокування» [1, 2].

Порушення кожної з трьох складових призводить до порушення інформаційної безпеки в цілому. Так, порушення доступності призводить до відмови в доступі до інформації, порушення цілісності призводить до фальсифікації інформації і, нарешті, порушення конфіденційності призводить до розкриття інформації [1, 2].

Захист даних (data protection), заходи, що забезпечують доступ до конфіденційної, особливо комп'ютерної, інформації, тільки тим, хто має на це право. При цьому переслідуються дві мети: забезпечується конфіденційність особистої і ділової інформації, а також гарантується точність інформації, що зберігається. У багатьох країнах разом із застосуванням електронних методів, що перешкоджають доступу сторонніх до комп'ютерів, діє ряд законодавчих заходів. Як правило, особи, що зберігають інформацію про інших людей, підлягають реєстрації в контрольному агентстві, що зобов'язує їх підкорятися певним правилам і дає можливість перевірки даних і внесення в них необхідних змін [3].

Захист даних (Data protection) - організаційні, програмні і технічні методи і засоби, спрямовані на задоволення обмежень, встановлених для типів даних або екземплярів типів даних в системі обробки даних. Захист даних від несанкціонованого доступу (DATA SECURITY) - запобігання несанкціонованому використанню, перегляду і зміні даних, а також їх псуванню при відмові програмного або технічного забезпечення[3]. Основним призначенням систем є захист інформації від несанкціонованого доступу. З технічного погляду система призначена для перетворення

відкритого тексту в зашифрований код (зашифрування, кодування) та відновлення тексту із зашифрованого коду (розшифрування, декодування).

1.2 Область застосування

Потреба в захисті інформації від несанкціонованого доступу сторонніх осіб, може бути потрібна при ремонті, втраті, крадіжці обладнання, Тож система призначена фахівцям, які у процесі своєї професійної діяльності оперують важливою інформацією, ознайомлення з якою попри наявного несамовитого бажання чи/або намагання сторонніх, що не мають законних на те підстав, має бути обмежена, й до того ж, система може використовуватись і фізичною особою. Таким чином, системи захисту даних можуть використовуватися практично в усіх сферах людської діяльності, де за тих чи інших умов вимагається обмежити доступ до інформації: у органах державного управління і місцевого самоврядування; у військовій справі; у медицині; у всіх наявних галузях промисловості, будівництві та транспорті; у торгівлі; у туризмі; у юриспруденції; у банківській справі та страхуванні; науково-дослідницькими та освітніми установами; архівами та фізичними особами.

Таким чином, виходячи з вищенаведеного, дослідження та програмна реалізація системи захисту даних є актуальною задачею, яка потребує вирішення у даній бакалаврській роботі.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським рівнем вищої освіти)

Захист конфіденційності даних - одна з ключових проблем сучасних інформаційних технологій, для розв’язання якої застосовуються програми, що базуються на методах шифрування; і на даний час вже існує велика кількість програмних рішень, але актуальність конфіденційності інформації при її збереженні вимагає все нових і нових розробок, проте загалом побудова подібних систем базується на одних і тих же принципах.

Шифрування як технологія захисту даних

Найкращою наявною технологією захисту даних на поточний момент є шифрування. яке широко застосовується в системах як при зберіганні, так і передаванні інформації.

І так, шифрування - це процес кодування інформації з метою запобігання несанкціонованого доступу. У разі викрадення або витоку зашифровані дані будуть недоступні для прочитання без відповідного ключа [4].

Технологічно, як процес, шифрування - це оборотне перетворення даних з метою приховування інформації [5].

Шифрування відбувається із застосуванням криптографічного ключа. Ключ - це певна кількість символів, сформованих вільним чином з символів, що доступні у системі шифрування [5].

Загалом виділяють два методи шифрування: симетричне та асиметричне [5, 6].

Метод симетричного шифрування базується на використанні одного криптографічного ключа для шифрування і дешифрування даних.

Асиметричне шифрування (метод відкритого ключа) базується на використанні пари відкритого (публічного) та закритого (секретного) ключів. В асиметричному шифруванні ключі співпрацюють у парі, інформація шифрується відкритим ключем, а розшифровування відбувається тільки за допомогою відповідного секретного ключа (неможливим є використовування відкритого ключа із однієї пари та секретного ключа із іншої пари, математичними залежностями пов'язані всі пари асиметричних ключів) [5, 6].

Процес шифрування проходить за наступним алгоритмом: вибирається певний математичний алгоритм шифрування; на його вхід подається текст і пароль; на виході отримуємо зашифрований текст.

Процес дешифрування проходить за алгоритмом: вибирається визначений математичний алгоритм дешифрування; на його вхід подається зашифрований текст і пароль; на виході отримуємо відтворений текст.

Залежно від принципу роботи алгоритми симетричного шифрування модна поділити на два типи: блокові та потокові . Потокове шифрування даних передбачає обробку кожного біта інформації з використанням гамування , тобто зміни цього біта за допомогою відповідного йому біта псевдовипадкової секретної послідовності чисел, яка формується на основі ключа і має ту ж довжину, що і шифровані повідомлення. Блокові алгоритми шифрують дані блоками фіксованої довжини (якщо все повідомлення або його фінальна частина менші розміру блоку, система доповнює його передбаченими алгоритмом - символами, які так і називаються - доповненням). Для потокового шифрування на даний час використовують алгоритми: RC4; Salsa20; HC-256; WAKE. До актуальних блокових алгоритмів відносяться: AES; RC5; Blowfishя; Twofish. До асиметричних алгоритмів шифрування належать RCA та ECC. [5, 6, 7].

Якщо оцінювати алгоритми шифрування з метою визначити найкращий, приходимо до висновку і кожен з них має як свої переваги, так і недоліки. Метод симетричного шифрування відмінно підходить для

швидкого шифрування великих обсягів даних, але не забезпечують перевірку особистості, що є необхідним, коли мова заходить про безпеку в Інтернеті. З іншого боку, асиметричне шифрування надає доступ до даних передбачуваного одержувача, однак ця перевірка робить процес шифрування занадто повільним. З точки зору безпеки, асиметричне шифрування, безсумнівно, краще, оскільки воно забезпечує аутентифікацію;, проте продуктивність симетричного шифрування є значною перевагою, особливо коли справа заходить про значні об'єми [5, 6].

Серед загальних недоліків шифрування, як методу захисту даних в системі з великою кількістю користувачів в інформаційному середовищі, слід відзначити те, що зазвичай на практиці необхідно використовувати також і значну кількість ключів, а умова ж регламентного оновлення ключів (необхідність частої зміни ключів для запобігання втрати їхньої ефективності).

Програми шифрування

Програми шифрування даних здебільшого розглядаються як системні інструменти; й на даний час досить обширно представлені на ринку програмного забезпечення як безкоштовними, так і комерційними продуктами. Повний всеохоплюючий огляд такого різноманіття неможливий в рамках даної роботи. Отже обмежимося деякою множиною представників даного класу програм. Вибір програмного забезпечення для огляду не носить ніякого системного характеру , а здійснюється лише за власними уподобаннями.

TrueCrypt

TrueCrypt (розробник TrueCrypt Foundation) - комп'ютерна програма для шифрування на «льоту» для ОС сімейств MS Windows, Linux і Mac (шифрування «на льоту» - on-the-fly encryption - означає, що дані автоматично шифруються або дешифруються прямо під час їх зчитування або запису, не відволікаючи користувача), рис.2.1. За допомогою TrueCrypt можна створювати віртуальні зашифровані логічні диски і повністю

шифрувати розділ жорсткого диска або іншого носія інформації [8,9,10,11] .

TrueCrypt має наступні можливості:

- створення зашифрованого віртуального диска;

- забезпечення двох рівнів правдоподібного заперечення наявності зашифрованих даних, необхідного в разі вимушеного відкриття пароля користувачем;

- переносимість, що дозволяє запускати TrueCrypt без установки в операційній системі;

- підтримка створення зашифрованого динамічного файлу на дисках NTFS;

- шифрування системного фізичного або логічного диска для Microsoft Windows-систем з дозазвантаженою аутентифікацією;

- зміна паролів і ключових файлів для томи без втрати зашифрованих даних;

- можливість використовувати TrueCrypt на комп'ютері з правами звичайного користувача [10].

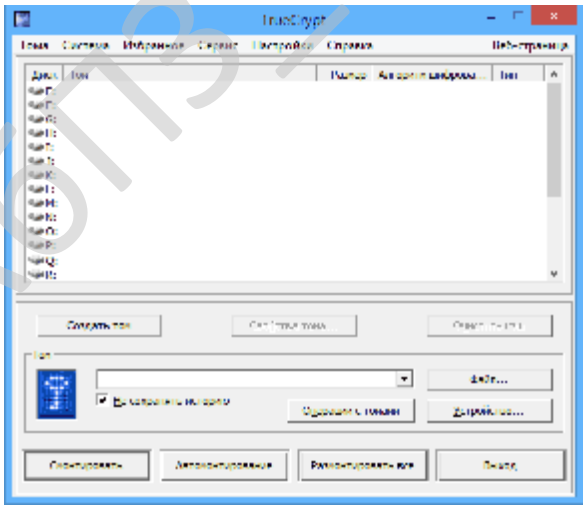

Рисунок 2.1 -TrueCrypt

Одна з основних особливостей програми TrueCrypt - відсутність в заголовку створеного «диска» специфічної сигнатури, характерної для інших

подібних програм, що робить неможливим ідентифікувати його, оскільки жодна з частин віртуального диска не відрізняється від випадкових даних.

TrueCrypt дозволяє створювати віртуальний зашифрований логічний диск, що зберігається у вигляді файлу. Також за допомогою TrueCrypt можна повністю зашифрувати розділ жорсткого диска або іншого носія інформації. Всі збережені дані в томі TrueCrypt повністю шифруються, включаючи імена файлів і каталогів.

У список підтримуваних TrueCrypt алгоритмів шифрування входять AES, Serpent і Twofish. Можливе використання каскадного шифрування різними шифрами, наприклад: AES+Twofish+Serpent.

Всі алгоритми шифрування використовують режим XTS.

TrueCrypt дозволяє вибрати одну з трьох хеш-функцій: HMAC-RIPEMD-160, HMAC-Whirlpool, HMAC-SHA-512 для генерації ключів шифрування, солі і ключа заголовка.

Для доступу до зашифрованих даних можна застосовувати пароль (ключову фразу), ключові файли (один або декілька) або їх комбінації.

Забезпечення двох рівнів правдоподібного заперечення наявності зашифрованих даних, необхідного в разі вимушеного відкриття пароля користувачем.

Підтримка створення зашифрованого динамічного файлу на дисках NTFS. Такі томи TrueCrypt збільшуються в розмірі в міру накопичення нових даних аж до зазначеного максимального розміру.

Змонтований том TrueCrypt подібний до звичайного логічного диску, тому з ним можна працювати за допомогою звичайних утиліт перевірки та дефрагментації файлової системи.

Переваги:

- зрозумілий інтерфейс програми;

- максимальний захист;

-безкоштовне розповсюдження.

Недоліки:

- більше не підтримується розробником;

- багато функцій не призначені для новачків.

Folder Lock

Folder Lock (розробник NewSoftwares, LLC) - комп'ютерна програма, що призначена для шифрування файлів і папок, рис. 2.2. Програма містить в собі цілий комплекс інструментів, які служать для дотримання таємності. Для збереження даних утиліта створює спеціальні зашифровані сховища, звані Lockers, куди можна помістити скільки завгодно файлів і папок і захистити їх паролем. Folder Lock також вміє обмежувати доступ до USB-накопичувача або CD, володіє прихованим режимом роботи, відстежує спроби злому [12, 13].

Folder Lock має наступні можливості:

- робота з віртуальними зашифрованими дисками;

- шифрування повідомлень електронної пошти;

- робота з архівами, що саморозшифровуються;

- хмарне резервне копіювання;

- приховування окремих файлів;

- приховування розділів жорсткого диска;

- робота по зберіганню платіжної інформації[14].

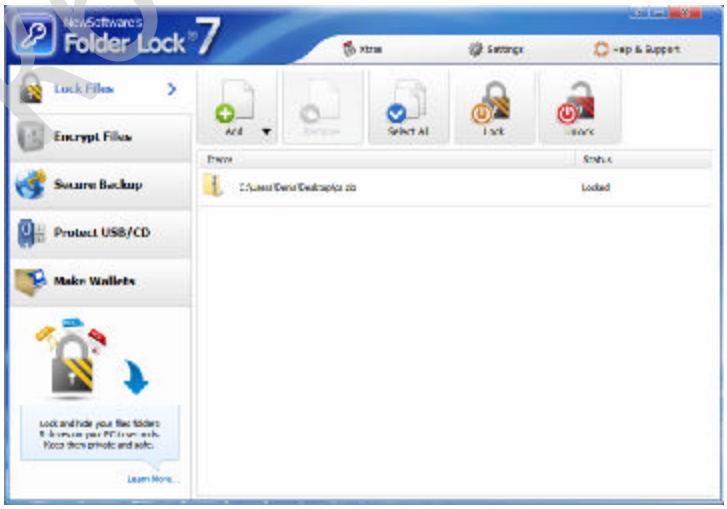

Рисунок 2.2 - T Folder Lock

Переваги програми Folder Lock:

- привабливий і зрозумілий інтерфейс, який сподобається починаючим користувачам, які володіють англійською мовою;

- прозоре шифрування «на льоту», створення віртуальних зашифрованих дисків, з якими можна працювати як зі звичайними дисками.;

- можливість резервного онлайн-копіювання і синхронізації зашифрованих контейнерів (сейфів);

- можливість створення саморозшифровуючих контейнерів на USB/CD/DVD-дисках.

Недоліки програми:

- немає широкої мультимовної підтримки, що ускладнить роботу з програмою користувачів, які не знайомі з англійською мовою;

- сумнівні функції Lock Files (яка просто приховує, а не «замикає» файли) і Make Wallets (малоефективна без експорту інформації);

- відсутність можливості підписання файлів, перевірки цифрового підпису;

- при відкритті сейфа не дозволяє вибрати букву диска, яка буде призначена для віртуального диска, який відповідає сейфу (у налаштуваннях програми можна вибрати тільки порядок, в якому програма буде призначати букву диска за зростанням від A до Z або за спаданням від Z до A);

- відсутня інтеграція з поштовими клієнтами, є тільки можливість зашифрувати вкладення;

- висока вартість хмарного резервного копіювання.

PGP Desktop

Pretty Good Privacy (PGP, розробник PGP Corporation) - комп'ютерна програма захисту інформації, яка забезпечує криптографічний конфіденційність і аутентифікацію, рис 2.3. PGP Desktop вміє шифрувати файли і каталоги, в тому числі і в локальній мережі, захищати поштові вкладення і повідомлення, створювати шифровані віртуальні диски,

безповоротно видаляти дані шляхом багатопрохідного перезапису[11,14, 15].

PGP Desktop має наступні можливості:

- робота з віртуальними зашифрованими дисками;
- шифрування всього розділу;
- шифрування системного диска;
- шифрування повідомлень електронної пошти;
- шифрування файлів;
- робота з ЕЦП підписами;
- ЕЦП перевірка;
- робота з архівами, що саморозшифровуються;
- робота з власним сервером ключів [14].

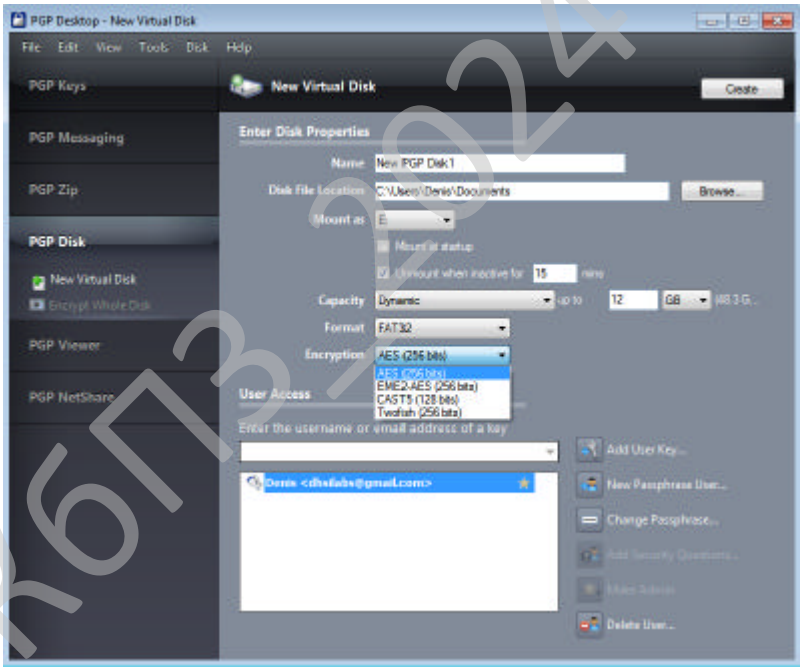

Рисунок 2.3 - PGP Desktop

Програма вбудовується в оболонку (Провідник), а доступ до функцій здійснюється через контекстне меню Провідника. У контекстному меню є функції шифрування, підпису файлу тощо. Досить цікавою є функція створення саморозшифровуючого архіву - за принципом саморозгортальних, тільки замість розпакування архіву також ще і розшифровується. Втім, у програм Folder Lock і CyberSafe також є аналогічна функція. Доступ до

функцій програми можна отримати через системний трей. Команда Open PGP Desktop відкриває основне вікно програми.

PGP Keys - керування ключами (як власними, так і імпортованими з keyserver.pgp.com).

PGP Messaging - управління службами обміну повідомленнями. При установці програма автоматично виявляє ваші облікові записи і автоматично шифрує комунікації AOL Instant Messenger.

PGP Zip - управління зашифрованими архівами. Програма підтримує прозоре і непрозоре шифрування. Цей розділ якраз і реалізує непрозоре шифрування.

PGP Disk - це реалізація функції прозорого шифрування. Програма може шифрувати весь розділ жорсткого диска (або навіть весь диск, або створити новий віртуальний диск (контейнер). Тут же є функція Shred Free Space, яка дозволяє затерти вільний простір на диску.

PGP Viewer - тут можна розшифрувати PGP-повідомлення та вкладення.

PGP NetShare - засіб «розшарування» папок, при цьому «кулі» шифруються за допомогою PGP, а у вас є можливість додати/видалити користувачів (користувачі ідентифікуються на основі сертифікатів), які мають доступ до «кулі».

Переваги програми PGP Desktop:

- повноцінна програма, що використовується для шифрування файлів, підписання файлів і перевірки електронного підпису, прозорого шифрування (віртуальні диски і шифрування всього розділу), шифрування електронної пошти;

- підтримка сервера ключів keyserver.pgp.com;

- можливість створення саморозшифровуючих архівів;

- можливість шифрування системного жорсткого дискк;.

- наявність функції PGP NetShare;

- можливість затирання вільного місця;

- інтеграція з "Провідником."

Недоліки програми:

- відсутність мультимовної підтримки, що ускладнить роботу з програмою користувачам;

- нестабільна робота програми;

- низька продуктивність програми;

- наявна підтримка AOL IM, але відсутня підтримка Skype і Viber.

- розшифровані листи залишаються незахищеними на клієнті;

- захист пошти працює тільки в режимі перехоплення, який вам швидко набридне, оскільки вікно захисту пошти буде з'являтися кожного разу для кожного нового сервера.

CyberSafe Top Secret

CyberSafe Top Secret (розробник CyberSafe) - комп'ютерна програма захисту інформації, яка використовує сучасні алгоритми шифрування (RSA, AES, BlowFish і т. Д.)., рис 2.4. Асортимент інструментів і функцій, що надаються CyberSafe Top Secret, корисний у всіх областях, які працюють з інформацією: захист конфіденційної інформації, захист електронної пошти, створення і перевірка цифрових підписів. CyberSafe Top Secret підтримує шифрування на основі інфраструктури відкритих ключів, і дозволяє шифрувати розділи жорстких дисків комп'ютера, створювати зашифровані віртуальні диски будь-якого розміру, а також приховувати логічні диски і зашифровані файли і папки на комп'ютері користувача. [14, 16].

CyberSafe Top Secret має наступні можливості:

- робота з віртуальними зашифрованими дисками;

- шифрування всього розділу;

- інтеграція з поштовими клієнтами;

- шифрування повідомлень електронної пошти;

- шифрування файлів;

- робота з ЕЦП підписами;
- ЕЦП перевірка;
- шифрування папок;
- робота з архівами, що саморозшифровуються;
- хмарне резервне копіювання;
- система довірених додатків;
- підтримка сертифікованого криптопровайдера;
- робота з власним сервером ключів;
-двофакторна аутентифікація;
- архівування розділів жорсткого диска [14].

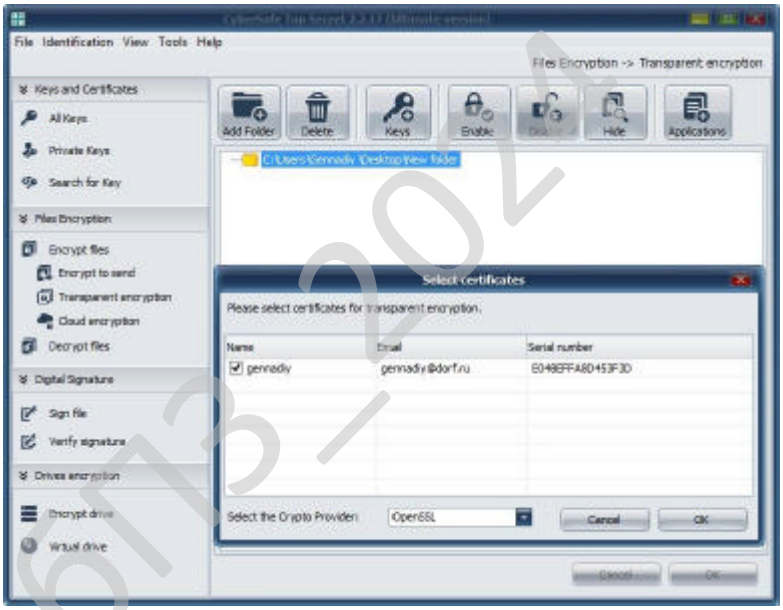

Рисунок 2.4 - CyberSafe Top Secret

Програма CyberSafe Top Secret містить засоби керування ключами і сертифікатами, а наявність в CyberSafe власного сервера ключів дозволяє користувачеві опублікувати на ньому свій відкритий ключ, а також отримати відкриті ключі інших співробітників компанії. Програма може використовуватися для шифрування окремих файлів. Що стосується алгоритмів шифрування, то програма CyberSafe Top Secret підтримує

алгоритми ГОСТ і сертифікований криптопровайдер КриптоПро, що дозволяє використовувати її в державних установах та банках.

Також програма може використовуватися для прозорого шифрування папки, що дозволяє її використовувати в якості заміни для EFS. А, враховуючи, що програма CyberSafe виявилася надійнішою і швидшою (в деяких випадках), ніж EFS, то використовувати її не тільки можна, але і потрібно.

Функціонал програми CyberSafe Top Secret нагадує функціонал програми PGP Desktop, програма також може використовуватися для шифрування повідомлень електронної пошти, а також для електронного підпису файлів і перевірки підпису.

Як і програма PGP Desktop, програма CyberSafe Top Secret вміє створювати віртуальні зашифровані диски, шифрувати повністю розділи жорсткого диска. Потрібно зазначити, що програма CyberSafe Top Secret вміє створювати віртуальні диски тільки фіксованого розміру, на відміну від програм Folder Lock і PGP Desktop. Однак цей недолік нейтралізується можливістю прозорого шифрування папки, а розмір папки обмежений тільки розміром вільного простору на жорсткому диску.

На відміну від програми PGP Desktop, програма CyberSafe Top Secret не вміє шифрувати системний жорсткий диск, вона обмежується лише шифруванням зовнішніх і внутрішніх несистемних дисків. Зате у CyberSafe Top Secret є можливість хмарного резервного копіювання, причому, на відміну від Folder Lock, дана можливість абсолютно безкоштовна, точніше функцію хмарного резервного копіювання можна налаштувати на будь-сервіс - як платний, так і безкоштовний.

Переваги програми CyberSafe Top Secret:

- підтримка алгоритмів шифрування ГОСТ та сертифікованого криптопровайдера КриптоПро, що дозволяє використовувати програму не

тільки приватним особам і комерційним організаціям, але і державним установам;

- підтримка прозорого шифрування папки, що дозволяє використовувати програму в якості заміни EFS. Враховуючи, що програма забезпечує кращий рівень продуктивності і безпеки, така заміна більш ніж виправдана;

- можливість підписання файлів електронного цифрового підписом і можливість перевірки підпису файлу;

- вбудований сервер ключів, що дозволяє публікувати ключі і отримувати доступ до інших ключів, які були опубліковані іншими співробітниками компанії;

- можливість створення віртуального зашифрованого диска і можливість шифрування всього розділу;

- можливість створення саморозшифровуючих архівів;

- можливість безкоштовного хмарного резервного копіювання, яка працює з будь-яким сервісом - як платним, так і безкоштовним;

- двофакторна аутентифікація користувача;

- система довірених додатків, що дозволяє надати доступ до зашифрованих файлів тільки певних програм;

- додаток CyberSafe підтримує набір інструкцій AES-NI, що позитивно позначається на продуктивності програми (цей факт буде продемонстровано далі);

- драйвер програми CyberSafe дозволяє працювати по мережі, що дає можливість організувати корпоративне шифрування.

Недоліки програми:

- в програмі виникають нелокалізовані повідомлення на кшталт «Password is weak»;

- програма не вміє шифрувати системний диск, але таке шифрування не завжди і не всім потрібно;

- наявні зависанням програми;

- зависока вартість програми.

AxCrypt

AxCrypt (розробник AxCrypt AB) - це утиліта з відкритим вихідним кодом, призначена для захисту даних методом шифрування, рис.2.5. Ця утиліта об'єднує в собі кодувальник і менеджер зашифрованих файлів. Робота першого грунтується на використанні криптографічного алгоритму AES-128 (AES-256 доступний у версії Premium). Все, що потрібно користувачеві для захисту власних документів, це увійти до облікового запису AxCrypt і вказати бажаний пароль. Подальший процес керування файлами здійснюється або за допомогою вбудованого менеджера, або шляхом використання відповідних пунктів в контекстному меню Windows [17,18,19,20].

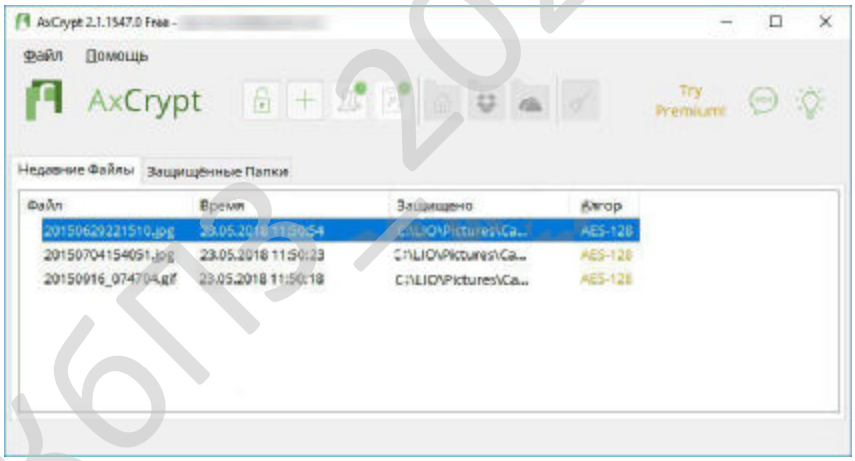

Рисунок 2.5 – AxCrypt

Основні можливості AxCrypt:

- режим шифрування алгоритмом шифрування AES-128 і SHA-1 із заміщенням або без заміщення оригіналу;

- можливість захисту ключовим файлом, що згенерував програмою;

- створення саморозпоковуючого шифрованого файлу (exe). В цьому випадку для розшифровки одержувачу не потрібна буде програма AxCrypt. Він повинен буде знати пароль і, якщо необхідно, мати файл ключа;

- можливість запуску зашифрованого файлу по паролю (файлу ключа) без дешифрування. При цьому оргінал файлу так і не з'явиться на носії.

- пакетне шифрування вмісту папок і підпапок.

- захист від відновлення при видаленні (при цьому місце, займане видаляється файлом, заповнюється шумом - випадковими кодами) [19].

Утиліта підтримує роботу з популярними хмарними сховищами, в першу чергу визначаючи, які з них встановлені на комп'ютері користувача, після чого створює там захищені каталоги. При цьому абсолютно не спостерігається будь-якого ускладнення процесу керування хмарними сервісами.

Інтерфейс AxCrypt дуже простий в освоєнні. Автоматичне декодування і відкриття файлів відбувається за допомогою подвійного кліку мишею. Єдиним недоліком оформлення програми можна назвати хоч і незвичайний, однак, досить непростий для читання шрифт.

Переваги AxCrypt:

- безкоштовне розповсюдження продукту;

- відкритий вихідний код;

- підтримка сучасних операційних систем сімейства Microsoft Windows;

- використання алгоритму шифрування AES в якості основи;

- зрозумілий інтерфейс, що не вимагає тривалого налаштування;

- швидке і зручне керування файлами за допомогою декількох кліків мишею;

- можливість керування з контекстного меню;

- підтримка файлів розміром більше чотирьох Гігабайт [17,18,19,20].

Недоліки AxCrypt

- незважаючи на захист файлів за допомогою криптографічних методів, зловмисник здатний просто видалити їх, отримавши доступ до ПК, і тим самим завдати шкоди користувачеві;

- шрифт, який важко читати;

- відсутність підтримки української мови (є російська);

- необхідність подання особистих даних, а саме адреси електронної пошти [17,18,19,20].

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

За наявності в якості засобу розробки для реалізації проекту побудови системи захисту даних шифрувальними методами обрано Delphi версії 7.

Загалом інтегроване середовище розробки Delphi фірми Embarcadero Technologies (в минулому розробка фірми Borland) працює в операційній системі Microsoft Windows. Мови інтерфейсу: англійська, французька, німецька, японська. Базовою мовою візуального середовища є Delphi (початкова назва перших версій Object Pascal). [21].

Основні характеристики Delphi - це комбінація кількох найважливіших технологій: високопродуктивний компілятор в машинний код: об'єктно-орієнтована модель - компонент; візуальної (а отже, і швидкісної) побудови додатків з програмних прототипів; Масштабуються кошти для побудови баз даних [22].

Розробка сучасних програмних продуктів практично неможлива без використання інструментальних засобів швидкої розробки програм RAD, яким оснащено Delphi [23].

У середовищі Delphi процес розробки програми складається з двох етапів, перший з яких орієнтований на роботу з візуальними конструкторами, а другий - на роботу з програмним кодом. Характерним при цьому є те, що в

процесі виконання другого етапу (безпосереднього програмування) завжди можна повернутися до першого з метою корекції проекту та наступним продовженням написання програмного коду. Ця особливість забезпечує гнучкість процесу розробки програми [23].

Візуальним відображенням працюючої програми, що написана у Delphi, є форма, яка розробляється в процесі проектування (спеціальне вікно форми створюється автоматично, Delphi автоматично створює оформлений за певними правилами спеціальний файл опису форми, що має розширення .DFM; також створюється і пов'язаний з формою та оформлений спеціальним чином файл, що називається модулем форми) [23].

Проектування форми програми полягає в наповненні її елементами керування, які називаються компонентами. Компоненти, які фактично є програмними заготівками для майбутньої програми, містять у собі як програмний код, так і дані, необхідні для його функціонування, входять у бібліотеку візуальних компонентів Delphi. Тому для проектування при виборі Delphi важливим є і наявність бібліотек з компонентами. Для проектування програмного додатку обмежимося використанням лише компонентів, що входять до стандартних бібліотек, які входять в комплект поставки Delphi .

Оскільки Delphi розраховано на програмування різних застосувань та надає велику кількість стандартних компонентів ([24, 25, 26 ,27, 28, 29, 30, 31]). а можливості, що надає Delphi версії 7 з пакету поставки, повністю відповідають вимогам проектування, то в якості засобу розробки і було обрано зазначене середовище програмування.

В якості додаткового засобу контролю текстового файлу, який буде підлягати шифруванню, передбачається використання текстових редакторів, що входять до наявного програмного забезпечення ПК - стандартного ПЗ для ОС сімейства Windows: WordPad або Nootepad/блокнот.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на бакалаврську роботу, реалізації підлягає програмне забезпечення, яке призначено для системи захисту даних шифрувальними методами. Підставою для розробки служить завдання на бакалаврську роботу, видане на кафедрі програмування комп'ютерних систем і мереж

У процесі розробки бакалаврської роботи необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів функціонування;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі;

д) розробити рекомендації щодо організаційних та методичних заходів, які забезпечать впровадження розробки в промислову експлуатацію та її подальшу успішну експлуатацію;

е) сформувати висновки про виконаний обсяг робіт та одержані результати.

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Якщо керуватися твердженням. що не існує єдиного універсального для всіх цілей та випадків алгоритму шифрування, то постає задача вибору з існуючих або розробки власного алгоритму шифрування.

Вибір за якістю наявних чи первинно розроблених алгоритмів носить більш суб’єктивний характер, однак при виборі можна керуватися такими загальними принципами:

- надійніші алгоритми шифрування зазвичай споживають більше ресурсів, ніж менш надійні;

- використання довгих ключів, як правило, дає більш надійні результати, ніж шифрування за допомогою коротких ключів;

- асиметричне шифрування повільніше симетричного;

- довгі складні паролі надійніше, ніж короткі паролі;

- симетричне шифрування зазвичай рекомендується використовувати тільки тоді, якщо ключ зберігається локально; асиметричне - якщо ключі повинні передаватися по каналу зв'язку;

- при необхідності шифрувати великі обсяги даних, шифруйте дані за допомогою симетричного ключа, а симетричний ключ - за допомогою асиметричного ключа;

- зашифровані дані не піддаються стисненню, але стислі дані можуть бути зашифровані [32].

Як уже зазначалось. алгоритми шифрування можна розділити на дві категорії: алгоритми симетричного та асиметричного шифрування. Проте переважна більшість сучасних алгоритмів шифрування працюють за дуже схожим принципом: над зашифрованим текстом виконується якесь

перетворення за участю ключа шифрування, яке повторюється певну кількість разів (раундів). При цьому, за видом повторюваного перетворення алгоритми шифрування прийнято ділити на декілька категорій. Тут також існують різні класифікації, одна з них:

- алгоритми на основі мережі Фейстеля (мережа Фейстеля - розбиття оброблюваного блоку даних на кілька субблоків, один з яких обробляється якоюсь функцією f () і накладається на один або кілька інших субблоків);

- алгоритми на основі узагальненої або розширеної мережею Фейстеля (це більш складна структура мережі Фейстеля відносно традиційної мережі Фейстеля, яка використовується значно рідше, приклад такої мережі Фейстеля - алгоритм RC6);

- алгоритми на основі підстановлювальння-перестановки мереж / SP-мережа - Substitution-permutation network (SP-мережі обробляють за один раунд цілком шифруючий блок; обробка даних зводиться, в основному, до замін, фрагмент вхідного значення замінюється іншим фрагментом відповідно до таблиці замін, яка може залежати від значення ключа, і перестановок, що залежать від ключа);

- алгоритми зі структурою "квадрат/ Square " (для структури "квадрат" характерно уявлення шифруючого блоку відсутня у вигляді двовимірного байтового масиву; криптографічні перетворення можуть виконуватися над окремими байтами масиву, а також над його рядками або стовпцями);

- алгоритми з нестандартною структурою, тобто ті алгоритми, які неможливо зарахувати до жодного з перерахованих типів (винахідливість може бути безмежна, тому класифікувати всі можливі варіанти алгоритмів шифрування представляється складним, як приклад алгоритму з нестандартною структурою можна привести унікальний за своєю структурою алгоритм FROG, в кожному раунді якого за досить складним правилом виконується модифікація двох байт шифрованих даних) [33].

Суворого розмежування між описаними структурами не має, тому досить часто зустрічаються алгоритми, що можуть відповідати різним типам структур (наприклад - алгоритм CAST-256 відноситься його автором до SP-мережі, а багатьма експертами називається розширеною мережею Фейстеля; інший приклад - алгоритм HPC, званий його автором мережею Фейстеля, але відноситься експертами до алгоритмів з нестандартною структурою) [33].

Отже, переглянувши загальні принципи побудови та проаналізувавши наявні алгоритми шифрування для системи, що проектується, обираємо симетричне шифрування.

Реалізації симетричного шифрування зазвичай діляться на дві великі категорії: блокові шифри і потокові шифри. Блокові шифри шифрують вихідне повідомлення невеликими блоками, потокові шифри шифрують повідомлення, цілком зіставляючи його з якоюсь гамою, тобто, важливо також розташування символу повідомлення у вихідному повідомленні [32].

Спираючись на зазначене, в загальних рисах, пропонується побудувати алгоритм шифрування на змішуванні послідовності, що шифрується, з псевдовипадковою послідовністю, яка генерується на базі секретного ключа відповідно до закону F(k).

Відповідно до процесу шифрування, процес розшифровування (дешифрування) представлятиме собою виділення з адитивної суміші інформативної послідовності шляхом видалення псевдовипадкової послідовності, яка генерується на базі все того ж секретного ключа відповідно до того ж закону F(k).

Задумуючи реалізацію алгоритму, в найзагальніших рисах було розуміння, що псевдовипадкова послідовність повинна бути лише одна й та, тому й генерується вона на базі все того ж секретного ключа відповідно одного й до того ж закону F(k).

Для генерації псевдовипадкової послідовності пропонується функція F(k), що являє собою генерацію поточного коду символу генеруючої

псевдовипадкової послідовності в якості добутку кодів символів збідльшених на одиницю останнього сгенерованого та ковзного з першого цієї самої псевдовипадкової послідовності, при умові, що перші символи псевдовипадкової послідовності являють собою секретній ключ, що використовується для генерації псевдовипадкової послідовності.

Використання такої функції F(k) для генерації псевдовипадкової послідовності приводить до досить великого періоду повторення символів послідовності, що генерується (в загальному випадку період повторення символів псевдовипадкової послідовності на пряму залежить від довжини секретного ключа).

Запропонована функція F(k) для генерації псевдовипадкової послідовності надає можливість для проведення досить ефективного (з точки зору надійності до "взлому") шифрування

Системи захисту даних шифрувальними методами призначена для шифрування деякої послідовності символів (файлу) для зберігання або передачі з подальшим відтворенням початкового представлення (файлу). Виходячи з призначення, в системі має бути реалізований наступний функціонал:

- запуск програми та забезпечення її працездатності при встановленому факту на правове використання програного продукту;

- блокування працездатності при виявленні факту неправомірного використання програмного продукту;

- завершення роботи програми;

- вибір режиму роботи програми: шифрування, дешифрування, отримання довідкової інформації, завершення;роботи;

- для режиму шифрування наступні операції: вибір файлу для шифрування, введення ключа шифрування (пароля). введення назви файлу для збереження зашифрованих даних, виведення інформації про хід та результати відпрацювань операції шифрування;

- для режиму дешифрування наступні операції: вибір файлу для розшифрування, введення ключа шифрування (пароля). введення назви файлу для збереження розшифрованих даних, виведення інформації про хід та результати відпрацювань операції дешифрування;

- робота додатку має супроводжуватись індикацією її відпрацювань при виконанні .

Порядок виконання процесів при режимі шифрування:

- вибір файлу, що підлягає шифруванню;

- введення секретного ключа - пароля шифрування;

- введення назви файлу для збереження результату шифрування;

- генерація псевдовипадкової послідовності;

- безпосереднє шифрування шляхом змішування з змішування інформаційної послідовності з псевдовипадковою;

- збереження зашифрованого файлу.

Порядок виконання процесів при режимі дешифрування:

- вибір файлу, що підлягає дешифруванню;

- введення секретного ключа - пароля шифрування;

- введення назви файлу для збереження результату відтворення інформації;

- генерація псевдовипадкової послідовності;

- безпосереднє дешифрування - виокремлення інформаційної послідовності шляхом видалення псевдовипадкової послідовності з адитивної суміші інформаційної послідовності та псевдовипадковою;

- збереження відтвореного файлу.

3.2 Розробка структурної схеми

Проектування системи (програмного забезпечення) проводиться з розробки структурних одиниць самої системи (програми) та описання зв’язків між різними

структурними одиницями (коли структурні одиниці розглядаються в якості окремих частини - блоків, що виконують різні функції) та проходженням інформаційних потоків через них [34]. Оскільки, для реалізації проекту в якості інструменту вибрано Delphi, то за такої реалізації, програма складатиметься з файлу проекту (файл з розширенням DPR) і одного чи кількох модулів (файли з розширенням PAS). Структурно модулі - це окремі програмні одиниці, що реалізують окремі частини програми. [29, 30].

При розробці структури програми накладемо умову: модуль - одна структурна одиниця. Розглянемо структурну програму через призму виконання її з боку функції. Структурна схема системи зображена на рисунку 3.1.

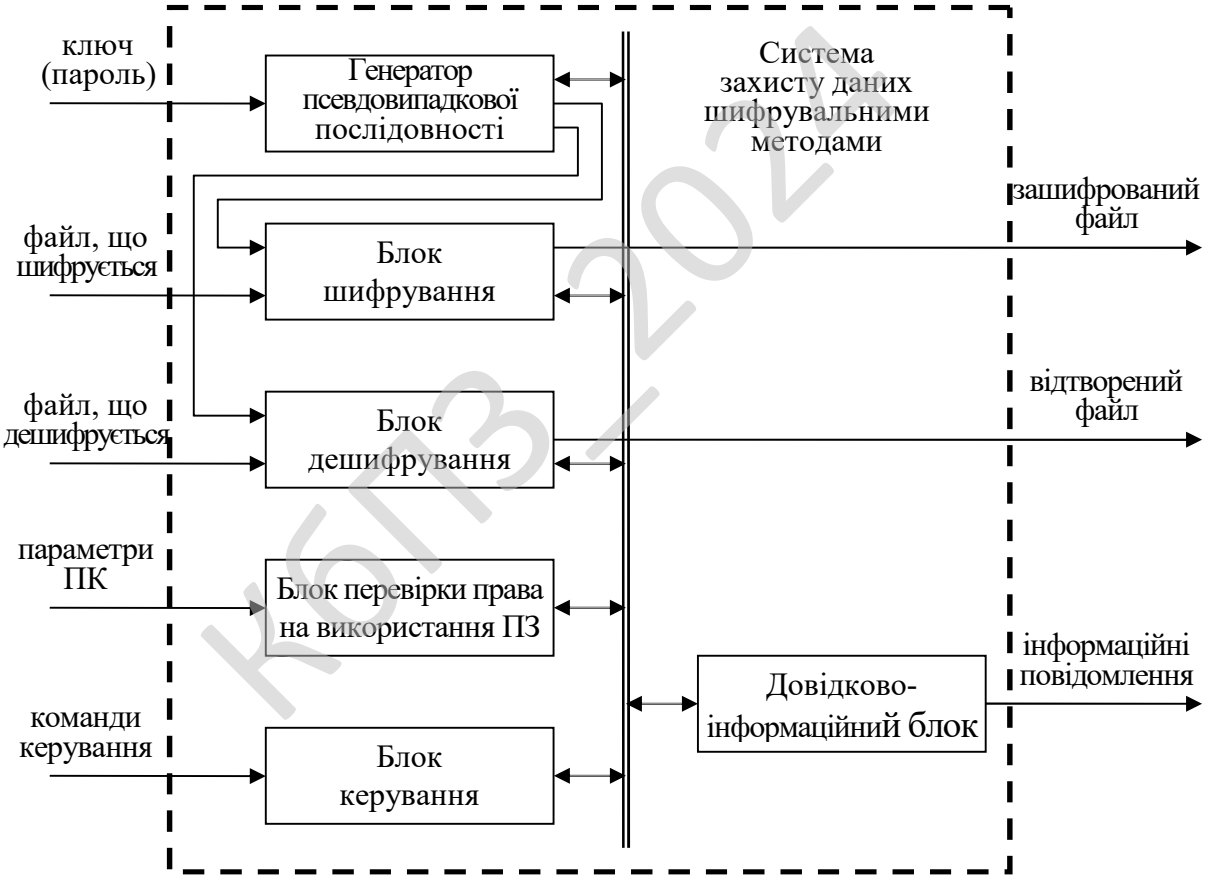

Рисунок 3.1 - Структурна схема

Зі схеми можна побачити, що працездатність забезпечується вхідними інформаційними потоками:

- ключ (пароль) - текстова послідовність, з якої генерується

псевдовипадкова послідовність для режимів шифрування та дешифрування;

- файл, що шифрується - вхідні дані для режиму шифрування;

- файл, що дешифрується - вхідні дані для режиму дешифрування;

- параметри ПК - данні, що зчитуються з ПК та його системного реєстру (серійні номера пристроїв, системна дата, ...) для реалізації захисту програмного продукту від несанкціонованого копіювання;

- команди керування - команди для керування процесами роботи системи захисту даних шифрувальними методами.

Також, із схеми можна побачити, що генеруються наступні інформаційні потоки:

- зашифрований файл - вихідні дані для режиму шифрування;

- відтворений файл - вихідні дані для режиму дешифрування;

- інформаційні повідомлення - вихідна інформація (повідомлення системи захисту даних шифрувальними методами) про стан та результати відпрацювання програмного додатку.

На схемі також показані внутрішні інформаційні зв’язки:

- зв’язок між генератором псевдовипадкової послідовності і блоками шифрування і дешифрування, де по запиту до блоків надходить потік псевдовипадкової послідовності;

- зв’язок між блокам керування та всіма наявними блоками для керування роботою в цілому, а саме: режимами шифрування та дешифрування, організацією захисту програмного забезпечення та наданням інформаційно-довідкових повідомлень про стан відпрацювання та іншої інформації.

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи зображена на рисунку 3.2.

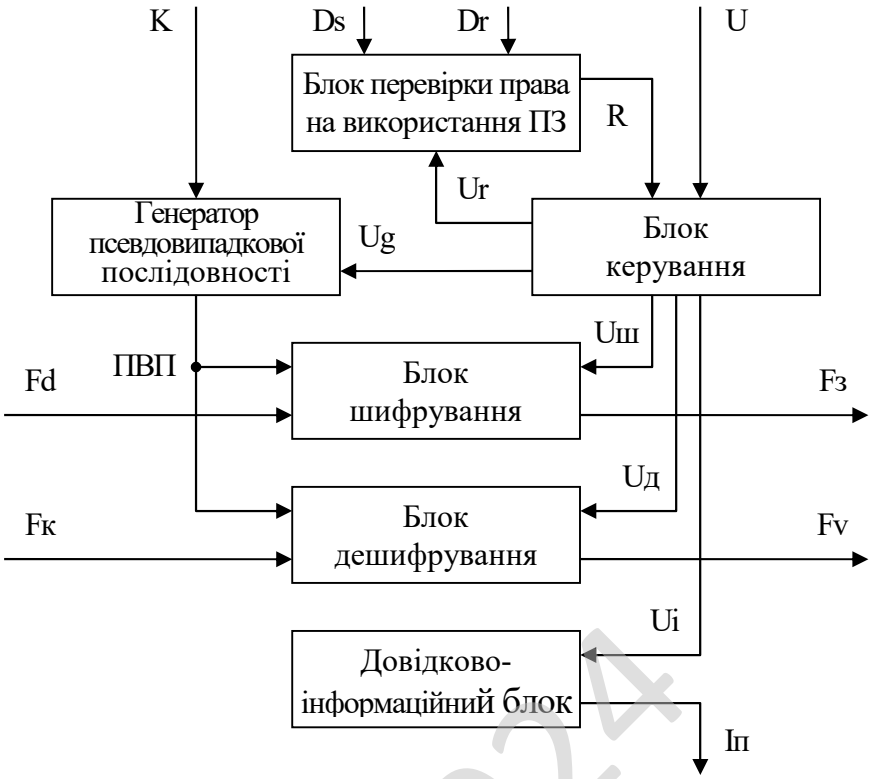

Рисунок 3.2 - Функціональна схема

З рисунка видно, що роботу розробленої системи можна представити функціонуванням сукупності частин: блоку перевірки права на використання ПО, генератора псевдовипадкової послідовності, блоку керування; блоку шифрування; блоку дешифрування; довідково-інформаційного блоку.

Розглянемо їх роботу. Незалежно від режиму роботи блок перевірки права на використання ПЗ в заданості від Ds (параметрів ПК: серійного номера пристроїв, системної дати, ...) та Dr (даних про параметри ПК, що занесено до системного реєстру при встановленні додатку) шляхом їх порівняння визначає дозвіл R (право на використання ПЗ)та подає його на блок керування.

Блок керування в залежності від U (команди керування) та наявності дозволу R (дозволу на використання ПЗ, що зчитується по запиту Ur від самого блоку керування) формує керування:

- керування Ug (сигнал на генерацію псевдовипадкової послідовності) та надсилає його до генератора псевдовипадкової послідовності;

- керування Uш (керування на відпрацювання блоку шифрування) та надсилає його до блоку шифрування;

- керування Uд (керування на відпрацювання блоку дешифрування) та надсилає його до блоку дешифрування;

- керування Ui (керування на вивід довідково-інформаційних повідомлень) та надсилає його до довідково-інформаційного блоку.

При появі керування Ug генератор псевдовипадкової послідовності по ключу K формує псевдовипадкову послідовність ПВП та надсилає її до блоків шифрування та дешифрування.

При появі керування Uш блок шифрування за визначеним алгоритмом шифрування, використовуючи ПВП з генератора псевдовипадкової послідовності, перетворює файл Fd в файл Fш.

При появі керування Uд блок дешифрування за визначеним алгоритмом дешифрування, використовуючи ПВП з генератора псевдовипадкової послідовності, перетворює файл Fk в файл Fv.

При появі керування Ui довідково-інформаційний блок видає відповідні довідково-інформаційні повідомлення Iп.

3.4 Розробка діаграми процесів

Оскільки для опису архітектури системи захисту даних шифрувальними методами можна скористатися одним із п’яти видів представлень, кожна з яких є одна з можливих проекцій організації і структури системи і відповідає окремому аспекту її функціонування (вид з погляду прецедентів використання, вид з погляду проектування, вид з погляду процесів, вид з погляду реалізації, вид з погляду розгортання

системи,) то для опису механізмів синхронізації взаємодій, станів і дій скористаємося діаграмою процесів [35].

Діаграма процесів розробленої системи зображена на рисунку 3.3.

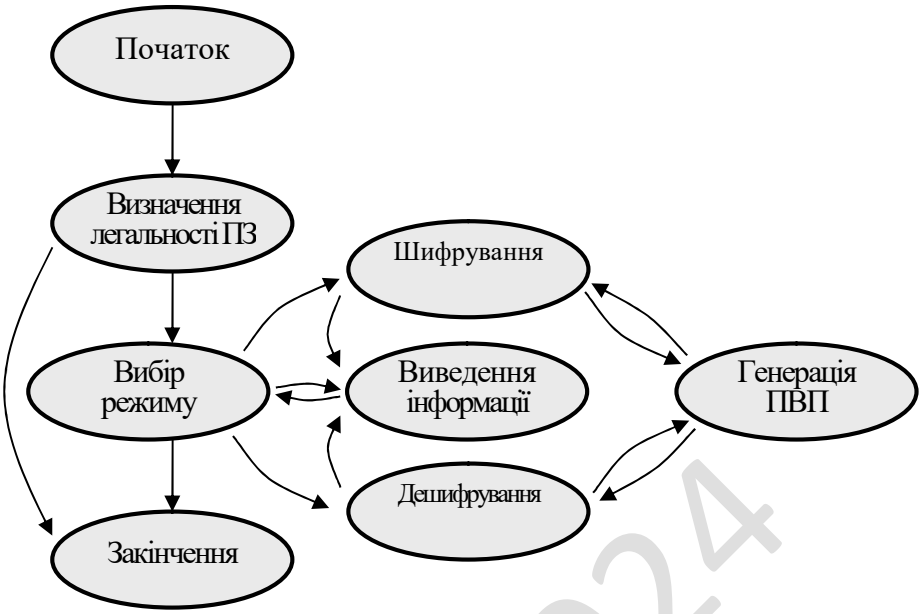

Рисунок 3.3 - Діаграма процесів

Діаграма процесів представлена у вигляді напрямленого графа, що описує переходи від одного процесу до іншого. Проаналізувавши діаграму процесів, можна визначити безпосередньо послідовність дій системи, що розробляється.

Так, процес роботи програми безальтернативно починається з визначення легальності використання додатку (програмного забезпечення, що розробляється), і у випадку, коли виявлено, що програмне забезпечення незаконно використовується, - робота програми припиняється (таким чином не надається можливість для використання нелегального ПЗ). У випадку проходження перевірки на легальність встановленого ПЗ - програма переходить до вибору режимів роботи. При виборі режиму шифрування - програма переходить до шифрування. потім до генерації псевдовипадкової послідовності і далі назад до шифрування, а через введення інформації знову до вибору режиму. При виборі режиму дешифрування - програма переходить

до дешифрування, потім також до генерації псевдовипадкової послідовності і далі назад до дешифрування, а через введення інформації - знову до вибору режиму. У випадку, коли безпосередньо обирається виведення інформації, то після опрацювання теж проходить повернення до вибору режиму. Завершується роботи також проходить з процесі вибору режиму.

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів, перейдемо до опису блок-схем основної програми та підпрограм, які використовуються для реалізації системи.

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ

4.1 Розробка блок-схем та опис алгоритмів функціонування системи

Розглянемо алгоритм роботи програми. Блок-схема алгоритму роботи основної програми зображена на рисунках 4.1, 4.2, 4.3, 4.4, 4.5.

Рисунок 4.1 - Блок-схема роботи основної програми (початкова частина алгоритму).

З представленої блок-схеми роботи програми видно, що спочатку відбувається зчитування параметрів ПК та системна дата. а потім проходить зчитування аналогічних параметрів, що записано до реєстру при встановленні програмного забезпечення.

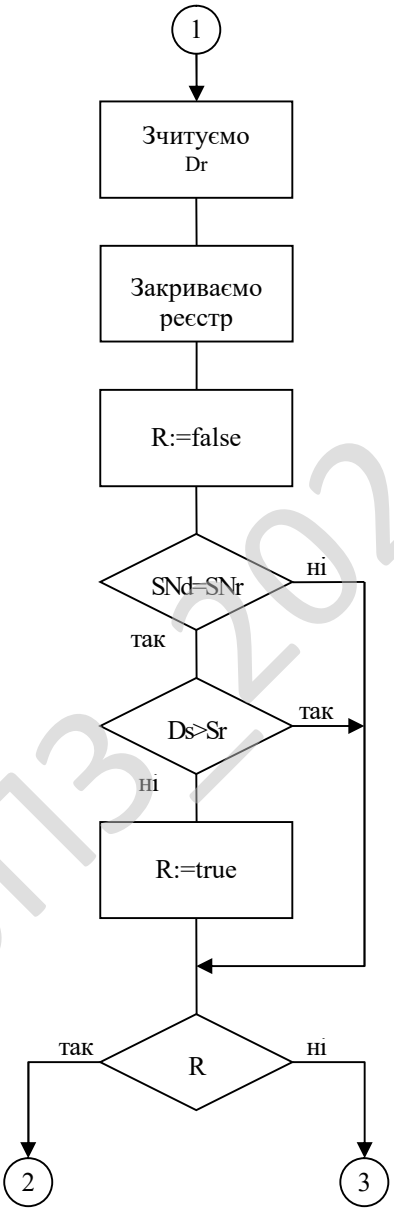

Рисунок 4.2 - Блок-схема роботи основної програми (частина алгоритму, де визначається правомірність використання ПЗ)

За цими даними проходить перевірка на легальність використання додатку, що проектувався, та формується дозвіл R на його використання.

Системна дата використовується для обмеження часу використання додатку. Вказані параметри зберігаються в розділі системного реєстру HKEY_LOCAL_MACHINE

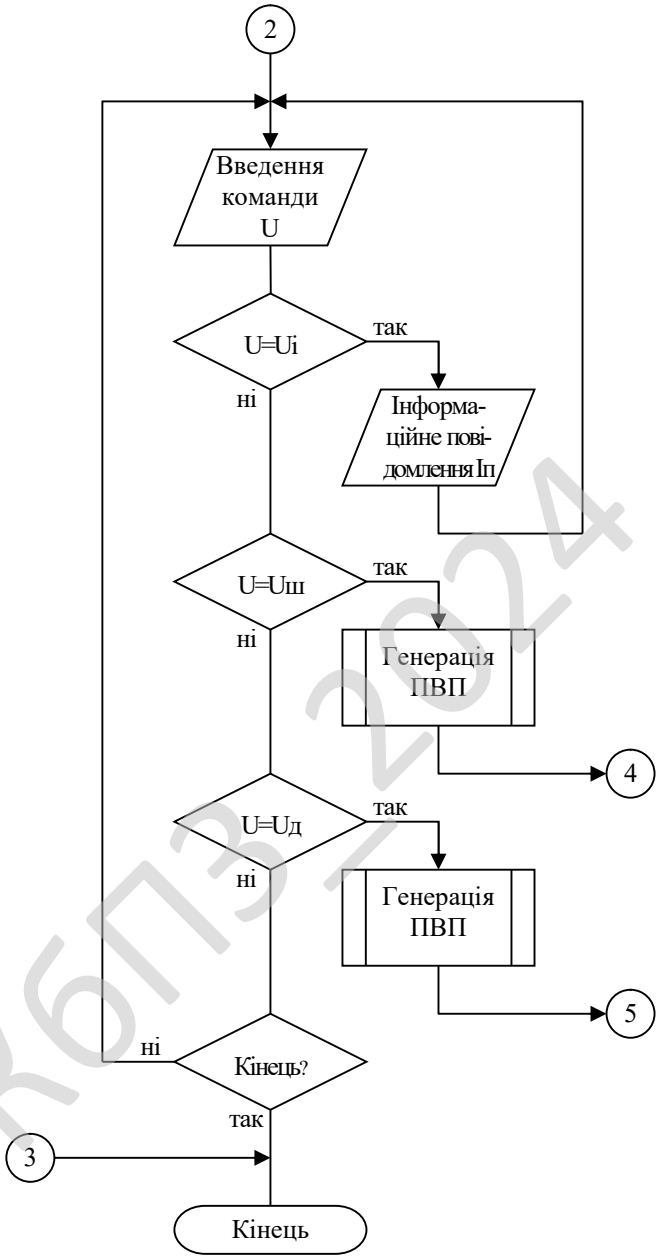

Рисунок 4.3 - Блок-схема роботи основної програми (частина алгоритму, де описується блок управління)

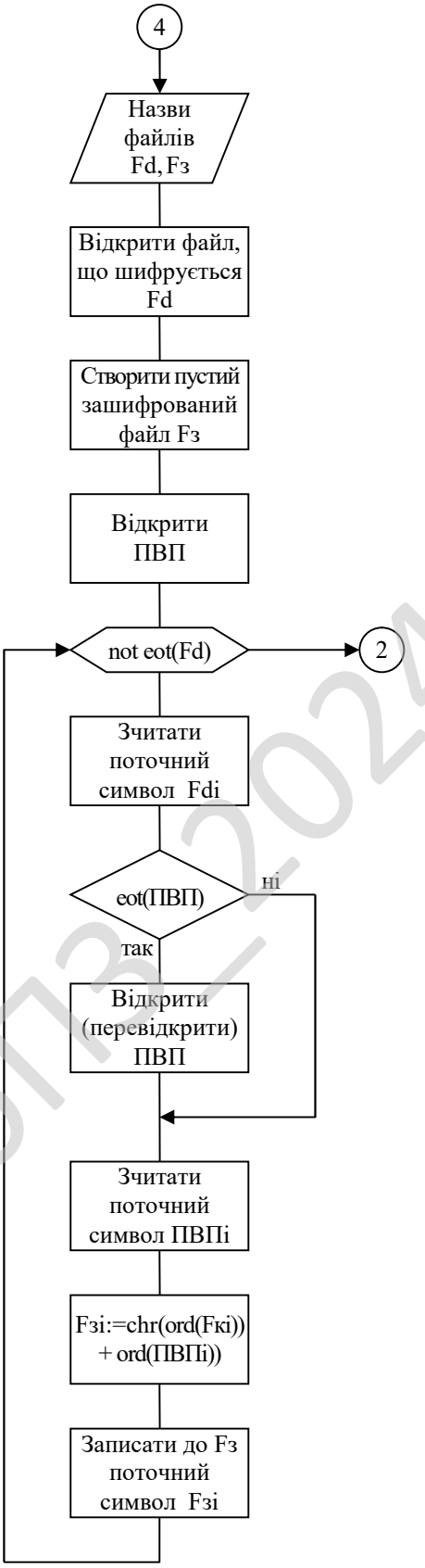

Рисунок 4.4 - Блок-схема роботи основної програми (опис шифрування)

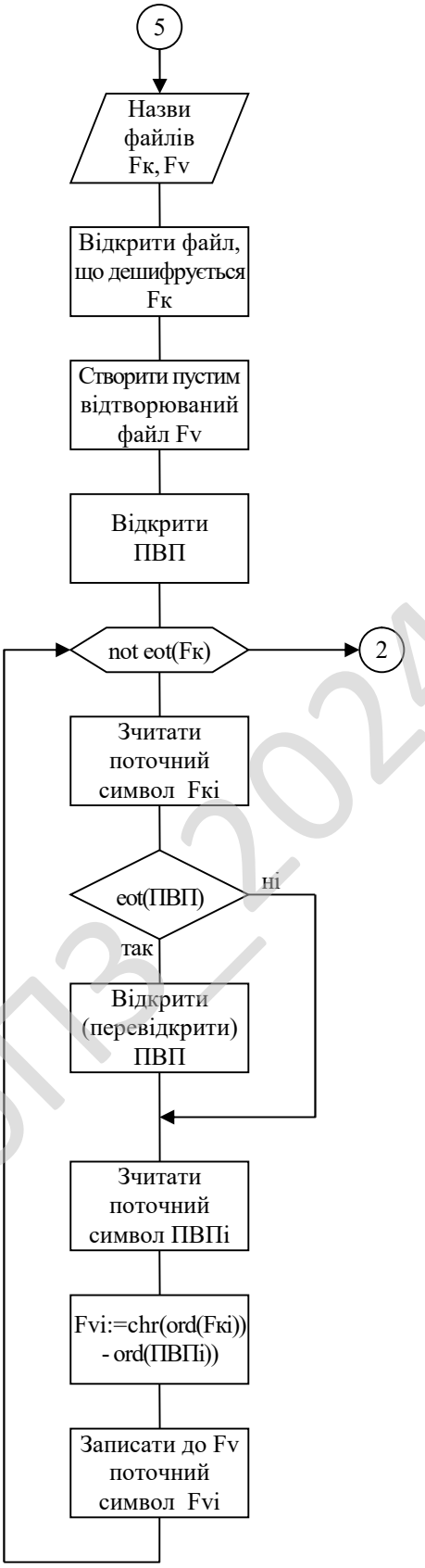

Рисунок 4.5 - Блок-схема роботи основної програми (опис дешифрування)

У випадку, коли легальність використання програмного забезпечення не встановлено, передбачено просто припинення виконання програми.

У випадку, коли правомірності використання програмного забезпечення, проходить вибір режиму роботи додатку.

У випадку вибору режиму шифрування проходить наступна послідовність виконання:

- генерується псевдовипадкова послідовність;

- вибирається файл для шифрування;

- задається назва файлу для збереження зашифрованого результуючого відпрацювання шифрування;

- відкривається файл, що шифрується на зчитування;

- створюється попередньо пустим зашифрований файлу для вмісту результату шифрування;

- відкривається на зчитування згенерована псевдовипадкова послідовність;

- далі до моменту поки не зафіксовано кінець файлу, що шифрується, проводиться наступна послідовність дій, що циклічно повторюється:

- зчитується поточний символ файлу, що шифрується;

- перевіряється на закінчення псевдовипадкова послідовність, і якщо її кінець досягнуто, то вона перевідкривається;

- зчитується поточний символ псевдовипадкової послідовності;

- шифрується поточний символ зашифрування - розраховується шляхом визначення його коду, як суми кодів символів: символу, що зашифровується та символу псевдовипадкової послідовності;

- і в кінці циклічноповторювального процесу. визначений поточний символ записується до результуючого зашифрованого файлу;

- по завершенні шифрування відбувається повернення на вибір режиму.

У випадку вибору режим дешифрування проходить наступна послідовність виконання:

- генерується псевдовипадкова послідовність;

- вибирається файл для дешифрування;

- задається назва файлу для збереження відтвореного результуючого відпрацювання дешифрування;

- відкривається файл, що дешифрується на зчитування;

- створюється попередньо пустим файлом. що відтворюється. для вмісту результату дешифрування;

- відкривається на зчитування згенерована псевдовипадкова послідовність;

- далі до моменту поки не зафіксовано кінець файлу, що дешифрується, проводиться наступна послідовність дій, що циклічно повторюється:

- зчитується поточний символ файлу, що дешифрується;

- перевіряється на закінчення псевдовипадкова послідовність, і якщо її кінець досягнуто, то вона перевідкривається;

- зчитується поточний символ псевдовипадкової послідовності;

- дешифрується поточний символ - розраховується шляхом визначення його коду, як рзниці кодів символів: символу, що дешифрується, та символу псевдовипадкової послідовності;

- і в кінці циклічноповторювального процесу. визначений поточний символ записується до результуючого файлу, що відтворюється;

- по завершенні дешифрування відбувається повернення на вибір режиму.

Наявний режим виведення довідково-інформаційних повідомлень і при його виборі виводяться повідомлення, а по закінчення відпрацювання режиму відбувається повернення на вибір режиму.

Завершення роботи додатку здійснюється, як вибір режиму.

Генерацію псевдо випадкової послідовності представлено окремим алгоритмом, та оформлено, як підпрограму, блок-схема алгоритму зображена на рисунках 4.6.

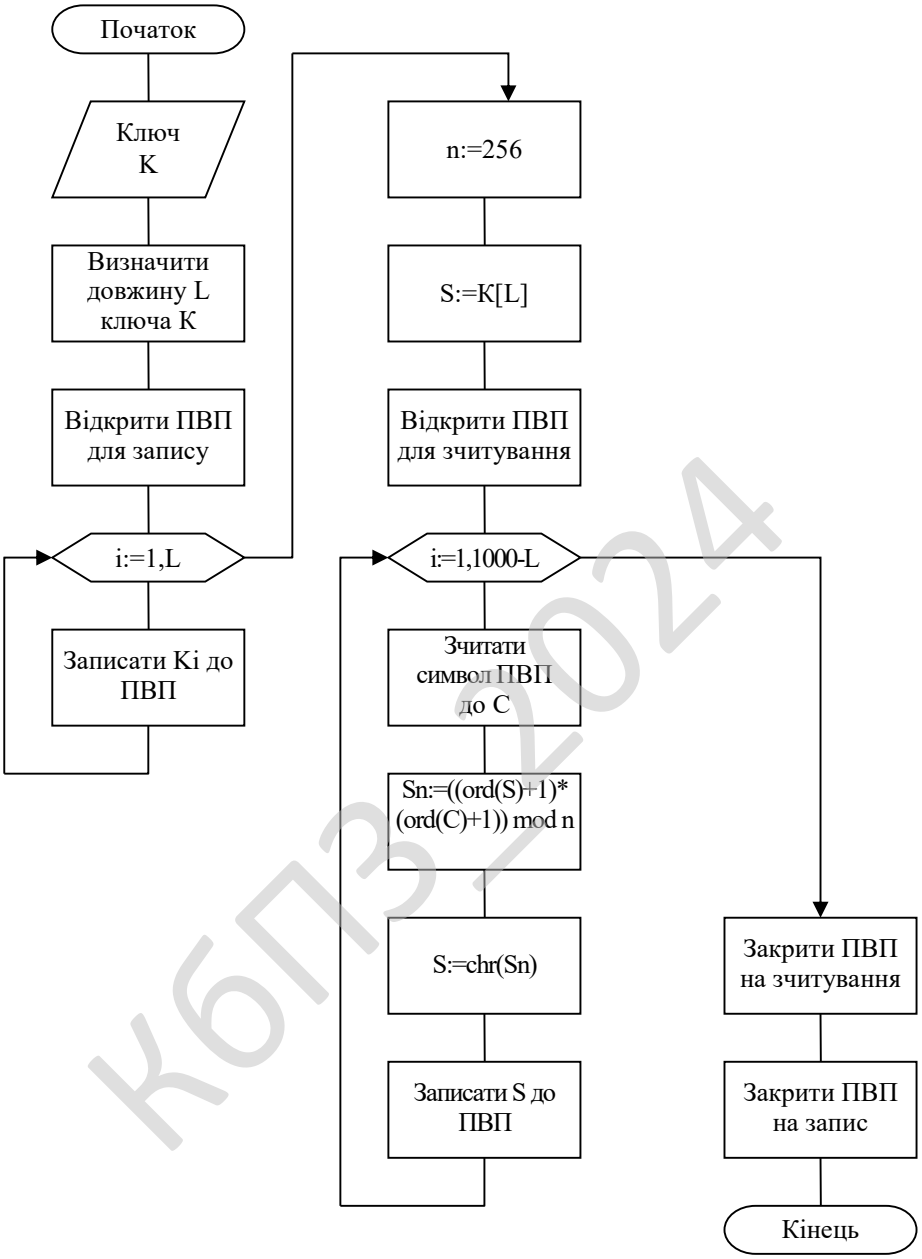

Рисунок 4.6 - Блок-схема роботи підпрограми генерації псевдовипадкової послідовності

З представленої блок-схеми роботи підпрограми генерації псевдовипадкової послідовності видно, що спочатку відбувається введення ключа (пароля - послідовності символів), потім визначається довжина L цієї

послідовності, відкривається (спочатку створюється) для запису файл псевдовипадкової послідовності, до якого посимвольно заноситься введений ключ, задається об'єм n алфавіту псевдовипадкової множини, запам'ятовується як змінна S останній символ ключа. Відкривається файл псевдовипадкової послідовності на зчитування з початку файла, далі послідовно 1000-L разів перечитується посимвольно файл псевдовипадкової послідовності й формується код його кінцевого символу, як залишок від ділення на n добутку кодів прочитаного символу та символу, який передує останньому (що знаходився до цього) з символів S та C без 0-го розряду, й на останок закривається файл псевдовипадкової послідовності на зчитування та для запису.

4.2 Захист розробленого програмного забезпечення

Для захисту програмного забезпечення від несанкціонованого розповсюдження існує ряд загальнорозповсюджених способів, один з яких - прив'язка до параметрів комп'ютера [36].

В більшості випадків, організовується прив'язка до жорсткого диска. Технічно прив'язка виконується до серійного номера вінчестера. В Delphi безпосередньо серійний номер можна визначати [37].

```
function TProtect.GetHDDSerial:dword;
var
a,b,SerialNum:dword;
VolumeName : array [0..255] of char;
begin
Result := 0;
if GetVolumeInformation(PChar('c:\'), VolumeName, SizeOf(VolumeName),
@SerialNum, a, b, nil, 0)
then Result := SerialNum;
end;
```

Також в Delphi можна організувати прив'язку до системної плати її серійного номера, що можна організувати за допомогою функції [36]:

```
function TProtect.GetSerialMotherBoard: String;
var  a, b, c, d: LongWord;
begin
asm
push EAX
push EBX
```

```
        push ECX
        push EDX
        mov eax, 1
        db $0F, $A2
        mov a, EAX
        mov b, EBX
        mov c, ECX
        mov d, EDX
        pop EDX
        pop ECX
        pop EBX
        pop EAX
        end;
        result:=inttohex(a, 8)+
                inttohex(b, 8)+
                inttohex(c, 8)+
                inttohex(d, 8)
        end;
```

Правомірність використання пропонується визначати шляхом порівняння ID ПК синтезованого за серійним номером пристрою (носія, системної плати. ...), який має бути внесений при встановлені програного забезпечення до системного реєстру. Для читання ID ПК з системного реєстру може бути використано функцію [36]:

```
        function GetRegID:dword;
        uses Registry;
        var reg:TRegistry;
        begin
        reg:=TRegistry.Create;
        reg.RootKey:=HKEY_LOCAL_MACHINE;
        reg.OpenKey('HardWareSN}', false);
        Result:=RegKeyGetDw(HKEY_LOCAL_MACHINE;'HardWareSN');
        reg.CloseKey;
        reg.Free;
        end;
```

Для проведення активації програмного забезпечення при його встановленні на визначеному обладнанні (прив’язки до нього) може використовуватися процедура для запису до системного реєстру:

```
        procedure SetRegID;
        uses  Registry;
        var reg:TRegistry;
        SerialNum,a,b:dword;
        VolumeName:array [0..255] of char;
        begin
        if GetVolumeInformation(PChar('c:\'), VolumeName, SizeOf(VolumeName),
@SerialNum, a, b, nil, 0)
        then
        begin
        reg:=TRegistry.Create;
        reg.RootKey:=HKEY_LOCAL_MACHINE;
        reg.OpenKey('HardWareSN', false);
        RegKeySettDw(HKEY_LOCAL_MACHINE,'HardWareSN',SerialNum);
        reg.CloseKey;
```

```
reg.Free;
end;
end;
```

Крім захисту програми від санкціонованого копіювання доцільно використати часове обмеження на функціонування програмного забезпечення шляхом встановлення кінцевої дати на використання, яку теж по аналогії з ID необхідно занести до системного реєстру. Термін використання визначатиметься шляхом порівняння системної дати і дати кінцевого використання, занесеної до реєстру (таке обмеження обумовлене економічним міркуванням - необхідністю отримувати прибуток за розробку протягом терміну її використання, для організації подовження комерційних угод на використання).

5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На етапі введення системи до експлуатації проводяться: випробування окремих компонентів та ІС в цілому, дослідна експлуатація системи в реальних умовах, переговори щодо результатів виконання проекту, можливих змін та нових контрактів. Ключові роботи цього етапу: комплексні випробування; підготовка кадрів для експлуатації системи; підготовка робочої документації, здавання системи замовнику і введення її в експлуатацію; супроводження, підтримка, сервісне обслуговування; оцінка результатів проекту та підготовка підсумкових звітів; вирішення конфліктних ситуацій і закриття робіт за проектом; накопичення дослідних даних для подальших проектів, аналіз досвіду, стану, визначення напрямів розвитку [38].

Враховуючи зазначене, виділимо для розгляду введення в експлуатацію програмного забезпечення ряд завдань. що підлягають освітленню. Це: підготовка користувачів програмного забезпечення, встановлення програмного забезпечення, його запуск та налагодження, контроль працездатності та супровід програмного забезпечення.

Експлуатація програмного забезпечення, що розроблялося., загалом не вимагає ніяких специфічних навиків від користувачів. Програма оснащена простим і інтуїтивно зрозумілим інтерфейсом, і користувачі, що мають загальні навики роботи з комп'ютерними програмами, без будь-яких труднощів можуть відразу в ній працювати. То ж перед введенням в експлуатацію системи немає необхідності у проведенні спеціальних навчань персоналу, а тому вони і не проводяться.

Запуск та налагодження програми. Програма призначена для експлуатації в середовищі сімейства ОС Windows., програмним файлом є Naum.exe, встановлення та запуск якого проводиться стандартними

способами для даної операційної системи. Відмінністю для конкретної реалізації є встановлення засобів захисту від несанкціонованого копіювання - програмний додаток оснащено вбудованим генератором ID ПК на основі серійного номера одного з пристроїв ПК (а саме HDD), рис.5.1. Обов'язковою умовою для працездатності системи є наявний доступ до системного реєстру, в якому зберігаються або заносяться дані. В даному конкретному випадку до системного реєстру в розділ HKEY_LOCAL_MACHINE блок параметрів HardWare заноситься параметр sn, значення якого має співпадати з серійним номером жорсткого диска, порівнюючи згадані величини програма визначає правомірність застосування програмного забезпечення. Для комерційного ж використання, передбачається встановлення ID ПК виокремленим програмним модулем.

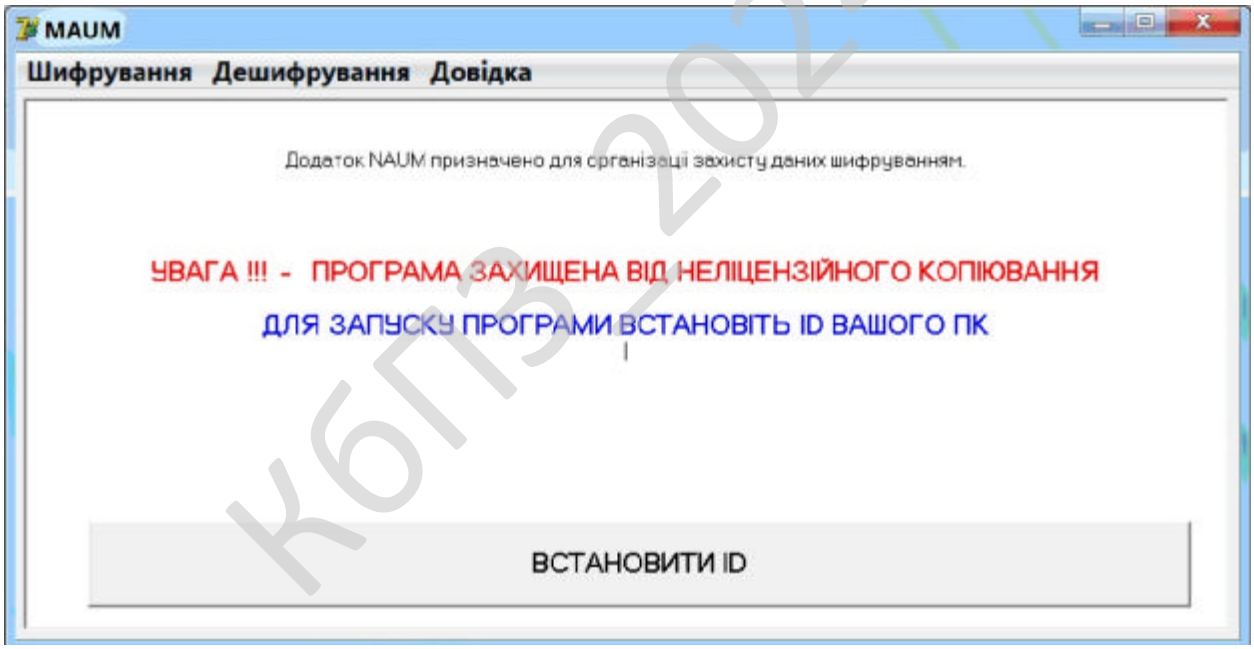

Рисунок 5.1 - Скріншот роботи додатку, встановлення ID на ПК.

Після встановлення ID на ПК необхідно провести перезапуск програмного файлу Naum.exe, що приведе до розкриття робочого вікна додатку, рис.5.2

Входження до режиму шифрування можливе вибором відповідної кнопки (рис.5.3) або шляхом введення одного з параметрів шифрування через меню (відповідний розділ в меню додатку)

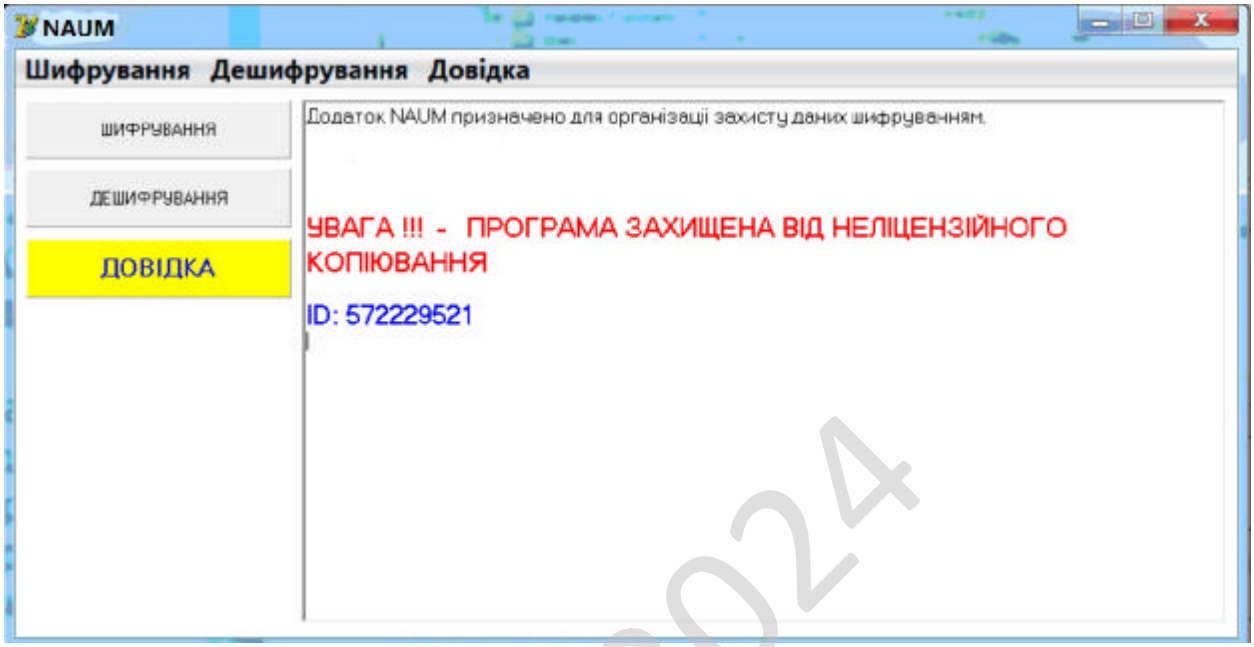

Рисунок 5.2 - Скріншот роботи додатку, стартова довідкова сторінка.

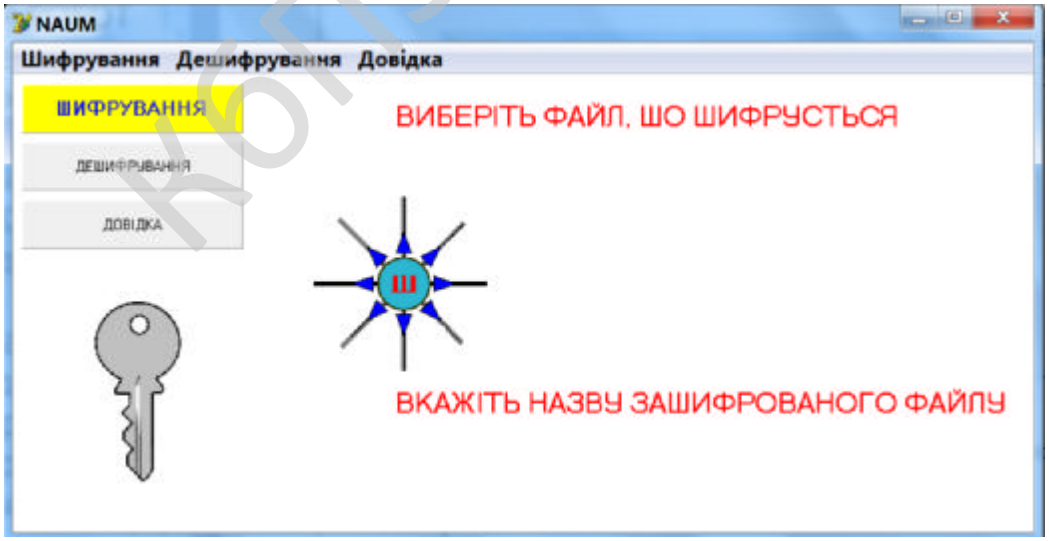

Рисунок 5.3 - Скріншот роботи додатку, режим шифрування.

Для вибору файлу, що шифрується, можна скористатися меню додатку або безпосередньо перейди до вибору по посиланню "ВИБЕРІТЬ ФАЙЛ, ЩО ШИФРУЄТЬСЯ", вибір проводиться через діалогове вікно OpenDialog , рис. 5.4. У випадку, коли зроблено помилковий вибір, наявна можливість змінити вибір файлу , що шифрується (повторний вибір).

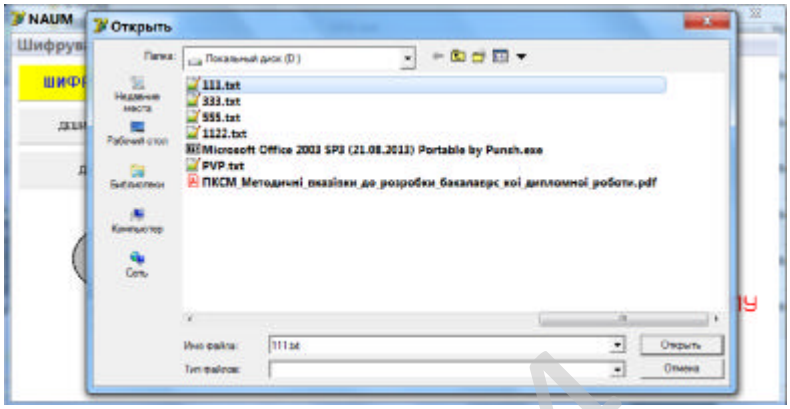

Рисунок 5.4 - Скріншот роботи додатку, вибір файлу, що шифрується.

Для введення назви файлу зашифрованого файлу також можна скористатися меню додатку або безпосередньо перейди до вибору по посиланню "ВКАЖІТЬ НАЗВУ ЗАШИФРОВАНОГО ФУЙЛУ ", вибір проводиться через діалогове вікно SaveDialog , рис. 5.5. У випадку, коли зроблено помилковий вибір, наявна можливість змінити вибір файлу , що шифрується (повторний вибір).

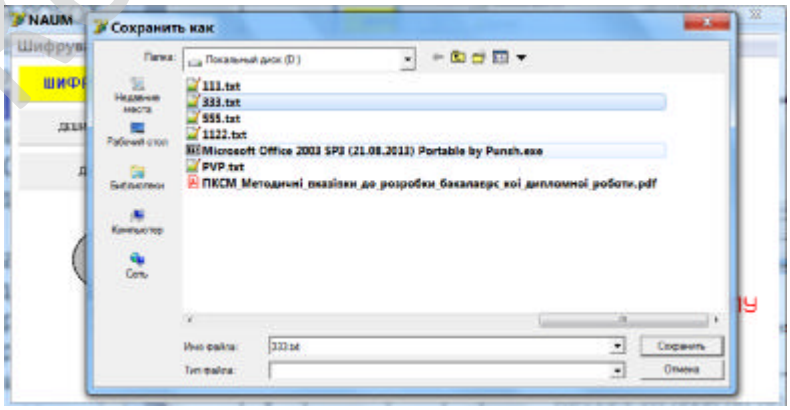

Рисунок 5.5 - Скріншот роботи додатку, введення назви зашифрованого файлу.

Для введення ключа (пароля) шифрування можна скористатися меню додатку або безпосередньо перейди до вікна введення натиснувши на кнопку в формі ключа (повторний натиск кнопки ховає вікно введення), рис. 5.6.

Саме шифрування запускається з меню додатку або безпосередньо натиском на кнопку у вигляді ручки сейфу з буквою "Ш". Вдале виконання супроводжується повідомленням (рис. 5.7). При виявлені відсутності якогось з необхідних параметрів для проведення шифрування, приведе до запиту на його введення

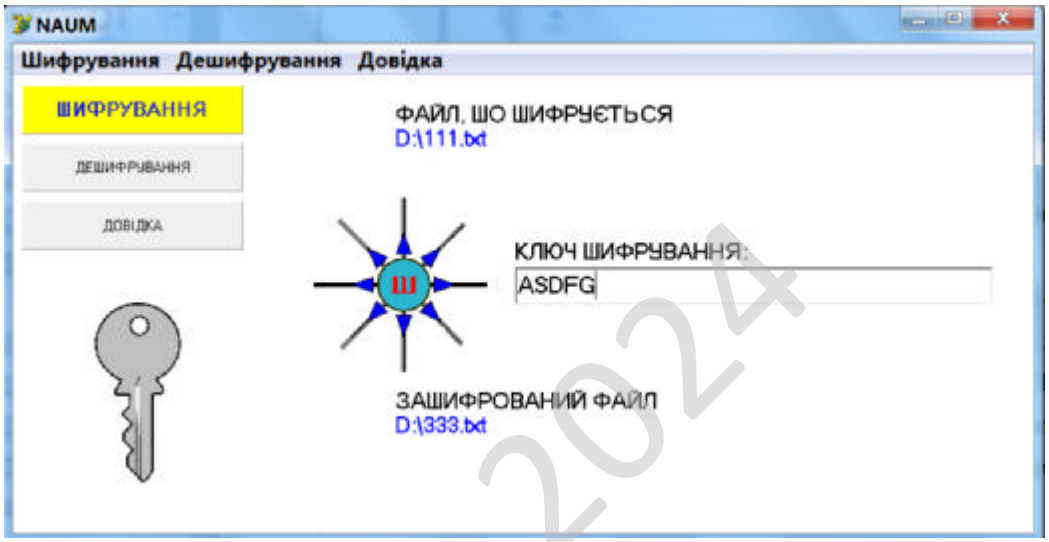

Рисунок 5.6 - Скріншот роботи додатку, введення ключа шифрування.

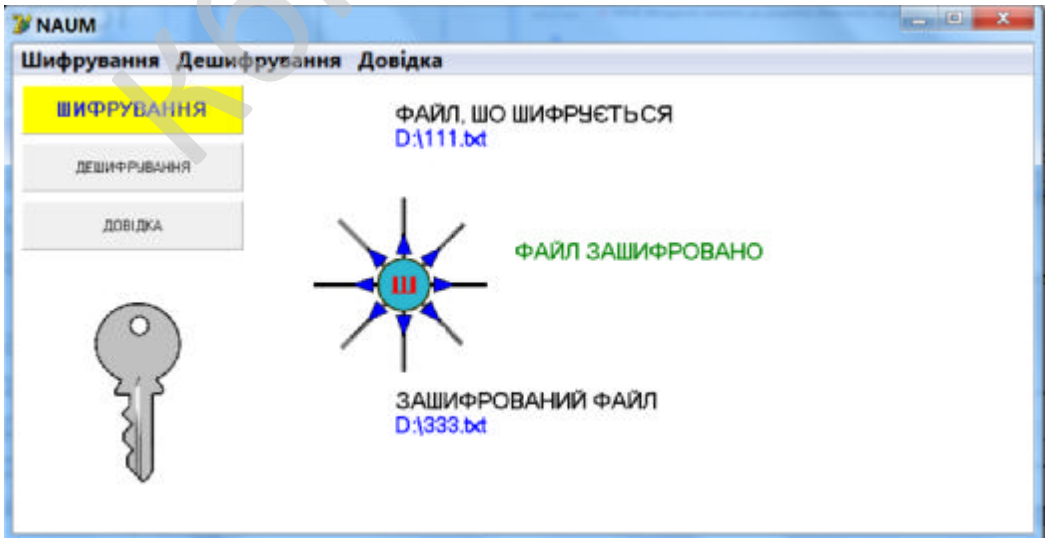

Рисунок 5.7 - Скріншот роботи додатку, повідомлення про шифрування.

Входження до режиму дешифрування можливе вибором відповідної кнопки (рис.5.8) або шляхом введення одного з параметрів дешифрування через меню (відповідний розділ в меню додатку)

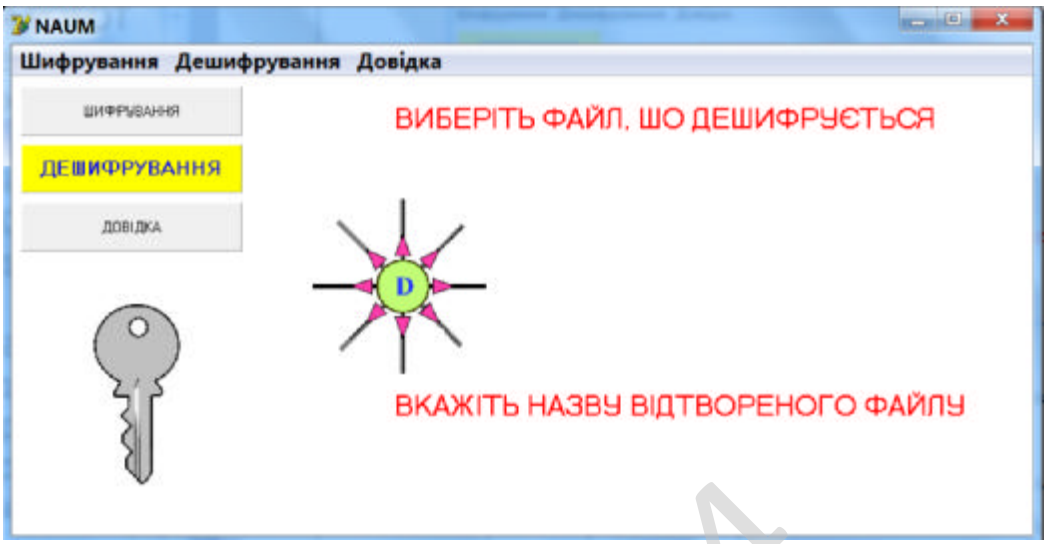

Рисунок 5.8 - Скріншот роботи додатку, режим дешифрування.

Для вибору файлу, що дешифрується, можна скористатися меню додатку або безпосередньо перейди до вибору по посиланню "ВИБЕРІТЬ ФАЙЛ, ЩО ДЕШИФРУЄТЬСЯ", вибір проводиться через діалогове вікно OpenDialog , рис. 5.9.

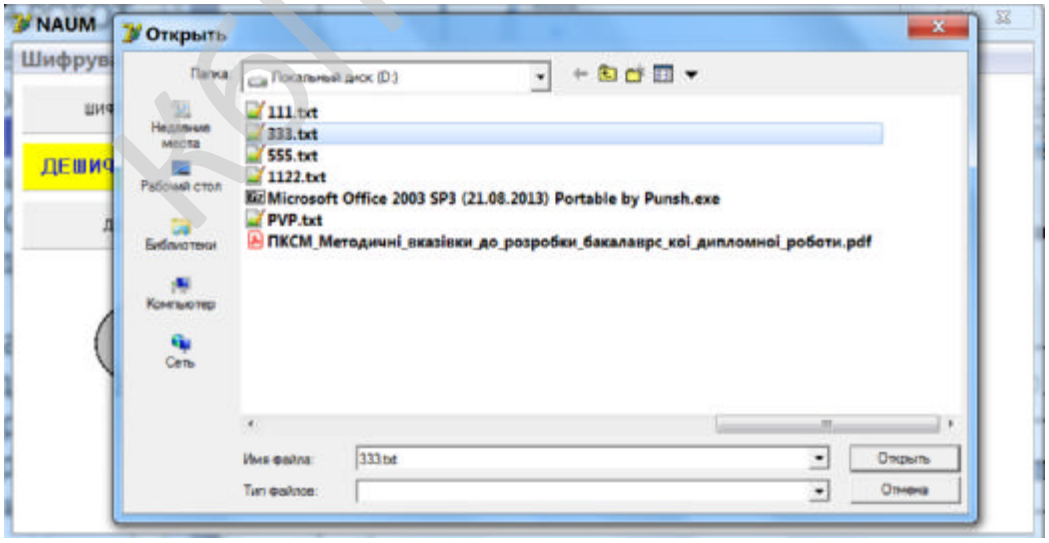

Рисунок 5.9 - Скріншот роботи додатку, вибір файлу, що дешифрується.

Для введення назви файлу, що відтворюється, можна також скористатися меню додатку або безпосередньо перейди до вибору по посиланню "ВКАЖІТЬ НАЗВУ ВІДТВОРЕНОГО ФУЙЛУ ", вибір проводиться через діалогове вікно SaveDialog , рис. 5.10. У випадку, коли зроблено помилковий вибір, наявна можливість змінити вибір файлів для дешифрування (повторний вибір).

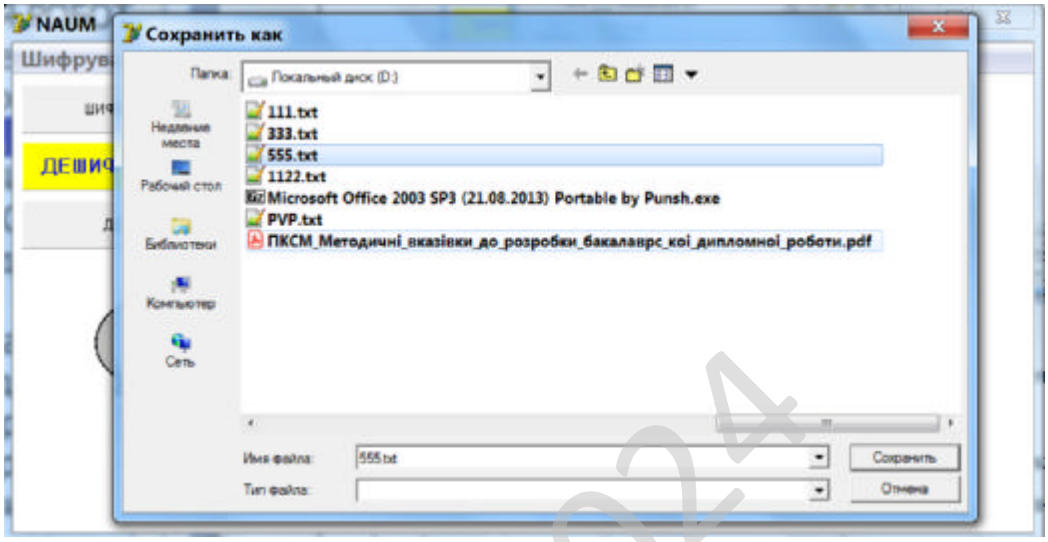

Рисунок 5.10 - Скріншот роботи додатку, введення назви файлу, що відтворюється.

Для введення ключа (пароля) для дешифрування також можна скористатися меню додатку або безпосередньо перейти до вікна введення, натиснувши на кнопку в формі ключа (повторний натиск кнопки ховає вікно введення). Загалом процес дешифрування зворотній процесу шифруванню, а от же значення ключа може залишатися після попередніх відпрацювань додатку незалежно шифрування це чи дешифрування.

Саме дешифрування запускається з меню додатку або безпосередньо натиском на кнопку у вигляді ручки сейфу з буквою "D". Вдале виконання супроводжується повідомленням (рис. 5.11). При виявленні відсутності якогось з необхідних параметрів для проведення шифрування, приведе до запиту на його введення

Результати відпрацювань процесів шифрування та дешифрування представлено в файлах: "111.txt", "333.txt" та "555.txt", рис.5.12..

Рисунок 5.11 - Скріншот роботи додатку, повідомлення про дешифрування.

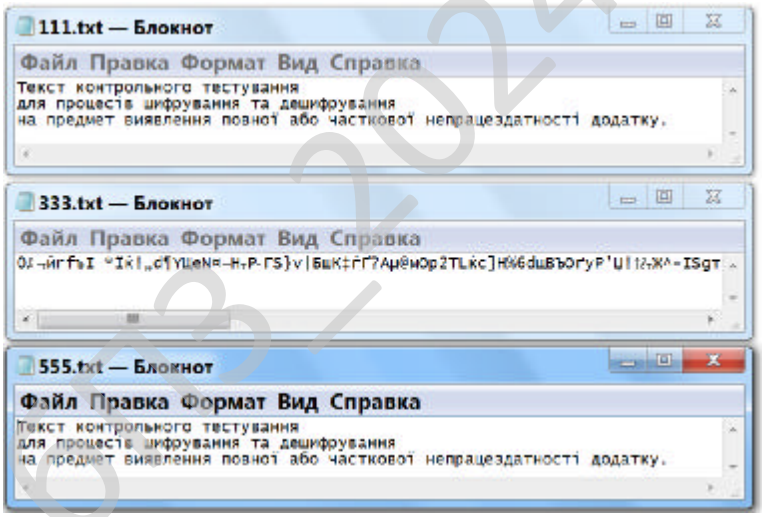

Рисунок 5.12 - Вміст файлів "111.txt", "333.txt" та "555.txt", візуально представлений в текстовому редакторі Блокнот.

Внаслідок можливих виявлених у процесі експлуатації помилок, процес підтримки працездатності ПЗ вимагає внесення зміни в програму, що в підсумку виллється в перехід до її нових версій. Таким чином, для повного супроводу розробленого програмного забезпечення вимагається наявність подальшого контролю розробки відповідним спеціалістом.

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання бакалаврської роботи, призначене для захисту даних

У бакалаврській роботі наведено теоретичне узагальнення й вирішення наукового завдання дослідження методів побудови систем захисту даних шифрувальними методами.

Вирішення даного завдання полягало у здійсненні наступних задач:

- було проведено огляд існуючих програмних рішень для захисту даних шифруванням;

- досліджено та обрано засоби розробки програмного забезпечення системи захисту даних шифрувальними методами;

- розроблено структуру та принципи функціонування системи захисту даних шифрувальними методами;

- проведено аналіз процесів, що протікають при шифруванні та дешифруванні;

- розроблено алгоритм роботи програмного забезпечення системи захисту даних шифрувальними методами;

- на основі отриманих результатів досліджень створена програмна реалізація системи захисту даних шифрувальними методами.

Розроблені під час виконання бакалаврської роботи алгоритми дозволяють успішно вирішувати завдання збереження даних шифрувальними методами.

Проведено аналіз предметної галузі, в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудовано алгоритм і вибрано середовище розробки.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість в освоєнні роботи

програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створенні програмного забезпечення було використано об’єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Delphi. Саме ця мова програмування дозволяє найбільш ефективно обробляти дані. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки, й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозначної операційної системи сімейства Windows.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати один із загально розповсюджених методів - прив’язка до параметрів комп’ютера, в нашому випадку організовано прив’язку до жорсткого диску.

У цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Технології захисту інформації: підручник для студ. спеціальності 122 «Комп'ютерні науки», спеціалізацій «Інформаційні технології моніторингу довкілля», «Геометричне моделювання в інформаційних системах» / Ю. А. Тарнавський; КПІ ім. Ігоря Сікорського. - Київ : КПІ ім. Ігоря Сікорського, 2018. - 162 с.

2. Остапов С. Е. Технології захисту інформації: навчальний посібник / С. Е. Остапов, С. П. Євсеєв, О. Г. Король. - Х. : Вид. ХНЕУ, 2013. - 476 с.

3. Шифрування [Електронний ресурс] -Режим доступу: https://www.eset.com/ua/support/information/entsiklopediya-ugroz/shifrovaniye/

4. Основи шифрування [Електронний ресурс] -Режим доступу: https://sites.google.com/site/detectivesdiary/kriptografia/osnovi-sifruvanna

5. Шифрування: типи і алгоритми. [Електронний ресурс] -Режим доступу: https://wiki.hostpro.ua/ua/knowledgebase/shifruvannja-tipi-i-algoritmi/

6. Симетричне шифрування [Електронний ресурс] - Режим доступу: https://encyclopedia.kaspersky.ru/glossary/symmetric-encryption/

7. Захист папок і файлів від несанкціонованого доступу. Програма TrueCrypt. [Електронний ресурс] - Режим доступу: http://3das.com.ua/zahist-papok-i-fajliv-vid-nesanktsionovanogo-dostupu-programa-truecrypt/

8. «TrueCrypt» - програма для шифрування[Електронний ресурс] - Режим доступу: https://te-st.ru/entries/truecrypt/

9. Програми для шифрування папок і файлів[Електронний ресурс] - Режим доступу: https://uk.soringpcrepair.com/program-to-encrypt-folders-and-files/#PGP_Desktop

10. Folder Lock для Windows [Електронний ресурс] - Режим доступу: http://softobase.com/ru/vopros/kak-polzovatsya-folder-lock

11. Порівняння настільних програм для шифрування [Електронний ресурс] - Режим доступу: https://cybersafesoft.com/post.php?id=6259

12. PGP Desktop [Електронний ресурс] - Режим доступу:

https://8d9.ua/program/pgp-desktop

13. CyberSafe Top Secret [Електронний ресурс] - Режим доступу: https://8d9.ua/program/cybersafe-top-secret

14. Огляд та рейтинг AxCrypt [Електронний ресурс] - Режим доступу: https://uk.sijiwae.net/axcrypt-premium-8779

15. Шифрування окремих файлів AxCrypt [Електронний ресурс] - Режим доступу: https://uk.anipi.org/free-file-encryption-software-windows-4657

16. Середовище візуального програмування Delphi [Електронний ресурс] - Режим доступу: http://bourabai.kz/einf/Delphi/index.htm

17. Огляд Borland Delphi [Електронний ресурс] - Режим доступу: http://bourabai.kz/einf/Delphi/LES01.htm

18. Безменов М. І. Основи програмування у середовищі Delphi : навч. посіб. -Харків : НТУ «ХПІ», 2010. 608 с.

19. Delphi. Робота з HDD, CD-ROM, ZIP - [Електронний ресурс] - Режим доступу: http://z-oleg.com/delphi/hardw1.htm

20. Delphi FAQ [Електронний ресурс] - Режим доступу: http://www.delphisources.ua/pages/faq/base/registry_influence.html

21. Вибір алгоритму шифрування [Електронний ресурс] - Режим доступу: https://docs.microsoft.com/ua-ua/sql/relational-databases/security/encryption/choose-an-encryption-algorithm?view=sql-server-ver15

22. Призначення і структура алгоритмів шифрування [Електронний ресурс] - Режим доступу: https://www.ixbt.com/soft/alg-encryption.shtml

23. Основи UML Проектування розподілених систем [Електронний ресурс] - Режим доступу: http://moodle.ipo.kpi.ua/moodle/mod/resource/view.php?inpopup=true&id=44416

24. Технології створення програмних продуктів та інформаційних систем : навч. посібник / М. Ю. Карпенко, Н. О. Манакова, І. О. Гавриленко ;- Харків : ХНУМГ ім. О. М. Бекетова, 2017. -93 с.

25. Що таке мова програмування [Електронний ресурс]. – Режим доступу : https://www.mathros.net.ua/shho-take-mova-programuvannja.html.

26. Методичні вказівки до виконання й захисту випускної випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти за спеціальністю 123 «Комп'ютерна інженерія». / проф. Смірнов О.А., проф. Мелешко Є.В., доц. Гермак В.С., доц. Буравченко К.О., доц. Якименко Н.М., доц. Смірнов С.А., доц.. Доренський О.П., доц. Смірнова Т.В. – Кропивницький: ЦНТУ, 2022. - 68 с.

27. Основи програмування в середовищі Delphi : навч. посібник / Безменов М.І., 2010. -608 с.

28. Штаєр Л.О. Технології розробки програмного забезпечення; конспект лекцій. – Івано-Франківськ: ІФНТУНГ, 2017. – 139 с. (Електронний ресурс)

29. Роберт М. Чистий код: створення, аналіз, рефакторинг / Роберт М. – Харків : Фабула, 2019 – 416с.

30. Роберт М. Чиста архітектура: мистецтво розробки програмного забезпечення / Роберт М. – Харків : Фабула, 2019 – 416с.

31. Кліффорд Ш. Алгоритми: Побудова та аналіз / Томас К., Чарльз Л., Рональд Р., Кліффорд Ш. – Київ : Діалектика, 2022 – 1328с.

32. Алан К. About Face: The Essentials of Interaction Design / Алан К., Роберт Р., Давід К., Крістофре Н. – Індіанаполіс : John Wiley & Sons, 2014 – 720с.

33. Стів К. Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability / Стів К. – Сан-Франциско : New Riders, 2014 – 216с.

34. Грег Н. Android Design Patterns: Interaction Design Solutions for Developers / Грег Н. – Індіанаполіс : John Wiley & Sons, 2013 – 456с.

35. П'єр-Олівер Л. Programming Android with Kotlin: Achieving Structured Concurrency with Coroutines / П'єр-Олівер Л., Аманда Х.-Д., Блейк М., Майк Д. – Себастопол : O'Reilly Media, 2022 – 352с.

36. Джош С. Kotlin Programming: The Big Nerd Ranch Guide / Джош С., Девід Г. – Індіанаполіс : Big Nerd Ranch, 2018 – 480с.

37. Білл Ф. Android Programming: The Big Nerd Ranch Guide / Білл Ф., Кріс С., Браян Х., Крістін М. – Індіанаполіс : Big Nerd Ranch, 2015 – 618с.

38. Марцін М. Android Development with Kotlin: Enhance your skills for Android development using Kotlin / Марцін М., Ігор В. – Бірмінгем : Packt Publishing, 2017 – 440с.

39. Нібедіт Д. ross-Platform Development with Qt 6 and Modern C++: Design and build applications with modern graphical user interfaces without worrying about platform dependency / Нібедіт Д. – Бірмінгем : Packt Publishing, 2021 – 442с.

40. Пітер С. Frontend Development with JavaFX and Kotlin: Build State-of-the-Art Kotlin GUI Applications / Пітер С. – Нью Йорк : Apress Media, 2023 – 152с.

41. Стаття «47 важливих порад для UI та UX дизайнерів» на сайті uxpub: https://ux.pub/editorial/47-vazhlivikh-porad-dlia-ui-ta-ux-dizainieriv-4h0j

42. Стаття «The ubiquitous digital file: A review of file management research» на сайті researchgate: https://www.researchgate.net/publication/354723356_The_ubiquitous_digital_file_A_review_of_file_management_research

43. Стаття «File system» на сайті techtarget: https://www.techtarget.com/searchstorage/definition/file-system

44. Стаття «A File Transfer Protocol (FTP)» на сайті sciencedirect: https://www.sciencedirect.com/science/article/abs/pii/0376507578900090

45. Стаття «What Is FTP» на сайті spiceworks: https://www.spiceworks.com/tech/networking/articles/what-is-ftp/

46. Стаття «A Complete Guide to Socket Programming in Python» на сайті datacamp: https://www.datacamp.com/tutorial/a-complete-guide-to-socket-programming-in-python

47. Стаття у Вікіпедії «FTP»: https://uk.wikipedia.org/wiki/FTP

48. Стаття у Вікіпедії «Регулярний вираз»: https://uk.wikipedia.org/wiki/Регулярний вираз

49. Стаття у Вікіпедії «Файлова система»: https://uk.wikipedia.org/wiki/Файлова система

50. Стаття у Вікіпедії «Сокет»: https://uk.wikipedia.org/wiki/Сокет (програмний інтерфейс)

Додаток А
(обов'язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку програмного забезпечення для захисту даних методами шифрування

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу, видане на кафедрі програмування та захисту інформації (нак. _____ від _____ року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної бакалаврської роботи є розробка програмного забезпечення для захисту даних методами шифрування.

4 Джерела розробки

Джерелом цієї кваліфікаційної бакалаврської дипломної роботи є відносна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частин системи, а також розробка взаємодії системи з ОС та з користувачем;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– роботу системи керування процесом сушіння деревини;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ архітектури IBM PC, працювати в ОС Windows XP/Vista/7/8/10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпеченням.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows XP/Vista/7/8/10/11.

5.8.1 Обладнання

Комп'ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14" 1Mb або сумісні з ним.

5.8.2 Мова програмування

Delphi.

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Перелік документів, що розробляються

– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму головної програми – 1 аркуш
– Блок-схема алгоритму роботи підпрограм – 1 аркуш.
– Пояснювальна записка – 61 аркушів.

8 Етапи розробки

8.1 Збір і обробка інформації по темі кваліфікаційної бакалаврської роботи.

Постановка задачі на виконання кваліфікаційної роботи (складання ТЗ).

8.2 Проведення досліджень або експериментальних робіт для уточнення основних положень кваліфікаційної роботи.

8.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

9.1 Подання бакалаврської дипломної роботи на попередній захист __.__. 2024 р.

9.2 Подання бакалаврської дипломної роботи на захист __.__.2024 р.

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник дипломного проекту

_____ Пархоменко Ю.М.

Програмне забезпечення захищеності даних методами шифрування

Лістинг програми

Код документу 12

Носій: DVD-RW диск

Загальна кількість аркушів: 22

Літера: РП

Кропивницький 2024

```
// -------------------- naum.pas -------------------- Програмний модуль
unit NAUM;

interface

uses
     Windows, Messages, SysUtils, Variants, Classes, Graphics,
     Controls, Forms, Dialogs, StdCtrls, Buttons, ExtCtrls, jpeg,
     Menus, ComCtrls, ActnList, Registry;

type
  TForm1 = class(TForm)
    Image1: TImage;
    Image2: TImage;
    Image3: TImage;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label5: TLabel;
    Label4: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    RichEdit1: TRichEdit;
    Edit1: TEdit;
    MainMenu1: TMainMenu;
    N1: TMenuItem;
    N11: TMenuItem;
    N21: TMenuItem;
    N31: TMenuItem;
    N4: TMenuItem;
    N2: TMenuItem;
    N12: TMenuItem;
    N22: TMenuItem;
    N32: TMenuItem;
    N5: TMenuItem;
    N3: TMenuItem;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    OpenDialog2: TOpenDialog;
    SaveDialog2: TSaveDialog;
    Button2: TButton;
    procedure GPVP(Sender: TObject);
    procedure Panel3Click(Sender: TObject);
    procedure Panel2Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Panel1Click(Sender: TObject);
    procedure Image3Click(Sender: TObject);
    procedure Image1Click(Sender: TObject);
    procedure N11Click(Sender: TObject);
    procedure N21Click(Sender: TObject);
    procedure N31Click(Sender: TObject);
    procedure N4Click(Sender: TObject);
    procedure N12Click(Sender: TObject);
    procedure N22Click(Sender: TObject);
    procedure N3Click(Sender: TObject);
    procedure Image2Click(Sender: TObject);
    procedure N32Click(Sender: TObject);
    procedure Label2Click(Sender: TObject);
    procedure Label4Click(Sender: TObject);
```

```
    procedure Label3Click(Sender: TObject);
    procedure Label5Click(Sender: TObject);
    procedure Label7Click(Sender: TObject);
    procedure Label6Click(Sender: TObject);
    procedure Label8Click(Sender: TObject);
    procedure Label9Click(Sender: TObject);
    procedure N5Click(Sender: TObject);
    procedure ZAHYST(Sender: TObject);
    procedure Button2Click(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  XK,YK,XD,YD,i1,i2:INTEGER;
  PVP: ARRAY [1..1000] OF INTEGER;
  A:BOOLEAN;
  ID : DWord;
  IDR:STRING;

implementation

{$R *.dfm}

//                    ГЕНЕРАТОР ПСЕВДОВИПАДКОВОЇ ПОСЛІДОВНОСТІ
procedure TForm1.GPVP(Sender: TObject);
var p:string;
    i,n:integer;
begin
n:=255;
p:=edit1.Text;
for i:=1 to 1000 do
begin
if  i>Length(p)
then pvp[i]:=( (1+pvp[i-1] )   * (1+pvp[i-Length(p)] )  mod n)
else pvp[i]:=ord(p[i]);
end;
end;


procedure TForm1.Panel3Click(Sender: TObject);
begin
IF A
THEN
BEGIN
Image1.Visible:=FALSE;
Image2.Visible:=FALSE;
Panel1.Font.Color:= clBlack;
Panel1.Color:=clBtnFace;
Panel1.Font.Size:=8;
Panel1.Font.Style:=Panel1.Font.Style-[fsbold];
Panel2.Font.Color:= clBlack;
Panel2.Color:=clBtnFace;
Panel2.Font.Size:=8;
Panel2.Font.Style:=Panel1.Font.Style-[fsbold];
Panel3.Font.Color:=clBlue;
Panel3.Color:=cLYellow;
Panel3.Font.Size:=12;
Panel3.Font.Style:=Panel1.Font.Style+[fsbold];
RichEdit1.Top:=8++XD;
```

```
RichEdit1.Left:=8+8+185;
RichEdit1.Width:=form1.Width-40-185;
RichEdit1.Height:=form1.Height-85-2* XD;
RichEdit1.Visible:=true;
RichEdit1.Lines.Clear;
with RichEdit1.SelAttributes do
begin
Color:=clBlack;
Size:=11;
end;
RichEdit1.Lines.Add('Додаток NAUM призначено для організації захисту даних
шифруванням.');
with RichEdit1.SelAttributes do
begin
Color:=clBlack;
Size:=11;
end;
RichEdit1.Lines.Add('Розробник: Науменко Ігор Олегович.');
with RichEdit1.SelAttributes do
begin
Color:=clBlack;
Size:=11;
end;
RichEdit1.Lines.Add('Рік розробки: 2021.');
RichEdit1.Lines.Add('');
RichEdit1.Lines.Add('');
with RichEdit1.SelAttributes do
begin
Color:=clred;
Size:=14;
end;
RichEdit1.Lines.Add('УВАГА !!!  -   ПРОГРАМА ЗАХИЩЕНА ВІД НЕЛІЦЕНЗІЙНОГО
КОПІЮВАННЯ');
RichEdit1.Lines.Add('');
with RichEdit1.SelAttributes do
begin
Color:=clBLUE;
Size:=16;
end;
RichEdit1.Lines.Add('ID: '+INTTOSTR(ID));
RichEdit1.ReadOnly:=true;
Image3.Visible:=false;
edit1.Visible:=false;
label1.Visible:=false;
LABEL2.Visible:=FALSE;
LABEL3.Visible:=FALSE;
LABEL4.Visible:=FALSE;
LABEL5.Visible:=FALSE;
END
ELSE
BEGIN
RichEdit1.Top:=2;
RichEdit1.Left:=2;
RichEdit1.Width:=form1.Width-28;
RichEdit1.Height:=form1.Height-79;
RichEdit1.Visible:=true;
RichEdit1.Lines.Clear;
with RichEdit1.SelAttributes do
begin
Color:=clBlack;
Size:=11;
end;
RichEdit1.Lines.Add('');
RichEdit1.Lines.Add('');
```

```
RichEdit1.Lines.Add('Додаток NAUM призначено для організації захисту даних
шифруванням.');
with RichEdit1.SelAttributes do
begin
Color:=clBlack;
Size:=11;
end;
RichEdit1.Lines.Add('Розробник: Науменко Ігор Олегович.');
with RichEdit1.SelAttributes do
begin
Color:=clBlack;
Size:=11;
end;
RichEdit1.Lines.Add('Рік розробки: 2021.');
RichEdit1.Lines.Add('');
RichEdit1.Lines.Add('');
with RichEdit1.SelAttributes do
begin
Color:=clred;
Size:=14;
end;
RichEdit1.Lines.Add('УВАГА !!!  -   ПРОГРАМА ЗАХИЩЕНА ВІД НЕЛІЦЕНЗІЙНОГО
КОПІЮВАННЯ');
RichEdit1.Lines.Add('');
with RichEdit1.SelAttributes do
begin
Color:=clBLUE;
Size:=14;
end;
RichEdit1.Lines.Add('ДЛЯ ЗАПУСКУ ПРОГРАМИ ВСТАНОВІТЬ ID ВАШОГО ПК');
RichEdit1.SetFocus;
RichEdit1.ReadOnly:=true;
Image3.Visible:=false;
edit1.Visible:=false;
label1.Visible:=false;
LABEL2.Visible:=FALSE;
LABEL3.Visible:=FALSE;
LABEL4.Visible:=FALSE;
LABEL5.Visible:=FALSE;
BUTTON2.Visible:=TRUE;
BUTTON2.Caption:='ВСТАНОВИТИ ID';
BUTTON2.Top:=300;
BUTTON2.Left:=50;
BUTTON2.FONT.SIZE:=16;
BUTTON2.Width:=FORM1.Width-100;
BUTTON2.Height:=60;
END;
end;


procedure TForm1.Panel2Click(Sender: TObject);
begin
label1.font.Color:=clBlack;
label1.caption:='КЛЮЧ ШИФРУВАННЯ:';
if edit1.Visible=false
then label1.Visible:=false;
label1.font.Color:=clBlack;
label1.caption:='КЛЮЧ ШИФРУВАННЯ:';
Image2.Center := True;
Image2.Proportional := True;
Image2.Stretch := True;
Image2.Top:=XK;
Image2.Left:=YK;
Image2.Width:=150;
```

```
Image2.Height:=150;
Image2.BringtoFront;
Image1.Visible:=FALSE;
Image2.Visible:=TRUE;
RichEdit1.Visible:=FALSE;
Panel1.Font.Color:= clBlack;
Panel1.Color:=clBtnFace;
Panel1.Font.Size:=8;
Panel1.Font.Style:=Panel1.Font.Style-[fsbold];
Panel2.Font.Color:=clBlue;
Panel2.Color:=cLYellow;
Panel2.Font.Size:=12;
Panel2.Font.Style:=Panel1.Font.Style+[fsbold];
Panel3.Font.Color:= clBlack;
Panel3.Color:=clBtnFace;
Panel3.Font.Size:=8;
Panel3.Font.Style:=Panel1.Font.Style-[fsbold];
Image3.Visible:=true;
label6.Visible:=true;
label6.Color:=clWhite;
IF OpenDialog2.FileNAme=''
THEN
BEGIN
label7.Visible:=FALSE;
label6.Top:=20;
END
ELSE
BEGIN
label7.Visible:=TRUE;
label6.Top:=40;
END;
label8.Visible:=true;
label8.Color:=clWhite;
IF SaveDialog2.FileNAme=''
THEN
BEGIN
label9.Visible:=FALSE;
label8.Top:=260;
END
ELSE
BEGIN
label9.Visible:=TRUE;
label8.Top:=280;
END;
label6.Visible:=true;
label6.Color:=clWhite;
IF OpenDialog2.FileNAme=''
THEN
BEGIN
label7.Visible:=FALSE;
label6.Top:=20;
END
ELSE
BEGIN
label7.Visible:=TRUE;
label6.Top:=40;
END;
label8.Visible:=true;
label8.Color:=clWhite;
IF SaveDialog2.FileNAme=''
THEN
BEGIN
label9.Visible:=FALSE;
label8.Top:=260;
```

```
END
ELSE
BEGIN
label9.Visible:=TRUE;
label8.Top:=280;
END;
label2.Visible:=FALSE;
label3.Visible:=FALSE;
label4.Visible:=FALSE;
label5.Visible:=FALSE;
end;


procedure TForm1.FormCreate(Sender: TObject);
begin
BUTTON2.Visible:=FALSE;
ZAHYST(Sender);
IF A
THEN
BEGIN
Form1.caption:='NAUM';
XK:=100;
YK:=250;
XD:=0;
Form1.Color:= clWhite;
Image1.Visible:=FALSE;
Image2.Visible:=FALSE;
Panel1.Top:=8+XD;
Panel1.Left:=8;
Panel1.Width:=185;
Panel1.Height:=41;
Panel2.Top:=56+XD;
Panel2.Left:=8;
Panel2.Width:=185;
Panel2.Height:=41;
Panel3.Top:=105+XD;
Panel3.Left:=8;
Panel3.Width:=185;
Panel3.Height:=41;
Image3.Center := True;
Image3.Proportional := True;
Image3.Stretch := True;
Image3.Top:=190;
Image3.Left:=30;
Image3.Width:=150;
Image3.Height:=150;
Image3.BringtoFront;
edit1.Top:=160;
edit1.Left:=420;
edit1.Width:=400;
edit1.font.size:=16;
edit1.font.Color:=clBlack;
edit1.text:='';
label1.Top:=135;
label1.Left:=420;
label1.caption:='КЛЮЧ ШИФРУВАННЯ:';
label2.Top:=20;
label2.Left:=320;
label2.Width:=form1.Width-390;
label2.Height:=75;
label2.Visible:=true;
label2.FONT.Color:=clRED;
label2.FONT.Size:=18;
label2.caption:='ВИБЕРІТЬ ФАЙЛ, ШО ШИФРУСТЬСЯ';
```

```
label3.Visible:=FALSE;
label3.Top:=20;
label3.Left:=320;
label3.Width:=form1.Width-390;
label3.Height:=75;
label3.FONT.Color:=clBlack;
label3.FONT.Size:=16;
label3.caption:='ФАЙЛ, ШО ШИФРУЄТЬСЯ';
label4.Top:=260;
label4.Left:=320;
label4.Width:=form1.Width-390;
label4.Height:=75;
label4.Visible:=true;
label4.FONT.Color:=clred;
label4.FONT.Size:=18;
label4.caption:='ВКАЖІТЬ НАЗВУ ЗАШИФРОВАНОГО ФАЙЛУ';
label5.Visible:=FALSE;
label5.Top:=260;
label5.Left:=320;
label5.Width:=form1.Width-390;
label5.Height:=75;
label5.FONT.Color:=clBlack;
label5.FONT.Size:=16;
label5.caption:='ЗАШИФРОВАНИЙ ФАЙЛ';
label6.Top:=20;
label6.Left:=320;
label6.Width:=form1.Width-390;
label6.Height:=75;
label6.Visible:=true;
label6.FONT.Color:=clRED;
label6.FONT.Size:=18;
label6.caption:='ВИБЕРІТЬ ФАЙЛ, ШО ДЕШИФРУЄТЬСЯ';
label7.Visible:=FALSE;
label7.Top:=20;
label7.Left:=320;
label7.Width:=form1.Width-390;
label7.Height:=75;
label7.FONT.Color:=clBlack;
label7.FONT.Size:=16;
label7.caption:='ФАЙЛ, ШО ДЕШИФРУЄТЬСЯ';
label8.Top:=260;
label8.Left:=320;
label8.Width:=form1.Width-390;
label8.Height:=75;
label8.Visible:=true;
label8.FONT.Color:=clred;
label8.FONT.Size:=18;
label8.caption:='ВКАЖІТЬ НАЗВУ ВІДТВОРЕНОГО ФАЙЛУ';
label9.Visible:=FALSE;
label9.Top:=260;
label9.Left:=320;
label9.Width:=form1.Width-390;
label9.Height:=75;
label9.FONT.Color:=clBlack;
label9.FONT.Size:=16;
label9.caption:='ЗАШИФРОВАНИЙ ФАЙЛ';
Panel3Click(Sender);
END
ELSE
BEGIN
label1.Visible:=FALSE;
label2.Visible:=FALSE;
label3.Visible:=FALSE;
label4.Visible:=FALSE;
```

```
label5.Visible:=FALSE;
label6.Visible:=FALSE;
label7.Visible:=FALSE;
label8.Visible:=FALSE;
label9.Visible:=FALSE;
Panel1.Visible:=FALSE;
Panel2.Visible:=FALSE;
Panel3.Visible:=FALSE;
Image1.Visible:=FALSE;
Image2.Visible:=FALSE;
Image3.Visible:=FALSE;
EDIT1.Visible:=FALSE;
N11.Enabled:=false;
N21.Enabled:=false;
N31.Enabled:=false;
N4.Enabled:=false;
N22.Enabled:=false;
N12.Enabled:=false;
N32.Enabled:=false;
N5.Enabled:=false;
RichEdit1.Top:=1;
RichEdit1.Left:=1;
RichEdit1.Width:=form1.Width-20;
RichEdit1.Height:=form1.Height-40;
RichEdit1.Visible:=true;
RichEdit1.Lines.Clear;
RichEdit1.Paragraph.Alignment:=taCenter;
with RichEdit1.SelAttributes do
begin
Color:=clBlue;
Size:=14;
end;
RichEdit1.Lines.Add('');
RichEdit1.Lines.Add('НЕЛІЦЕНЗІЙНА КОПІЯ ПРОГРАМИ ! ! ! ');
RichEdit1.Lines.Add('');
RichEdit1.Lines.Add('');
with RichEdit1.SelAttributes do
begin
Color:=clRED;
Size:=18;
end;
RichEdit1.Lines.Add('ФУНКЦІОНУВАННЯ ДОДАТКУ ОБМЕЖЕНО');
END;
end;


procedure TForm1.Panel1Click(Sender: TObject);
begin
label1.font.Color:=clBlack;
label1.caption:='КЛЮЧ ШИФРУВАННЯ:';
if edit1.Visible=false
then
label1.Visible:=false;
Image1.Center := True;
Image1.Proportional := True;
Image1.Stretch := True;
Image1.Top:=XK;
Image1.Left:=YK;
Image1.Width:=150;
Image1.Height:=150;
Image1.BringtoFront;
Image2.Visible:=FALSE;
Image1.Visible:=TRUE;
RichEdit1.Visible:=FALSE;
```

```
Panel1.Font.Color:=clBlue;
Panel1.Color:=cLYellow;
Panel1.Font.Size:=12;
Panel1.Font.Style:=Panel1.Font.Style+[fsbold];
Panel2.Font.Color:= clBlack;
Panel2.Color:=clBtnFace;
Panel2.Font.Size:=8;
Panel2.Font.Style:=Panel1.Font.Style-[fsbold];
Panel3.Font.Color:= clBlack;
Panel3.Color:=clBtnFace;
Panel3.Font.Size:=8;
Panel3.Font.Style:=Panel1.Font.Style-[fsbold];
Image3.Visible:=true;
label2.Visible:=true;
label2.Color:=clWhite;
IF OpenDialog1.FileNAme=''
THEN
BEGIN
label3.Visible:=FALSE;
label2.Top:=20;
END
ELSE
BEGIN
label3.Visible:=TRUE;
label2.Top:=40;
END;
label4.Visible:=true;
label4.Color:=clWhite;
IF SaveDialog1.FileNAme=''
THEN
BEGIN
label5.Visible:=FALSE;
label4.Top:=260;
END
ELSE
BEGIN
label5.Visible:=TRUE;
label4.Top:=280;
END;
label6.Visible:=FALSE;
label7.Visible:=FALSE;
label8.Visible:=FALSE;
label9.Visible:=FALSE;
end;


procedure TForm1.Image3Click(Sender: TObject);
begin
label1.font.Color:=clBlack;
label1.caption:='КЛЮЧ ШИФРУВАННЯ:';
if edit1.Visible=true
then
begin
edit1.Visible:=false;
label1.Visible:=false;
end
else
begin
label1.Visible:=true;
edit1.Visible:=true;
end;
end;

//                                                ШИФРУВАННЯ
```

```
procedure TForm1.Image1Click(Sender: TObject);
var
//////////////////////////////////////////////////////
// ОРИГІНАЛЬНЕ ШИФРУВАННЯ ПІД ТЕКСТОВИЙ ФАЙЛ:
{
    Fd,Fz : TextFile;
}
// АДАПТАЦЫЯ РЕАЛІЗАЦІЇ
// ДЛЯ ВИКЛЮЧЕННЯ МОЖЛИВОЪ ПОЯАИ
// СИМВОЛУ З КОДОМ 26 (КІНЦЯ ФАЙЛУ)
    Fd : TextFile;
    Fz : file of byte;
    ss:byte;
 //////////////////////////////////////////////////////
    i:integer;
    p:char;
begin
if OpenDialog1.FileNAme=''
then
BEGIN
label1.Visible:=true;
label1.font.Color:=clRED;
label1.caption:='НЕ ВВЕДЕНО ФАЙЛ, ЩО ШИФРУЄТЬСЯ';
edit1.Visible:=FALSE;
END
ELSE
if SaveDialog1.FileNAme=''
then
BEGIN
label1.Visible:=true;
label1.font.Color:=clRED;
label1.caption:='НЕ ВВЕДЕНО НАЗВУ ФАЙЛУ ЗБЕРЕЖЕННЯ';
edit1.Visible:=FALSE;
END
ELSE
if Length(edit1.text)<4
then
BEGIN
edit1.Visible:=true;
label1.Visible:=true;
label1.font.Color:=clRED;
label1.caption:='НЕНАДІЙНИЙ КЛЮЧ ШИФРУВАННЯ';
END
ELSE
begin
label1.font.Color:=clBlack;
label1.caption:='КЛЮЧ ШИФРУВАННЯ:';
GPVP(Sender);
AssignFile(Fd,OpenDialog1.FileNAme);
ReSet(Fd);
AssignFile(Fz,SaveDialog1.FileNAme);
ReWrite(Fz);
i:=1;
while (not EOF(Fd))
do
begin
Read(Fd,p);
//////////////////////////////////////////////////////
// ОРИГІНАЛЬНЕ ШИФРУВАННЯ ПІД ТЕКСТОВИЙ ФАЙЛ:
{
Write(Fz,chr((ORD(P)+pvp[((i-1) mod 1000)+1])));
}
// АДАПТАЦЫЯ РЕАЛІЗАЦІЇ
// ДЛЯ ВИКЛЮЧЕННЯ МОЖЛИВОЪ ПОЯАИ
```

```
// СИМВОЛУ З КОДОМ 26 (КІНЦЯ ФАЙЛУ)
SS:=((ORD(P)+pvp[((i-1) mod 1000)+1]) mod 256);
Write(Fz,SS);
////////////////////////////////////////////////
i:=i+1;
end;
CloseFile(Fd);
CloseFile(Fz);
edit1.Visible:=false;
label1.Visible:=true;
label1.font.Color:=clGREEN;
label1.caption:='ФАЙЛ ЗАШИФРОВАНО';
ENd;
end;


procedure TForm1.N11Click(Sender: TObject);
begin
if OpenDialog1.Execute=true
then
begin
Panel1Click(Sender);
label2.font.Color:=clBlue;
label2.font.size:=14;
label2.Color:=clWhite;
label2.Caption:=OpenDialog1.FileNAme;
end;
end;


procedure TForm1.N21Click(Sender: TObject);
begin
if SaveDialog1.Execute=true
then
begin
Panel1Click(Sender);
label4.font.Color:=clBlue;
label4.font.size:=14;
label4.Color:=clWhite;
label4.Caption:=SaveDialog1.FileNAme;
end;
end;


procedure TForm1.N31Click(Sender: TObject);
begin
Panel1Click(Sender);
label1.Visible:=true;
edit1.Visible:=true;
end;


procedure TForm1.N4Click(Sender: TObject);
begin
Panel1Click(Sender);
Image1Click(Sender);
end;


procedure TForm1.N12Click(Sender: TObject);
begin
if OpenDialog2.Execute=true
then
begin
```

```
Panel2Click(Sender);
label6.font.Color:=clBlue;
label6.font.size:=14;
label6.Color:=clWhite;
label6.Caption:=OpenDialog2.FileNAme;
end;
end;


procedure TForm1.N22Click(Sender: TObject);
begin
if SaveDialog2.Execute=true
then
begin
Panel2Click(Sender);
label8.font.Color:=clBlue;
label8.font.size:=14;
label8.Color:=clWhite;
label8.Caption:=SaveDialog2.FileNAme;
end;
end;


procedure TForm1.N3Click(Sender: TObject);
begin
Panel3Click(Sender);
end;

//                                          ДЕШИФРУВАННЯ
procedure TForm1.Image2Click(Sender: TObject);
var
//////////////////////////////////////////////////////
// ОРИГІНАЛЬНЕ ШИФРУВАННЯ ПІД ТЕКСТОВИЙ ФАЙЛ:
{
    Fk,Fv : TextFile;
}
// АДАПТАЦЫЯ РЕАЛІЗАЦІЇ
// ДЛЯ ВИКЛЮЧЕННЯ МОЖЛИВОЪ ПОЯАИ
// СИМВОЛУ З КОДОМ 26 (КІНЦЯ ФАЙЛУ)
    Fv : TextFile;
    Fk : file of byte;
    ss:byte;
//////////////////////////////////////////////////
    i:integer;
    p:char;
begin
if OpenDialog2.FileNAme=''
then
BEGIN
label1.Visible:=true;
label1.font.Color:=clRED;
label1.caption:='НЕ ВВЕДЕНО ФАЙЛ, ЩО ДЕШИФРУЄТЬСЯ';
edit1.Visible:=FALSE;
END
ELSE
if SaveDialog2.FileNAme=''
then
BEGIN
label1.Visible:=true;
label1.font.Color:=clRED;
label1.caption:='НЕ ВВЕДЕНО НАЗВУ ФАЙЛУ ВІДТВОРЕННЯ';
edit1.Visible:=FALSE;
END
ELSE
```

```
if Length(edit1.text)<4
then
BEGIN
edit1.Visible:=true;
label1.Visible:=true;
label1.font.Color:=clRED;
label1.caption:='НЕНАДІЙНИЙ КЛЮЧ ШИФРУВАННЯ';
END
ELSE
begin
label1.font.Color:=clBlack;
label1.caption:='КЛЮЧ ШИФРУВАННЯ:';
GPVP(Sender);
AssignFile(Fk,OpenDialog2.FileNAme);
ReSet(Fk);
AssignFile(Fv,SaveDialog2.FileNAme);
ReWrite(Fv);
i:=1;
while (not EOF(Fk))
do
begin
//////////////////////////////////////////////////////
// ОРИГІНАЛЬНЕ ШИФРУВАННЯ ПІД ТЕКСТОВИЙ ФАЙЛ:
{
Read(Fk,p);
if ord(з)-pvp[i]<0
then
Write(Fv,chr(ord(з)-pvp[((i-1) mod 1000)+1]+256))
else
Write(Fv,chr(ord(з)-pvp[((i-1) mod 1000)+1]));
}
// АДАПТАЦЫЯ РЕАЛІЗАЦІЇ
// ДЛЯ ВИКЛЮЧЕННЯ МОЖЛИВОЪ ПОЯАИ
// СИМВОЛУ З КОДОМ 26 (КІНЦЯ ФАЙЛУ)
Read(Fk,ss);
if ss-pvp[i]<0
then
Write(Fv,chr(ss-pvp[((i-1) mod 1000)+1]+256))
else
Write(Fv,chr(ss-pvp[((i-1) mod 1000)+1]));
//////////////////////////////////////////////
i:=i+1;
end;
CloseFile(Fk);
CloseFile(Fv);
edit1.Visible:=false;
label1.Visible:=true;
label1.font.Color:=clGREEN;
label1.caption:='ФАЙЛ ДЕШИФРОВАНО';
ENd;
ENd;


procedure TForm1.N32Click(Sender: TObject);
begin
Panel2Click(Sender);
label1.Visible:=true;
edit1.Visible:=true;
end;


procedure TForm1.Label2Click(Sender: TObject);
begin
N11Click(Sender);
```

```
end;


procedure TForm1.Label4Click(Sender: TObject);
begin
N21Click(Sender);
end;


procedure TForm1.Label3Click(Sender: TObject);
begin
N11Click(Sender);
end;


procedure TForm1.Label5Click(Sender: TObject);
begin
N21Click(Sender);
end;


procedure TForm1.Label7Click(Sender: TObject);
begin
N12Click(Sender);
end;


procedure TForm1.Label6Click(Sender: TObject);
begin
N12Click(Sender);
end;

procedure TForm1.Label8Click(Sender: TObject);
begin
N22Click(Sender);
end;


procedure TForm1.Label9Click(Sender: TObject);
begin
N22Click(Sender);
end;


procedure TForm1.N5Click(Sender: TObject);
begin
Panel2Click(Sender);
Image2Click(Sender);
end;

//                        ЗАХИСТ ВІД КОПЮВАННЯ
procedure TForm1.ZAHYST(Sender: TObject);
VAR
  VolumeName,
  FileSystemName : array [0..MAX_PATH-1] of Char;
  VolumeSerialNo,snr : DWord;
  MaxComponentLength,FileSystemFlags: Cardinal;
  reg:TRegistry;
  RDate:tDate;
  j:integer;
  RRR:STRING;
begin
//чтение номера диска
```

```
GetVolumeInformation('C:\',VolumeName,MAX_PATH,@VolumeSerialNo,
MaxComponentLength,FileSystemFlags, FileSystemName,MAX_PATH);
ID:=VolumeSerialNo;
reg:=TRegistry.Create;
reg.RootKey:=HKEY_LOCAL_MACHINE;
IF reg.OpenKey('\HardWare\SN',FALSE)
THEN
BEGIN
IDR:=REG.ReadString('\HardWare\SN');
IF IDR=INTTOSTR(ID)
THEN
A:=TRUE
ELSE
A:=FALSE;
END
ELSE
A:=FALSE;
end;

//                          ПРОПИСКА ЛІЦЕНЗІЇ
procedure TForm1.Button2Click(Sender: TObject);
VAR
  VolumeName,
  FileSystemName : array [0..MAX_PATH-1] of Char;
  VolumeSerialNo,snr : DWord;
  MaxComponentLength,FileSystemFlags: Cardinal;
  reg:TRegistry;
begin
//чтение номера диска
GetVolumeInformation('C:\',VolumeName,MAX_PATH,@VolumeSerialNo,
MaxComponentLength,FileSystemFlags, FileSystemName,MAX_PATH);
ID:=VolumeSerialNo;
reg:=TRegistry.Create;
reg.RootKey:=HKEY_LOCAL_MACHINE;
IF reg.OpenKey('\HardWare\SN',TRUE)
THEN
BEGIN
IDR:=REG.ReadString('\HardWare\SN');
IF (IDR=INTTOSTR(ID))
THEN  //
ELSE  REG.WRITEString('\HardWare\SN',INTTOSTR(ID));
END
ELSE
REG.WRITEString('\HardWare\SN',INTTOSTR(ID));
RichEdit1.Lines.Clear;
with RichEdit1.SelAttributes do
begin
Color:=clBlack;
Size:=16;
end;
RichEdit1.Lines.Add('');
RichEdit1.Lines.Add('ВСТАНОВЛЕНИЙ ID:   '+INTTOSTR(ID));
RichEdit1.Lines.Add('');
RichEdit1.Lines.Add('ДЛЯ КОРЕКТНОЇ РОБОТИ ПЕРЗАПУСТІТЬ ДОДАТОК');
RichEdit1.SETFOCUS;
BUTTON2.Visible:=FALSE;
N1.Enabled:=false;
N2.Enabled:=false;
N3.Enabled:=false;
end;

end.
```

```
// -------------------- Form1.dfm -------------------- Опис Форми Form1
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs;

type
  TForm1 = class(TForm)
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

object Form1: TForm1
  Left = 232
  Top = 169
  Width = 870
  Height = 450
  Caption = 'Form1'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Menu = MainMenu1
  OldCreateOrder = False
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  object Image1: TImage
    Left = 696
    Top = 8
    Width = 41
    Height = 41
    Center = True
    Proportional = True
    Stretch = True
    OnClick = Image1Click
  end
  object Image2: TImage
    Left = 696
    Top = 56
    Width = 41
    Height = 41
    Center = True
    Proportional = True
    Stretch = True
    OnClick = Image2Click
  end
  object Image3: TImage
    Left = 72
```

```
  Top = 192
  Width = 33
  Height = 33
  Center = True
  Proportional = True
  Stretch = True
  OnClick = Image3Click
end
object Label1: TLabel
  Left = 8
  Top = 200
  Width = 57
  Height = 24
  Caption = 'КЛЮЧ:'
  Color = clWhite
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -21
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentColor = False
  ParentFont = False
end
object Label2: TLabel
  Left = 432
  Top = 8
  Width = 129
  Height = 13
  Caption = 'ФАЙЛ, ЩО ШИРУСТЬСЯ'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentColor = False
  ParentFont = False
  OnClick = Label2Click
end
object Label3: TLabel
  Left = 208
  Top = 8
  Width = 165
  Height = 16
  Caption = 'ФАЙЛ, ЩО ШИФРУСТЬСЯ'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = 16
  Font.Name = 'MS Sans Serif'
  Font.Pitch = fpFixed
  Font.Style = []
  ParentFont = False
  OnClick = Label3Click
end
object Label5: TLabel
  Left = 208
  Top = 29
  Width = 165
  Height = 16
  Caption = 'ЗАШИФРОВАНИЙ ФАЙЛ  '
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = 16
  Font.Name = 'MS Sans Serif'
```

```
  Font.Style = []
  ParentFont = False
  OnClick = Label5Click
end
object Label4: TLabel
  Left = 432
  Top = 32
  Width = 132
  Height = 13
  Caption = 'ЗАШИФРОВАНИЙ ФАЛЙ'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  OnClick = Label4Click
end
object Label6: TLabel
  Left = 432
  Top = 56
  Width = 156
  Height = 13
  Caption = 'ФАЙЛ, ЩО ДЕШИФРУЄТЬСЯ'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  OnClick = Label6Click
end
object Label7: TLabel
  Left = 208
  Top = 56
  Width = 186
  Height = 16
  Caption = 'ФАЙЛ, ЩО ДЕШИФРУЄТЬСЯ'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = 16
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  OnClick = Label7Click
end
object Label8: TLabel
  Left = 432
  Top = 80
  Width = 116
  Height = 13
  Caption = 'ВІДТВОРЕНИЙ ФАЙЛ'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  OnClick = Label8Click
end
object Label9: TLabel
  Left = 216
  Top = 80
  Width = 141
```

```
  Height = 16
  Caption = 'ВІДТВОРЕНИЙ ФАЙЛ'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = 16
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  OnClick = Label9Click
end
object Panel1: TPanel
  Left = 8
  Top = 8
  Width = 185
  Height = 41
  Caption = 'ШИФРУВАННЯ'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 0
  OnClick = Panel1Click
end
object Panel2: TPanel
  Left = 8
  Top = 56
  Width = 185
  Height = 41
  Caption = 'ДЕШИФРУВАННЯ'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 1
  OnClick = Panel2Click
end
object Panel3: TPanel
  Left = 8
  Top = 104
  Width = 185
  Height = 41
  Caption = 'ДОВІДКА'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 2
  OnClick = Panel3Click
end
object RichEdit1: TRichEdit
  Left = 216
  Top = 104
  Width = 177
  Height = 49
  Lines.Strings = (
    'RichEdit1')
  TabOrder = 3
end
```

```
object Edit1: TEdit
  Left = 112
  Top = 192
  Width = 57
  Height = 21
  TabOrder = 4
  Text = 'Edit1'
end
object Button2: TButton
  Left = 304
  Top = 280
  Width = 321
  Height = 49
  Caption = 'ВСТАНОВИТИ ID'
  TabOrder = 5
  OnClick = Button2Click
end
object MainMenu1: TMainMenu
  BiDiMode = bdRightToLeft
  ParentBiDiMode = False
  Left = 16
  Top = 160
  object N1: TMenuItem
    Caption = 'Шифрування'
    object N11: TMenuItem
      Caption = 'Вибрати фаайл , що шифрусться'
      OnClick = N11Click
    end
    object N21: TMenuItem
      Caption = 'Ввести назву файлу для зберешення шифрування'
      OnClick = N21Click
    end
    object N31: TMenuItem
      Caption = 'Ввести ключ шифрування'
      OnClick = N31Click
    end
    object N4: TMenuItem
      Caption = 'Зашифрувати файл'
      OnClick = N4Click
    end
  end
  object N2: TMenuItem
    Caption = 'Дешифрування'
    object N12: TMenuItem
      Caption = 'Вибрати файл для дешифрування'
      OnClick = N12Click
    end
    object N22: TMenuItem
      Caption = 'Ввести назву для файлу , що відтворюється'
      OnClick = N22Click
    end
    object N32: TMenuItem
      Caption = 'Ввести ключ дешифрування'
      OnClick = N32Click
    end
    object N5: TMenuItem
      Caption = 'Дешифрувати файл'
      OnClick = N5Click
    end
  end
  object N3: TMenuItem
    Caption = 'Довідка'
    OnClick = N3Click
  end
```

```
  end
  object OpenDialog1: TOpenDialog
    Left = 592
    Top = 16
  end
  object SaveDialog1: TSaveDialog
    Left = 640
    Top = 16
  end
  object OpenDialog2: TOpenDialog
    Left = 592
    Top = 64
  end
  object SaveDialog2: TSaveDialog
    Left = 640
    Top = 64
  end
end
```