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
“___” _____ 2024 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти

на тему

“ Дослідження та програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів”

Виконав здобувач вищої освіти
ІІ курсу, групи КІ-23М
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Козак А.І.
«____» _____ 2024 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Кислун О.А.
«____» _____ 2024 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Рівень вищої освіти *магістр*

Галузь знань _12 _____*"Інформаційні технології"*_____

Спеціальність 123 *"Комп'ютерна інженерія"*

Освітньо-професійна (освітньо-наукова) програма _*"Комп'ютерна інженерія"*

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« » _____2024 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Козаку Артему Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів*

2. Керівник роботи *Кислун Олег Андрійович, канд. техн. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу №19-13 від 07.08.2024 року

3. Строк подання студентом роботи до захисту *02.12.2024 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є Дослідження та програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів*

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

1. Призначення та область використання.	*6. Наукова новизна.*
2.Перегляд аналогічних існуючих систем.	*7. Маркетингове та економічне обґрунтування ІТ проекту.*
3. Опис і обґрунтування проектних рішень.	*8. Заходи з охорони праці та техніки безпеки.*
4. Етапи програмування системи.	*9. Висновки.*
5. Впровадження системи в промислову експлуатацію	

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

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*3 аркуша*
Маркетингове та економічне обґрунтування ІТ проекту	*1 аркуш*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Доренська А.О.	05.10.2024	14.11.2024
Охорона праці	Марченко К.М.	06.10.2024	16.11.2024

7. Дата видачі завдання « » 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2024р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2024р.	
3.	Розробка моделі компонента	20.10.2024р.	
4.	Розробка структур даних	25.10.2024р.	
5.	Розробка алгоритмів зв’язку та відображення	30.10.2024р.	
6.	Програмування алгоритмів	10.11.2024р.	
7.	Маркетингове та економічне обґрунтування ІТ проекту	13.11.2024р.	
8.	Розрахунки з охорони праці та техніки безпеки	15.11.2024р.	
9.	Оформлення ПЗ	17.11.2024р.	
10.	Попередній захист роботи	02.12.2024р.	

Дата видачі завдання
« » 2024р.

Підпис керівника
_____ Кислун О.А.
(прізвище та ініціали)

Завдання прийнято до виконання
« » 2024 р.

Підпис здобувача
_____ Козак А.І.
(прізвище та ініціали)

АНОТАЦІЯ

Козак А.І. Дослідження та програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2024.

В даній магістерській роботі розроблено програмне забезпечення, яке призначено для системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів.

Метою роботи є Дослідження та програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів.

Об'єктом дослідження є процес захисту даних на основі генерації псевдовипадкових алгоритмів.

Предметом дослідження є методи захисту даних на основі генерації псевдовипадкових алгоритмів.

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

- Результат роботи – програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів.

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, захист даних, захист даних від несанкціонованого доступу.

ABSTRACT

Kozak A.I. Research and software implementation of a personal data protection system using pseudo-random algorithms. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2024.

This master's thesis developed software intended for a personal data protection system using pseudo-random algorithms.

The purpose of the work is Research and software implementation of a personal data protection system using pseudo-random algorithms.

The object of the research is the process of data protection based on the generation of pseudo-random algorithms.

The subject of the research is data protection methods based on the generation of pseudo-random algorithms.

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

- The result of the work is a software implementation of a personal data protection system using pseudo-random algorithms.

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 user-friendly interface has been developed. Instructions for working with the software are provided.

The program is designed to run under the Windows operating system.

The program was developed in the Delphi 7 environment.

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

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

АС	–	Автоматизована система
IT	–	Інформаційні технології
AEP Pro	–	Advanced Encryption Package Professional
RSA	–	Rivest, Shamir & Adleman
SHA	–	Secure Hash Algorithm
CAST	–	Алгоритм криптозахисту Carlisle Adams & Stafford Tavares
AES	–	Алгоритм шифрування Advanced Encryption Standard
Triple DES	–	Симетричний блочний шифр
Blowfish	–	Криптографічний алгоритм блочносеметричного шифрування
NTFS	–	Файлова система «New Technology File System»
RAD	–	Концепція створення засобів розробки rapid application development
SNOW	–	Алгоритм шифрування побудований на основі генератора підсумовування
MARS	–	Алгоритм шифрування
SERPENT	–	Симетричний блочний алгоритм шифрування
TwoFish	–	Симетричний блочний алгоритм шифрування
OAEP	–	Optimal Asymmetric Encryption Padding асиметричне шифрування з доповненням

ВСТУП

Актуальність теми. На даний час значно виріс обсяг інформації, що зберігається в електронному вигляді, а отже, зросли й можливості, щодо одержання доступу до неї. Роботи, відповідно до інформації, яку зберігають в електронному вигляді, ведуться за таких основних напрямків: доступність, цілісність та конфіденційність. За класифікацією інформація поділяється на відкриту, конфіденційну та таємну. А оскільки значна частина інформації не призначена для публічного перегляду або навіть становить таємницю, то завдання обмеження доступу до інформації, що зберігається в електронному вигляді, є та буде актуальним.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів.

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

- Огляд існуючих систем захисту.
- Дослідження системи захисту.
- Програмна реалізація системи захисту даних .

Об'єктом дослідження є процес захисту.

Предметом дослідження є методи захисту.

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

Наукова новизна отриманих результатів. У процесі вирішення завдань, обумовлених цілями дослідження, отримані наступні результати:

- Удосконалено метод захисту даних на основі генерації псевдовипадкових алгоритмів.

- Розроблено вітчизняний продукт захисту даних на основі генерації псевдовипадкових послідовностей, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі захисту даних на основі генерації псевдовипадкових послідовностей .

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведених у науковій літературі.

Робота апробована на Всеукраїнській науково-практичній on-line конференції “Проблеми енергоефективності та автоматизації в промисловості та сільському господарстві”, яка відбулася 13-14 листопада 2024 року у ЦНТУ м. Кропивницький.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів, є актуальним завданням, яке потребує вирішення у даній магістерській роботі.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

«Захист інформації - сукупність організаційно-технічних заходів і правових норм для запобігання заподіянню шкоди інтересам власника інформації чи АС та осіб, які користуються інформацією». Під автоматизованою системою (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].

1.1 Призначення системи захисту даних на основі генерації псевдовипадкових послідовностей

У самій назві систем зазначено її призначення - захист даних.

З технічної точки зору система призначена для перетворення відкритого тексту в шифротекст (зашифрування, кодування) та відновлення відкритого тексту із шифротексту (розшифрування, декодування) при відомих ключових даних.

Захист інформації в системі проводиться шляхом шифрування даних. Зашифрована інформація не може бути переглянута, спотворена або

пошкоджена іншими користувачами, колегами, адміністраторами чи зловмисниками-«хакерами», що одержали доступ до комп'ютера, мережі, каналу зв'язку тощо. До того ж, несанкціонований доступ до інформації сторонніх осіб не повинен бути можливий при ремонті, втраті, крадіжці обладнання.

Отже, з позиції: «кому?», система призначена фахівцям усіх напрямків, які у процесі виконання 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 приватним користувачем.

1.2 Область застосування

Система може застосовуватися практично в усіх галузях діяльності підприємств і фізичних осіб, де вимагається використовувати захист даних, включаючи:

- Органи державного управління, місцевого самоврядування - захист персональних даних, забезпечення безпечної пересилки конфіденційної інформації, організація юридично-важливого значимого електронного документообігу, у тому числі міжвідомчого.

- Промисловість, будівництво, торгівлю - захист персональних даних, забезпечення безпечної пересилки; організація збереження конфіденційності ділової інформації, у тому числі технічних, проектних, звітних документів, планів, договорів.

- Банки, фінанси - захищений інформаційний обіг із зовнішніми організаціями, фізичними особами.

- Медицина - захист персональних даних у процесі створення, заповнення, пересилки, обміну картками хворих між фахівцями-лікарями, а

також між різними лікувальними установами..

- Страхування - захист персональних даних при заповненні, пересилці страхових заяв, полісів, протоколів ДТП, документів по страхових випадках, звітів по роботі з клієнтами і виплатам по страхових випадках.

- Туризм - захист персональних даних при оформленні документів та збереженні конфіденційності юридично значимої електронної документації.

- Транспорт. Логістика - пересилка захищеної інформації.

- Юриспруденція - пересилка захищених документів.

- Наукові і науково-дослідні організації - пересилка результатів досліджень і випробувань, обмін результатами досліджень з іншими НДІ, архів документації та дисертацій, що мають обмежений доступ.

- Компанії по підбору персоналу - шифруванням обмін з працедавцями персональними даними про претендентів. Захист зберігання резюме.

- Видавництво/поліграфія - захист збереження рекламних макетів, інтерв'ю, текстів статей.

- Фізичні особи використання шифрування для обміну конфіденційною інформацією з кореспондентами. Захист шифруванням конфіденційних даних на персональних комп'ютерах.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи захисту даних на основі генерації псевдовипадкових послідовностей, є актуальною задачею, яка потребує вирішення у даній магістерській роботі.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

Програми для запобігання несанкціонованому доступу до конфіденційної інформації умовно можна розділити на три типи: програми, що шифрують інформацію; програми, що приховують інформацію; програми, що не шифрують інформацію, але блокують несанкціонований доступ або обмежують доступ до даних. На практиці багато програм можуть одночасно відноситися до різних типів. Наприклад, деякі програми, що не шифрують інформацію, але блокують несанкціонований доступ до неї, можуть також приховувати цю інформацію на жорсткому диску, щоб відповідні файли або теки не відображалися у провіднику.

Аналогом системі, що розробляється, є програми шифрування. Програми, що дозволяють зашифровувати інформацію, можна умовно розділити на два типи: програми, що реалізовують симетричне шифрування, тобто шифрування з використанням одного і того ж ключа для шифрування і розшифровки інформації; програми, що реалізовують асиметричне шифрування на основі пари ключів, один з яких, що називається публічним, або відкритим (public), застосовується для шифрування, а другий, такий, що називається секретним, або часткою (private), - для розшифровки [4, 5, 6].

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми магістерської роботи

Архіватори WinZip и WinRAR

Напевно, найпростіший спосіб забезпечення конфіденційності інформації - це використання архіваторів WinZip або WinRAR. Обидва архіватори застосовуються переважною більшістю користувачів, проте не всі знають, що, окрім можливості створювати архіви, ці програми дозволяють

зашифровувати їх вміст.

Архіватор WinZip припускає використання шифрування AES з 128 - або 256-бітовим ключем, а архіватор WinRAR - шифрування AES з 128-бітовим ключем.

В інтернеті можна знайти безліч різноманітних утиліт, призначених для підбору (чи злому - кому як більше подобається) паролів до RAR, - і Zip - архивам. Проте це зовсім не означає, що паролі до них легко підібрати. Шифрування AES з 128-бітовим ключем є дуже стійким, і підібрати пароль методом перебору неможливо (не кажучи вже про шифрування AES 256-біт). Єдино можливий спосіб підбору пароля - це атака по словнику. Тобто, якщо в якості пароля застосовується осмислене слово, яке є присутнім у словнику, то такий пароль підібрати дуже просто і часу для цього багато не буде потрібно. Як же створити стійкий пароль, який неможливо підібрати по словнику? Ми вже неодноразово писали про це, але нагадаємо ще раз. З одного боку, пароль повинен легко запам'ятовуватися, а з іншого - бути безглуздим (відносно граматики) набором символів. Найпростіше використати в якості пароля не одне слово, а фразу, записану без пропусків між словами. Ще краще, якщо фраза пишеться по-російськи, але при включеній англійській розкладці клавіатури. Тоді виходить слово, якого немає ні в одному словнику, але в той же час такий пароль легко запам'ятати. Наприклад, набір символів «Rhfcyfzfhvbzdct[cbkmytq» здається повною нісенітницею, але якщо зіставити латинські букви на клавіатурі з російськими, то вийде фраза «Червона армія усіх сильніше», записана без пропусків. Подібний пароль неможливо зламати ніякими утилітами, які використовують словники, що підключаються, або метод перебору [6, 7,8].

Advanced Encryption Package Professional

Advanced Encryption Package Professional (AEP Pro) від компанії SecureAction - одна із кращих у своєму класі програма для шифрування даних. Її вартість - всього 49,95 дол. На сайті виробника доступна для

скачування 30-денна демоверсія програми, яка, правда, має обмежену функціональність.

Програма AEP Pro підтримує декілька інтерфейсних мов, у тому числі й російську, так що освоїти роботу з нею не складе труднощів. Єдине, що не перекладене російською мовою, - це файл допомоги (Help). Втім, при роботі з програмою звертатися до нього навряд чи доведеться. Усе досить просто і логічно.

Отже, розглянемо коротко функціональні можливості AEP Pro. Ця програма дозволяє зашифровувати як окремі файли і теки, так і різні директорії. При цьому підтримуються усі сучасні алгоритми шифрування : DESX 128, Blowfish 448, Rijndael 256 (AES), CAST 256, Triple - DES 192, RC2 1024, Diamond 2 2048, Tea 128, Safer 128, 3 - Way 96, GOST 256, Shark 128, Square 128, Skipjack 80, Twofish 256, MARS (IBN) 448 Serpent 256.

При установці (підтримуються 32-бітові операційні системи сімейства Windows) програма автоматично інтегрується в контекстне меню. Тобто, виділивши будь-який файл або теку і клацнувши правою кнопкою миші, в контекстному меню можна вибрати новий пункт AEP, що дозволяє реалізувати швидкий доступ до основних функцій програми AEP Pro (зашифрувати, розшифрувати, знищити тощо).

Програма AEP Pro підтримує шифрування файлів на будь-якому типі носіїв, тобто файли, які необхідно зашифрувати або розшифрувати, можуть бути як на жорсткому диску, так і на USB флеш-носії, карті пам'яті, дискеті і так далі.

AEP Pro підтримує як симетричне, так і асиметричне шифрування. У разі симетричного шифрування необхідно задати пароль і вибрати один із 17 алгоритмів шифрування. Максимальна довжина пароля складає 56 символів, а при завданні пароля спеціальний індикатор PQ (Password Quality) покаже, наскільки надійним є введений вами пароль. У програмі є вбудований словник, що містить 45 тис. слів. Якщо використовуваний вами

пароль виявиться в нім, то програма видасть попередження, що пароль уразливий і легко може бути знайдений. Правда, зазначимо, що в цей словник входять тільки англійські слова.

При застосуванні асиметричного шифрування (RSA - шифрування використовуються два ключі: публічний і приватний. Публічний ключ доступний для усіх і застосовується тільки для шифрування файлів. Проте його принципово не можна використати для розшифровки файлів. Приватний ключ - секретний ключ, призначений для розшифрування.

У програмі AEP Pro є спеціальний менеджер ключів, який дозволяє створювати публічні і приватні (секретні) ключі. При цьому стійкість ключів (довжина ключа) може складати 512, 768, 1024 і 2048 біт. Приватний ключ додатково можна зашифрувати на основі пароля по одному з 17 передбачених алгоритмів. В цьому випадку для розшифровки файлів недостатньо мати тільки приватний ключ - треба знати ще і пароль до цього секретного ключа.

При шифруванні/розшифруванні файлів програма AEP Pro надає широкі можливості по налаштуванню. Так, можна задати теку, в якій створюватимуться зашифровані файли, при шифруванні або розшифруванні початкові файли можна відразу ж видаляти. Крім того, при роботі з великою кількістю файлів можна настроювати фільтр для зручності вибору файлів.

Ще однією цікавою і корисною особливістю програми є можливість створення архівів (SFX), що само розпаковуються. Ця функція використовується у тому випадку, коли треба зашифрувати дані, але при цьому передбачається, що їх розшифровуватимуть на комп'ютері, де програма AEP Pro не встановлена. В цьому випадку для розшифровки даних досить знати пароль.

Крім того, програма AEP Pro може застосовуватися для шифрування (розшифрування) поштових або будь-яких інших текстових повідомлень. У разі шифрування текстових повідомлень використовується алгоритм

шифрування AES 256 або асиметричне RSA - шифрування. При цьому необхідно, щоб програма AEP Pro була встановлена як у відправника повідомлень, так і у одержувача.

Ще однією чудовою особливістю програми AEP Pro є можливість безслідно знищувати файли, використовуючи для цього такі алгоритми, як U.S. DoD 5200.28, NATO Data Destruction Standard, U.S. Air Force Standard, U.S. Army Standard і Russian GOST P50739 - 95 Data Destruction Standard. Простіше кажучи, для знищення файлів застосовуються такі алгоритми, які не дозволяють ні за яких умов відновити знищений файл.

Ну і останнє, що хотілося б відмітити відносно програми AEP Pro, - ця наявність вбудованого Zip - архіватора, що дозволяє архівувати або розархівовувати файли [6, 9, 10].

CryptoExpert Professional

CryptoExpert Professional від компанії SecureAction- це програма для шифрування даних в режимі реального часу на комп'ютері.

Вона сумісна з операційними системами сімейства Windows, включаючи 32 - bit і 64 - bit версії.

Програма CryptoExpert Professional дозволяє створювати віртуальні логічні диски (контейнери), в яких інформація зберігається в зашифрованому виді. CryptoExpert існують в двох варіантах: CryptoExpert Lite і CryptoExpert Professional. Програма CryptoExpert Lite є безкоштовною і відрізняється від CryptoExpert Professional урізаною функціональністю. Зокрема, в CryptoExpert Lite розмір віртуального контейнера складає 50 Мбайт, тоді як в програмі CryptoExpert Professional він обмежений лише вільним простором на жорсткому диску. Ще одна відмінність між версіями полягає в тому, що в програмі CryptoExpert Lite використовується тільки один алгоритм шифрування - CAST 128 bit, а в програмі CryptoExpert Professional підтримуються чотири типи шифрування : AES (256 bit), CAST (128 bit), Blowfish (448 bit) і 3DES (168 bit). Є і інші відмінності, проте, щоб зрозуміти,

про що йде мова, необхідно навчитися працювати з програмою.

На жаль, опис до програми (файл допомоги) російською мовою відсутній, та і інтерфейс тільки англійський, до того ж його не можна назвати простим і інтуїтивно зрозумілим - перш ніж скористатися цією програмою, доведеться вивчити інструкцію.

Якщо коротко, то основні принципи роботи з програмою наступні. Спочатку створюються віртуальні контейнери, для кожного з яких встановлюється розмір, вибирається алгоритм шифрування і задається пароль. Контейнер може знаходитися в двох станах: підключеному і відключеному. З підключеним контейнером можна працювати як із звичайним логічним диском. Природно, що для підключення контейнера необхідно знати пароль. У програмі CryptoExpert Professional одночасно можна підключати необмежене число контейнерів, а в програмі CryptoExpert Lite підтримується тільки один підключений контейнер. Крім того, версія CryptoExpert Professional підтримує роботу з USB-носіями, а також можливість створювати і підключати контейнери, створені на загальних ресурсах локальної мережі.

Програма також дозволяє видаляти файли без можливості їх відновлення. Причому підтримується невідновне видалення файлів як з логічних контейнерів, так і із звичайних тек.

З недоліків програми CryptoExpert відмітимо нестабільність її в роботі. Ми тестували програму з операційною системою Windows Vista 32 - bit, і вона частенько зависала [6,10,11].

FineCrypt

Утиліта FineCrypt призначена для шифрування даних з метою їх подальшого безпечного зберігання на комп'ютері або передачі через інтернет. Вона підтримує тільки англійську мову і сумісна з усіма операційними системами сімейства Windows. Із сайту компанії можна викачати безкоштовну демоверсію програми, яка відрізняється від повної версії

урізаною функціональністю. На жаль, досконалий опис до програми типу «How to» відсутній (навіть англійською мовою), тому освоювати утиліту доведеться методом проб і помилок. Але нічого складного в цьому немає. Після інсталяції на ПК програма інтегрується в оболонку Windows і в контекстному меню з'являється відповідний пункт. Після цього досить виділити мишкою будь-який файл, клацнути по ньому правою кнопкою миші і, вибравши відповідний пункт меню, зашифрувати або розшифрувати файл.

Програма FineCrypt підтримує шифрування як окремих файлів, так і цілих тек. При цьому підтримується як симетричне шифрування з використанням пароля, так і шифрування за допомогою секретних ключів. Крім того, підтримується асиметричне шифрування на основі публічного і секретного ключів.

Для шифрування можна застосовувати наступні алгоритми: AES (256 bit), Blowfish (576 bit), CAST (256 bit), GOST (256 bit), Square (128 bit), Mars (448 bit), RC - 6 (2040 bit), Serpent (256 bit), TripleDES (192 bit) і Twofish (256 bit).

У разі симетричного шифрування при наборі пароля, який програмно перетвориться в ключ шифрування потрібної довжини, спеціальний індикатор нагадає про стійкість пароля (чим довше пароль, тим краще), що вводиться.

Якщо треба забезпечити найвищий рівень безпеки даних, то замість пароля рекомендується застосовувати секретний ключ. Програма FineCrypt дозволяє генерувати і зберігати секретні ключі шифрування. Крім того, при генерації секретного ключа користувач може повністю управляти вектором ініціалізації ключа (IV-вектор). Нагадаємо, що вектор ініціалізації не є секретною інформацією і використовується для реалізації блокового алгоритму шифрування. У програмі FineCrypt вектор ініціалізації ключа шифрування зберігається разом із ключем, а не в зашифрованому файлі.

Окрім генерування ключа і вектора ініціалізації ключа, програма FineCrypt дозволяє створювати ключ і вектор ініціалізації вручну.

Програма FineCrypt також підтримує використання асиметричного RSA - шифрування на основі публічного і секретного (частки) ключів. Нагадаємо, що асиметричне шифрування застосовується при необхідності передачі інформації в зашифрованому вигляді іншим користувачам. При цьому інформація шифрується за допомогою публічного ключа, а розшифрувати її можна, тільки маючи секретний ключ. Програма FineCrypt дозволяє генерувати пари ключів, публічний і секретний, а також посилати іншим користувачам (чи отримувати від них) публічні ключі. Для простоти управління секретним і публічним ключами при їх генерації здійснюється прив'язка до імені користувача і його поштової адреси. Крім того, секретний ключ захищається паролем. Згенерований публічний ключ можна відправити поштою іншому користувачеві.

Як і більшість програм, призначених для забезпечення конфіденційності даних, утиліта FineCrypt дозволяє не просто видаляти файли, а видаляти їх без можливості подальшого відновлення, відповідно до стандарту DoD 5200.28 - STD (стандарт Міністерства оборони США).

І на останнє, про що хотілося б згадати, - це можливість створення зашифрованих архівів, що саморозпаковуються. У цьому випадку розшифрувати дані можна навіть на комп'ютері, де програма FineCrypt не встановлена [6, 12, 13]

File Securer

Утиліта File Securer призначена для блокування несанкціонованого доступу до файлів і тек, що зберігаються на комп'ютері. Ця програма сумісна з 32-бітовими операційними системами сімейства Windows. На сайті виробника для скачування доступна 30-денна демоверсія програми. Вартість повнофункціональної версії залежить від кількості ліцензій.

При використанні програми на одному комп'ютері вартість ліцензії складає 39,95 дол.

Користуватися цією утилітою дуже просто. При інсталяції на

комп'ютер вона інтегрується в Windows Explorer, і в контекстному меню (виклик меню здійснюється правою кнопкою миші) з'являються два нові пункти: Lock with file Securer (заблокувати доступ) і Unlock with file Securer (розблоковувати доступ). Для діставання доступу до файлу необхідно вказати пароль. Налаштування програми (вказівка пароля) робиться при виклику головного вікна File Securer.

Зазначимо, що програма File Securer не робить шифрування даних - вона лише блокує доступ до них. Обдурити програму не вдасться. Наприклад, ми намагалися скопіювати закритий паролем файл з одного комп'ютера на інший, на якому теж була встановлена програма File Securer, але з іншим паролем доступу. Проте на іншому ПК розблоковувати доступ до файла не вдалося. Причому не допоміг ніякий пароль. Так що якщо доступ до файла був закритий на одному ПК, то розблоковувати цей файл на іншому ПК вже не вийде.

Крім того, не знаючи пароля, програму не можна деінсталювати з комп'ютера, а при деінсталяції програми з використанням пароля усі заблоковані файли автоматично розблоковуються [6, 14].

FileAssurity OpenPGP Lite

Утиліта FileAssurity OpenPGP Lite від компанії ARTICSOFT побудована на основі легендарної програми PGP (є її комерційним клоном) і призначена для безпечного обміну даними через інтернет між користувачами. Нагадаємо, що PGP має безліч реалізацій, сумісних між собою і з рядом інших програм завдяки стандарту OpenPGP.

Програма FileAssurity OpenPGP Lite дозволяє реалізувати асиметричне RSA - шифрування даних з використанням публічних і секретних (приватних) ключів. Крім того, передбачена можливість використання цифрового підпису при передачі даних. Звичайно, застосування асиметричного шифрування за допомогою публічного і секретного (частки) ключів зовсім не означає, що цю програму не можна

використати просто для шифрування даних з метою їх подальшого зберігання на комп'ютері або на іншому носії, проте відразу обмовимося, що це не найкращий варіант.

Отже, як підсумок, то програма FileAssurity OpenPGP Lite дозволяє шифрувати дані (файли і теки), підписувати дані (цифровий підпис), зберігати дані в зашифрованому виді й автоматично пересилати їх електронною поштою. При шифруванні дані автоматично стискаються (архівуються), і в цьому плані програма FileAssurity OpenPGP Lite подібна до архіваторів.

Функціональні можливості програми FileAssurity OpenPGP Lite відповідають вимогам стандарту OpenPGP.

Робота з програмою розпочинається з того, що користувач створює пару ключів : публічний і секретний. При генерації ключів за допомогою FileAssurity OpenPGP Lite задаються їх власник (ім'я й адреса електронної пошти), тип ключа, довжина ключа і термін його дії.

FileAssurity OpenPGP Lite підтримує два типи ключів: RSA (1024, 2048, 3072, 4096 bit) і Diffie - Hellman (DH)/DSS (1024/2048 bit). При цьому ключ електронного підпису в ключах Diffie - Hellman/DSS завжди має розмір 1024. Термін дії для кожного типу ключів може бути визначений як необмежений або до конкретної дати. Для захисту ключа задається секретна фраза (пароль).

Електронний цифровий підпис формується шляхом шифрування хеша повідомлення (файла) закритим ключем відправника (автора).

Для формування хэша в програмі FileAssurity OpenPGP Lite використовується алгоритм SHA - 1 (Secure Hash Algorithm). При цьому довжина хеша фіксована і складає 160 біт.

Для шифрування файлів застосовується алгоритм AES (256 bit), а для сумісності з попередніми версіями програми (PGP v.5.x і v.6.x) підтримуються алгоритми CAST і TDES. При розшифруванні файлів

підтримується також алгоритм Twofish.

Роботу з програмою FileAssurity OpenPGP Lite освоїти досить просто. Її інтерфейс інтуїтивно зрозумілий, проте підтримку російської мови не передбачено. У програмі є дуже детальна інструкція користувача (англійською мовою). Втім, ще раз відмітимо, що відсутність підтримки російської мови в даному випадку некритично.

Після запуску програми відкривається головне вікно, в якому відображається деревовидна структура файлів і тек. Виділивши будь-яку теку або файл, їх можна зашифрувати і підписати, а якщо виділяється зашифрований файл, то його можна розшифрувати (за наявності відповідного ключа). Ось, власне, і все. Окрім перерахованих операцій, програма FileAssurity OpenPGP Lite дозволяє виконувати безпечне видалення файлів (без можливості їх відновлення), а також управляти ключами (створювати резервні копії, експортувати, імпортувати тощо).

Залишилося додати, що програма сумісна з 32-бітовими операційними системами сімейства Windows і є платною. Із сайту виробника можна викачати лише 15-денну демоверсію програми. Повнофункціональна версія програми коштує 65 долл [6].

Max File Encryption

Max File Encryption це невелика, проте платна утиліта, призначена для шифрування окремих файлів. Існує демоверсія програми, яка відрізняється від повнофункціональної урізаними можливостями. Вартість повнофункціональної версії програми складає 30 дол.

Програма має англійський інтерфейс, однак робота з нею не викликає складнощів. Після запуску утиліти відкривається головне вікно, розділене на дві частини. У лівій частині відображається деревовидна структура усіх директорій. Файли з лівої частини вікна можна перетягувати мишею в праву, після чого їх можна шифрувати або, навпаки, розшифровувати. Після закінчення шифрування файлів програма пропонує видалити початкові

файли.

Власне, на цьому усі особливості роботи з програмою закінчуються. Усе дуже просто, але в той же час можливостей досить мало.

Одним із недоліків програми є неможливість шифрувати теки і директорії. Якщо перетягнути мишачу теку з лівої частини вікна в праву, то там відобразяться усі файли, що містяться в цій теці, які шифруватимуться окремо.

До позитивних моментів варто віднести можливість створення зашифрованих архівів, що саморозпаковуються.

Згідно з технічної інформацією, програма Max File Encryption підтримує тільки один алгоритм шифрування - Blowfish.

Ще одна цікава функція цієї утиліти - це підтримка стеганографії, тобто можливості ховати файл усередині інших файлів (контейнерів). Контейнером може виступати відеофайл, музичний файл або файл цифрової фотографи [6, 15, 16].

Dekart Private Disk

Програма Dekart Private Disk від компанії Dekart - це розробка молдавських програмістів. Вона сумісна з операційними системами сімейства Windows.

Вартість програми Dekart Private Disk складає 45 дол., а на сайті виробника можна викачати її ознайомлювальну повнофункціональну 30-денну версію. Зазначимо, що із сайту виробника можна завантажити програму з російськомовним інтерфейсом, причому російською мовою написана і детальна інструкція з її використання.

Отже, програма Dekart Private Disk призначена для шифрування інформації з метою її безпечного зберігання на комп'ютері або на знімних носіях. Вона дозволяє створювати віртуальні логічні диски (контейнери), в яких інформація зберігається в зашифрованому виді. З віртуальними логічними дисками можна працювати точно так, як і із звичайними.

Віртуальний зашифрований диск є звичайним файлом - так званим файлом-образом диска. Файл-образ віртуального зашифрованого диска може мати будь-кого ім'я, розширення і шлях доступу (навіть мережевий).

Робота з програмою розпочинається із створення віртуального контейнера, для якого необхідно вказати розмір і місце розташування файла-образу, а також задати пароль доступу. Мінімальний розмір віртуального диска - 1 Мбайт, а максимальний - 1 Тбайт (для ОС Windows).

Кожен створений контейнер може знаходитися у двох станах: підключеному (змонтованому) і відключеному (демонтованому). При змонтованому стані контейнера з ним можна працювати як із звичайним логічним диском. Звичайно, для підключення контейнера необхідно знати пароль.

Після того, як віртуальний контейнер створений і змонтований, зашифрувати дані дуже просто - потрібно лише перенести їх на новий логічний диск. У програмі Dekart Private Disk використовується алгоритм шифрування AES з довжиною ключа 256 біт.

Треба відзначити, що програма Dekart Private Disk має дуже широкі можливості по налаштуванню, а також відрізняється різноманітними додатковими функціями.

Серед можливостей по налаштуванню відмітимо призначення гарячих клавіш і іконок, створення списку програм, яким дозволений доступ до віртуального диска, і списку програм, які автоматично запускаються при підключенні (монтуванні) і відключенні (розмонтуванні) віртуального диска.

Крім того, в програмі Dekart Private Disk передбачена можливість створення резервної копії віртуального диска і зашифрованої резервної копії ключа шифрування. Можна навіть спробувати відновити забутий пароль до віртуального диска, використовуючи для цього метод перебору можливих комбінацій символів пароля.

При спробі відновлення пароля можливе завдання набору символів і

вказівка довжини пароля. Хоча, звичайно, якщо пароль забутий, то намагатися відновити його - справа безнадійна, тому ця функція швидше демонструє надійність захисту, ніж має практичне значення [6, 17, 18].

Secure IT

Secure IT - це ще одна утиліта, призначена для шифрування файлів, тек і директорій для їх безпечного зберігання. Вона має англійський інтерфейс, проте працювати з нею настільки просто, що це навряд чи можна вважати недоліком. Щоб освоїти роботу з програмою, не знадобиться навіть вивчати інструкцію користувача. Усе дуже просто: після запуску програми в головному вікні відображається аналог провідника (Windows Explorer). У лівій частині провідника видається деревовидна структура усіх директорій. Якщо виділити будь-яку директорію в лівому вікні, то в правому вікні відобразиться її вміст. Будь-яку теку, файл або директорію, яка виділяється в правому вікні провідника, можна зашифрувати або, навпаки, розшифрувати. Наприклад, щоб зашифрувати теку, розташовану в кореневій директорії З :, необхідно виділити кореневу директорію в лівому вікні і потрібну теку в правому вікні провідника. Після цього можна зашифрувати цю теку, натиснувши на відповідну іконку на панелі інструментів.

Відмітимо, що програма Secure IT підтримує виділення відразу декількох тек, які необхідно зашифрувати. Крім того, вона дозволяє створювати зашифровані архіви, що саморозпаковуються, які можна розшифрувати на будь-якому комп'ютері без використання утиліти Secure IT.

І остання функціональна можливість Secure IT - це безпечне (без можливості відновлення) видалення (затирання) даних. При цьому програму можна настроїти так, щоб вона автоматично пропонувала затирати початкові дані кожного разу після їх шифрування.

Для шифрування даних програма Secure IT дозволяє вибрати один із двох алгоритмів шифрування: AES (256 bit) або Blowfish (448 bit), а при завданні пароля також можна зробити для себе підказку, щоб не забути

пароль надалі. Крім того, при шифруванні даних можна задати міру стискування архіву.

На закінчення додамо, що утиліта Secure IT є платною, але із сайту виробника є її 30-денна повнофункціональна демоверсія. Вартість програми складає 29,95 долл [6].

Universal Shield

Universal Shield - цей потужний засіб захисту конфіденційної інформації. Ця програма дозволяє приховувати файли, теки або диски цілком, шифрувати теки і файли, встановлювати режим доступу «Тільки читання», а також обмежувати доступ до тек на рівні користувачів. Утиліта підтримує декілька мов, проте російської в їх списку немає, так що доведеться задовольнятися англійським інтерфейсом.

Робота з утилітою Universal Shield досить проста, хоча назвати інтерфейс інтуїтивно зрозумілим не можна.

При інсталяції Universal Shield на ПК користувачеві пропонується задати пароль доступу до програми. Оптимальний варіант роботи з утилітою - це використання вбудованого майстра (Wizard). Після запуску майстра користувачеві пропонується вибрати одну з чотирьох дій : заховати файл, теку або диск, встановити режим «Тільки читання», зашифрувати файл (теку), встановити права доступу на призначеному ПК для користувача.

Далі вказується тека або файл, над яким необхідно виконати вибрану дію. Якщо передбачається зашифрувати файл або теку, то додатково можна вибрати алгоритм шифрування. Програма Universal Shield підтримує сім алгоритмів шифрування : Blowfish (448 bit), CAST (128 bit), Cobra128 (576 bit), PC1 (160 bit), AES (256 bit), Serpent (256 bit) і Triple - DES (168 bit).

Після вибору алгоритму шифрування і введенні пароля можна безпосередньо приступити до шифрування. Після шифрування майстер запропонує додатково приховати зашифровану теку. Статус зашифрованої

теки можна змінити у будь-який момент. Тобто можна зробити так, щоб зашифрована тека існувала нарівні з незашифрованою (режим повного доступу), щоб тека відображалася, але до неї був блокований доступ і т.д.

Взагалі, треба відмітити, що можливості програми Universal Shield дуже широкі - це один із кращих засобів для обмеження доступу до інформації.

В кінці додамо, що утиліта Universal Shield сумісна з операційними системами сімейства Windows і є платною, але на сайті виробника є 30-денна повнофункціональна демоверсію програми. Вартість програми складає 34,95 дол. [6, 19, 20].

File Encryption

Програма File Encryption XP по функціональних можливостях і навіть інтерфейсі багато в чому схожа із вже розглянутою утилітою Secure IT. Хоча звинувачувати когось в плагіаті в даному випадку навряд чи варто. Власне, придумувати що-небудь оригінальне і незвичайне в даному випадку немає сенсу, а тому більшість утиліт, призначених для шифрування інформації, дуже схожа одна на одну. Отже, утиліта File Encryption XP має англійський інтерфейс, але освоїти її дуже просто - для цього навіть не доведеться заглядати в інструкцію користувача. Після запуску програми в головному вікні відображається аналог провідника (Windows Explorer). У лівій частині провідника видається деревовидна структура усіх директорій. Якщо виділити будь-яку директорію в лівому вікні, то в правому вікні відобразиться її вміст. Будь-яку виділену в правому вікні провідника теку, файл або директорію можна зашифрувати або, навпаки, розшифрувати.

Програма File Encryption XP дозволяє створювати зашифровані архіви, що саморозпаковуються, які можна розшифрувати на будь-якому комп'ютері без використання утиліти.

І остання функціональна можливість програми File Encryption XP - це затирання файлів, тобто видалення їх без можливості відновлення. Програму

можна настроїти так, щоб вона пропонувала затирати початкові дані кожного разу після їх шифрування.

Шифрування даних у програмі File Encryption XP робиться по алгоритму Blowfish з довжиною ключа 448 біт. При введенні пароля можна скористатися вбудованим генератором пароля з можливістю вибору набору символів.

Програма File Encryption XP інтегрується в оболонку Windows, і відповідний пункт з'являється в контекстному меню. Після цього досить виділити мишкою будь-який файл, клацнути по ньому правою кнопкою миші і, вибравши відповідний пункт меню, зашифрувати або безповоротно видалити його.

На закінчення додамо, що утиліта File Encryption XP є платною, але є 30-денна повнофункціональна демоверсія програми. Вартість File Encryption XP складає 29,95 долл [6].

Steganos Safe

Steganos Safe потужний пакет, призначений для захисту конфіденційної інформації шляхом переміщення її у віртуальний логічний контейнер, де уся інформація зберігається в зашифрованому вигляді.

Програма Steganos Safe є платною. Її вартість складає 49,95 дол. Наявна 30-денна демоверсія програми. У процесі інсталяції утиліти потрібно підключення до Інтернету для завантаження сертифіката.

Програма має англійський інтерфейс і не підтримує російську мову.

При запуску утиліти користувачеві пропонується створити новий захищений диск (віртуальний контейнер) або підключитися до вже існуючого. При створенні нового захищеного диска йому привласнюються буква, назва і розмір. Максимальний розмір віртуального контейнера дорівнює 256 Гбайт (за умови використання файлової системи NTFS). Кількість самих віртуальних контейнерів не обмежена.

На останньому етапі задається пароль доступу до віртуального

контейнера. При цьому автоматично проводиться його перевірка на предмет наявності в словнику, що гарантує неможливість підбору пароля методом атаки по словнику. Крім того, можна згенерувати пароль, вказавши його довжину і набір використовуваних для пароля символів.

(Ще одна цікава можливість при введенні пароля, що надається програмою Steganos Safe, полягає в тому, щоб замість комбінації символів використати послідовність фотографій або картинок. При цьому треба просто запам'ятати послідовність цих картинок або фотографій).

Крім того, програма Steganos Safe дозволяє зберігати пароль на зовнішньому носії (USB флеш-носії тощо).

Після створення віртуального контейнера робота з ним відбувається точно так, як і із звичайним логічним диском. Але, перш ніж отримати доступ до захищеного диска, його необхідно підключити. Якщо диск не підключений, то він не відображається в провіднику і сторонній користувач може взагалі не здогадатися про його наявність.

Відзначимо, що усі дані на захищеному диску шифруються за допомогою алгоритму AES з довжиною ключа 256 біт.

Від розміру віртуального контейнера залежить час його створення. Чим більший розмір контейнера, тим більше часу піде і на його створення, і на його відкриття.

Швидкість копіювання даних на захищений диск невисока і складає приблизно 7 Мбайт/с. Зазначимо, що операція копіювання даних критична до об'єму оперативної пам'яті. Наприклад, ми копіювали на захищений диск файл розміром 900 Мбайт, при цьому об'єм оперативної пам'яті, задіяної в процесі копіювання, досяг 1,65 Гбайт.

Відзначимо, що в пакет Steganos Safe також входить той файл, що «знищує» дані, який дозволяє не просто видаляти дані, але і затирати їх без можливості відновлення.

Ще однією цікавою функцією програми Steganos Safe є гнучке

налаштування захищеного диска вже після того, як він створений. Зокрема, можна змінити пароль, назву і навіть букву захищеного диска. Окрім цього, можна змінити місце розташування віртуального диска, а також його розмір.

Крім того, програму Steganos Safe можна використати для шифрування усіх даних електронної пошти (кореспонденції, що входить, теки контактів і так далі). [6, 21 , 22, 23].

File and Folder Protector

File & Folder Protector - ще одна платна утиліта, призначена для приховання і обмеження доступу до конфіденційної інформації, що зберігається на ПК. Її вартість складає 49 дол., але існує ознайомлювальна версія програми.

Утиліта File & Folder Protector сумісна з усіма операційними системами сімейства Windows.

Програма має англійський інтерфейс, але користуватися нею досить просто. Головне вікно утиліти File & Folder Protector розбито на три частини (вікна). У верхньому лівому вікні відображається деревовидна структура усіх директорій і тек, у верхньому правому вікні - файли, що містяться усередині виділеної теки або директорії.

Теки й окремі файли можна переносити (перетягувати мишею) в нижнє вікно. Якщо файл або тека переміщені в нижнє вікно, то над ними можна виконувати наступні дії: приховувати і робити недоступними, приховувати імена тек або файлів, але робити їх доступними, встановлювати доступ до вмісту по паролю або режим «Тільки читання».

У режимі приховання без можливості доступу файли і теки не відображаються в провіднику, і жодна програма не може отримати до них доступ. У режимі приховання з можливістю доступу імена файлів і тек не відображаються в провіднику, проте доступ до них не блокується, і, використовуючи ім'я файла, його можна отримати у будь-якій програмі.

У режимі доступу по паролю файл або тека залишаються видимими,

але для доступу до їх вмісту вимагається ввести пароль. При цьому можливий режим введення пароля з використанням контекстного меню або за допомогою спливаючого діалогового вікна.

Власне, це усе премудрості роботи з утилітою File & Folder Protector. Зазначимо, що утиліта File & Folder Protector не шифрує дані, а тільки ховає їх і обмежує до них доступ [6].

Folder Guard

Програма Folder Guard - потужний пакет, призначений для приховання і обмеження доступу до інформації, що зберігається на ПК.

Програма Folder Guard існує в трьох варіантах: Folder Guard x32 Edition, Folder Guard x64 Edition і Folder Guard Professional Edition. Версія Folder Guard x32 Edition призначена для 32-бітових операційних систем сімейства Windows, версія Folder Guard x64 Edition - для 62-бітових операційних систем, а версія Folder Guard Professional Edition підтримує як 32-, так і 64-бітові операційні системи.

Існує 30-денна ознайомлювальна версія утиліти, та все ж ця утиліта є платною. Вартість версій Folder Guard x32 Edition і Folder Guard x64 Edition складає 39,95 дол., а версії Folder Guard Professional Edition - 59,95 дол.

Ця програма підтримує російський інтерфейс, так що проблем із освоєнням пакету не повинно бути. Правда, відмітимо, що, незважаючи на наявність російськомовного інтерфейсу, посібник користувача програми на російський не перекладений.

Як вже відзначалося, утиліта Folder Guard не зашифровує інформацію, а лише дозволяє приховувати теки і файли й обмежує доступ до них.

За допомогою Folder Guard можна зробити будь-який файл, теку або директорію невидимої або створити видимість того, що вони порожні. Можна заблокувати доступ до теки, директорії або файла по паролю або зробити так, щоб його можна було розблоковувати тільки при вході в програму Folder Guard з використанням майстер-пароля.

Програма має вбудованого майстра по налаштуванню, за допомогою якого можна швидко настроїти необхідний тип захисту для будь-якої теки, файла або директорії.

Окрім традиційних можливостей обмеження доступу до файлів і тек, програма Folder Guard дозволяє гнучко налаштовувати дозвіл на виконання тих або інших дій на комп'ютері. Наприклад, можна заблокувати використання панелі управління або можливість додавати або видаляти принтери тощо.

Незважаючи на наявність вбудованого майстра по налаштуванню, не можна сказати, що робота з програмою Folder Guard проста і не викликає питань. Попрацювавши один день із цією програмою, ми для себе зробили висновок, що при необхідності приховання даних на ПК краще застосовувати простіший в експлуатації софт [6, 24, 25, 26].

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Для побудови системи захисту даних на основі генерації псевдовипадкових послідовностей, базуючись на наявності в якості засобу розробки, обрано Delphi версії 7.

Система програмування Delphi версії 7 фірми Enterprise (Borland) надає широкі можливості для програмування додатків ОС Windows. Delphi - це продукт Borland International для швидкого створення додатків. Цей високопродуктивний інструмент візуальної побудови додатків включає справжній компілятор коду і надає засоби візуального програмування, дещо схожі на ті, які можна виявити в Microsoft Visual Basic або в інших інструментах візуального проектування.

В основі Delphi лежить мова Object Pascal, яка є розширенням об'єктно-орієнтованої мови Pascal.

До Delphi також входять локальний SQL-сервер, генератори звітів,

бібліотеки візуальних компонентів, і інше, необхідне для того, щоб почувати себе абсолютно упевненим при професійній розробці інформаційних систем або просто програм для Windows середовища.

Сам Delphi призначений для професійних розробників, бажаючих дуже швидко розробляти додатки в архітектурі клієнт-сервер. Delphi робить невеликі за розмірами високоефективні виконувані модулі (.exe і .dll), тому в Delphi мають бути, передусім, зацікавлені ті, хто розробляє продукти на продаж. З іншого боку, невеликі за розмірами і швидко виконувані модулі означають, що вимоги до клієнтських робочих місць істотно знижуються - це має важливе значення і для кінцевих користувачів.

Переваги Delphi у порівнянні з аналогічними програмними продуктами:

- швидкість розробки додатка (RAD);
- висока продуктивність розробленого застосування;
- низькі вимоги розробленого додатка до ресурсів комп'ютера;
- можливість розробки нових компонентів і інструментів власними засобами Delphi (існуючі компоненти й інструменти доступні в початкових кодах);
- вдале опрацювання ієрархії об'єктів.

Система програмування Delphi розрахована на програмування різних застосувань і надає велику кількість компонентів для цього. До того ж працедавців цікавить, передусім, швидкість і якість створення програм, а ці характеристики може забезпечити тільки середовище візуального проектування, здатне взяти на себе значні об'єми рутинної роботи з підготовки додатків, а також погоджувати діяльність групи постановників, кодувальників, тестерів і технічних письменників. Можливості Delphi повністю відповідають подібним вимогам і підходять для створення систем будь-якої складності [27, 28, 29, 30, 31, 32].

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на магістерську роботу, реалізації підлягає програмне забезпечення, яке призначено для системи захисту даних на основі генерації псевдовипадкових послідовностей.

У процесі розробки магістерської 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 контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

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

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

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

е) провести розрахунки з визначення економічної ефективності розробленої системи;

ж) розробити заходи з охорони праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

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

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

В основі всіх сучасних технологій захисту даних лежать криптографічні методи забезпечення конфіденційності інформації, в основному методи шифрування. Під шифруванням розуміється процес перетворення відкритої інформації в зашифровану інформацію (шифрований текст - шифротекст) або процес зворотного перетворення зашифрованої інформації у відкриту. Перетворення відкритої інформації в закриту отримало назву шифрування, а перетворення зашифрованої інформації у відкриту - розшифровування. Процес шифрування полягає в проведенні оборотних математичних, логічних, комбінаторних та інших перетворень вихідної інформації, в результаті яких зашифрована інформація являє собою хаотичний набір букв, цифр, інших символів і двійкових кодів. Для шифрування інформації використовуються криптографічний алгоритм і ключ. Як правило, алгоритм для певного методу шифрування є незмінним. Вихідними даними для алгоритму шифрування служать відкрита інформація і ключ шифрування. Ключ містить керуючу інформацію (двійковий код), яка визначає вибір перетворення на певних кроках алгоритму і величини операндів, використовувані при реалізації алгоритму шифрування. Іншими словами, ключ забезпечує вибір одного з можливих шляхів реалізації алгоритму шифрування [33,34,35].

У загальному випадку алгоритм шифрування може відрізнятися від алгоритму розшифрування. Відповідно, можуть розрізнятися ключі шифрування і розшифрування. Пара алгоритмів шифрування і розшифрування називається криптосистемою або шторосистемою. Процедуру шифрування і розшифрування можна представити в наступному вигляді [35,36]:

Позначимо відкритий текст (повідомлення) як M. Це може бути потік бітів, текстовий файл, бітове зображення, оцифрований звук, цифрове відеозображення й ін. Далі розглядатимуться тільки двійкові дані й комп'ютерна криптографія. Позначимо шифротекст як C. Це теж двійкові дані, іноді того ж розміру, що й M, іноді більші. (Якщо шифрування супроводжується стисненням, C може бути менше M. Однак саме шифрування не забезпечує стиснення інформації.) Функція шифрування E діє на відкритий текст, створюючи шифротекст [36]:

$$E(M) = C. \tag{3.11}$$

Процес відновлення відкритого тексту по шифротексту є розшифруванням і виконується за допомогою функції розшифрування [36]:

$$D:D(C) = M. \tag{3.2}$$

Оскільки змістом шифрування й наступного розшифрування повідомлення є відновлення первісного відкритого тексту, то має виконуватися тотожність [36]:

$$D(E(M)) = M. \tag{3.3}$$

Криптографічний алгоритм, також називаний шифром, являє собою математичну функцію, яка використовується для шифрування й розшифрування. Якщо безпека алгоритму заснована на збереженні самого алгоритму в таємниці, це обмежений алгоритм. Обмежені алгоритми являють тільки історичний інтерес, але вони зовсім не відповідають сьогоднішнім стандартам. Велика або непостійна група користувачів не може використати такі алгоритми, оскільки, коли користувач залишає групу, її члени мають переходити на інший алгоритм. Алгоритм також має бути замінений, якщо хто-небудь ззовні випадково довідається про секрет [36].

Сучасна криптографія розв'язує проблеми обмежених алгоритмів за допомогою ключа K. Ключ - це конкретний секретний стан певних параметрів алгоритму криптографічного перетворення даних, що забезпечує

вибір тільки одного варіанта перетворення з усіх можливих для даного алгоритму. Множину можливих ключів називають простором ключів. Ключ, що використовується для ініціалізації системи, часто називають майстер-ключем системи [36].

З урахуванням використання ключа, функції шифрування й розшифрування запишуться як [36]:

$$C = EK(C) \tag{3.4}$$

і

$$DK(C) = M, \tag{3.5}$$

при цьому має виконуватися тотожність:

$$DK(EK(M)) = M, \tag{3.6}$$

Якщо для деяких алгоритмів при шифруванні й розшифруванні використовуються різні ключі [36]:

$$C = EK1(M), \tag{3.7}$$

$$DK2(C) = M, \tag{3.8}$$

$$DK2(EK1(M)) \equiv M. \tag{3.9}$$

Якщо алгоритм перетворення даних залежить від ключа, тобто застосовуються управляючі операції, шифр називається шифром, що управляється. Криптографічні системи, у загальному випадку, класифікуються на основі таких трьох незалежних характеристиках: тип операцій з перетворення відкритого тексту в шифрований; число ключів, що використовуються; метод обробки відкритого тексту [36].

Якщо і відправник, і одержувач інформації використовують той самий ключ, система називається симетричною, системою з одним ключем, системою із секретним ключем або схемою традиційного шифрування. Якщо відправник і одержувач використають різні ключі (один відкритий, а інший

секретний), система називається асиметричною, системою із двома ключами або схемою шифрування з відкритим ключем [36].

Блочне шифрування передбачає обробку відкритого тексту блоками, так що в результаті обробки кожного блоку виходить блок шифрованого тексту. При потоковому шифруванні шифрування всіх елементів відкритого тексту здійснюється послідовно, одне за іншим, у результаті чого на кожному етапі отримують по одному елементу шифрованого тексту [36].

До шифрів, які використовуються для криптографічного захисту інформації, висувають низку вимог [36]:

- статистична безпека алгоритмів;
- надійність математичної бази алгоритмів;
- простота процедур шифрування й розшифрування;
- незначна надмірність інформації за рахунок шифрування;
- простота реалізації алгоритмів на різній апаратній базі.

Тією чи іншою мірою цим вимогам відповідають:

- шифри перестановок;
- шифри заміни;
- шифри гамування;
- шифри, засновані на аналітичних перетвореннях даних.

Основним питанням аналізу будь-якої криптографічної системи захисту інформації є визначення ступеня її стійкості. Стійкість криптографічної системи захисту інформації є її здатність протистояти атакам порушника на інформацію, що захищається. Під час оцінювання стійкості довільних криптографічних систем захисту інформації зазвичай дотримуються принципу Керкхофа: стійкість криптосистеми має бути забезпечена навіть тоді, коли порушникові відомий її повний опис. Тому в процесі аналізу стійкості криптосистем передбачається, що порушникові відомий детальний опис системи, статистичні характеристики алфавіту

повідомлення, простір можливих ключів і криптограм, контекст повідомлення тощо [36].

Сучасні алгоритми, що забезпечують захист інформації:

Симетричні:

- Rijndael (AES) з довжиною ключа в 128-256 bit J. Daemon, V.Rijmen, Бельгія;
- SNOW з довжиною ключа в 128, 256 bit Lund University, Швеція;
- RC6 з довжиною ключа в 128-256 bit RSA Security, США;
- 3DES з довжиною ключа в 168 bit Стандарт ANSI X9.52-1998;
- MARS з довжиною ключа в 128-400 bit IBM Corporation, США;
- TwoFish з довжиною ключа в 128-256 bit B. Schneir, США;
- SERPENT з довжиною ключа в 128-256 bit R. Anderson, E. Biham, L. Knudsen;
- ГОСТ 28147-89 з довжиною ключа в 256 bit Держстандарт СРСР. В Україні стандарт гармонізовано (ДСТУ 28147:2009).

Асиметричні:

- RSA з довжиною ключа в 1024-4096 bit RSA Laboratories, США;
- RSA-OAEP з довжиною ключа в 1024-4096 bit RSA Laboratories Europe, Швеція;
- ACE Encrypt з довжиною ключа в 1024-4096 bit IBM Zurich Research Laboratory, Швейцарія;
- EPOC з довжиною ключа в 1024-4096 bit Nippon Telegraph and Telephone, Японія.

Ці криптоалгоритми здатні забезпечити захист від: диференційного криптоаналізу; пошуку найкращої диференційної характеристики; лінійного криптоаналізу; інтерполяційного вторгнення; вторгнення із частковим угадуванням ключа; вторгнень на основі обробки апаратних помилок; пошуку лазівок [36].

3.1 Опис функціонування системи

По суті, робота системи захисту даних на основі генерації псевдовипадкових послідовностей зводиться до шифрування, тобто кодування визначеної послідовності (файла) для зберігання або передачі з подальшим відтворенням - розшифруванням (декодуванням). Послідовність кодується шляхом змішуванням з деякою псевдовипадковою послідовністю , що являє собою результат певного перетворення деякого ключа. Декодування проводиться шляхом виділення зашифрованої послідовності з суміші останньої та псевдовипадкової послідовності.

Виходячи із зазначеного, система, що розробляється, має виконувати наступні операції:

- запуск програми та автентифіка́ція;
- вибір режиму роботи програми: шифрування інформації або дешифрування коду:
- для режимів шифрування та дешифрування, вибір методу шифрування , якщо такий наявний:
- для режимів шифрування та дешифрування, введення первинного ключа;
- для режимів шифрування та дешифрування, одержання шляхом перетворення первинного ключа псевдовипадкової послідовності для змішування;
- для режиму шифрування, вибір файла для кодування;
- для режиму дешифрування, вибір закодованого файла;
- дії для налагодження інтерфейсу програми: вибір кольорів, фонтів тощо, реалізована довідкова система з роботи для програмного забезпечення, що розробляється.

3.2 Розробка структурної схеми

Розгляд структури програми можливий з двох позицій: розгляд, як структурних одиниць програми самої Delphi або як описання зв'язків між різними структурними одиницями програми (структурні одиниці програми розглядаються в якості окремих частини - блоків,. що виконують різні функції) та проходженням інформаційних потоків через них. Будь-яка програма Delphi складається з файла 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. [37]. Тож для побудови програми скористаємося можливістю мінімального використання модулів - один модуль. Розглянемо структурну програму через призму виконання її з боку функції. Структурна схема системи зображена на рисунку 3.1.

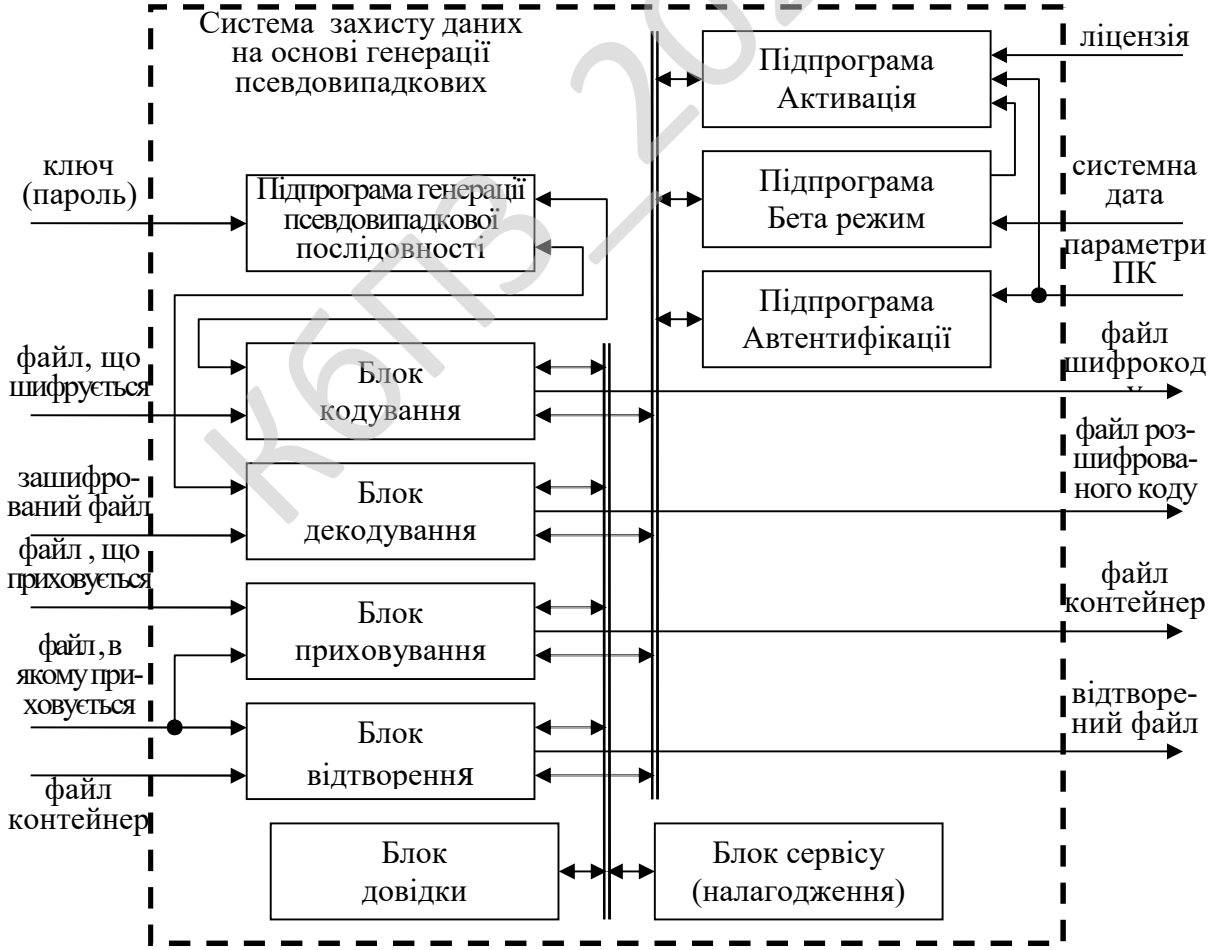

Рисунок 3.1 - Структурна схема

Зі схеми можна побачити, що до програми поступають наступні інформаційні потоки:

- ключ (пароль) - текстова послідовність, з якої генерується псевдовипадкова послідовність для режимів кодування та декодування;

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

- зашифрований файл - вхідні дані для режиму декодування;

- файл, що приховується - вхідні дані для режиму приховування;

- контейнерний файл - вхідні дані для режиму відтворення;

- файл, у якому приховується - вхідні дані для режимів приховування та відтворення;

- ліцензія - дані для пропису активації програми;

- системна дата - дані для керування бета (демо) режимом;

- параметри ПК (за наших умов доцільно скористатися серійним номером диску) дані для керування Автентифікацією та прописом активації під конкретний ПК.

Також, із схеми можна побачити, що програма генерує наступні інформаційні потоки:

- файл шифрокоду - вихідні дані для режиму кодування;

- файл розшифрованого коду - вихідні дані для режиму декодування;

- файл контейнер - вихідні дані для режиму приховування;

- відтворений файл - вихідні дані для режиму відтворення.

На схемі також показані внутрішні інформаційні зв’язки:

- зв’язок між підпрограмою генерації псевдовипадкової послідовності і блоками режимів кодування та декодування, де по запиту блоків надходить потік псевдовипадкової послідовності;

- зв’язок між блоками режимів та блоками довідки і сервісу;

- зв’язок між блоками режимів і підпрограмами керування захистом: «Автентифікації», «Бета режим» та «Активація»;

- зв’язок між підпрограмами керування захистом: «Автентифікації» та «Активація».

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи зображена на рисунку 3.2.

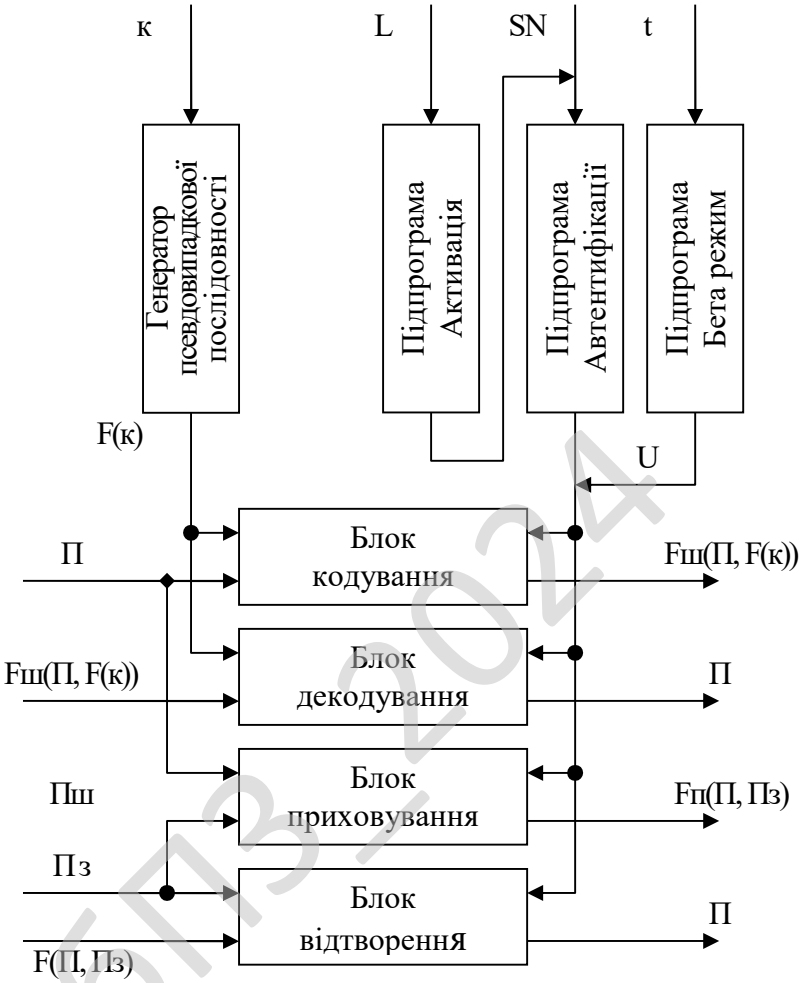

Рисунок 3.2 - Функціональна схема

З рисунка видно, що роботу розробленої системи можна представити функціонуванням сукупності частин: блоку кодування; блоку декодування; блоку приховування; блоку відтворення; генератора псевдовипадкових послідовностей; підпрограма «Активація»; підпрограма «Бета режим»; підпрограма «Автентифікації».

Розглянемо їх роботу. Незалежно від режиму підпрограми «Активація»; «Бета режим»; «Автентифікації» видають в залежності від t - системної дати, SN - серійного номеру диску та L - ліцензії U (вектор з двох складових Au -

Автентифікація та В - Бета режим, які є логічні) дозвіл на відпрацювання режиму. Для режимів кодування та декодування генератор псевдовипадкових послідовностей перетворює к - ключ (визначену множину символів) на деяку значно довшу псевдовипадкову послідовність символів F(к).

У режимі кодування блок кодування за алгоритмом змішує (сумує за певних правил) П - послідовність, що надходить, з псевдовипадковою послідовністю символів F(к) та генерує шифрокод Fш(П, F(к)).

У режимі декодування блок декодування за алгоритмом розшифровує (виділяє шляхом віднімання за певних правил) та генерує П - послідовність (розшифрований код), обробивши псевдовипадкову послідовність символів F(к) та шифрокод Fш(П, F(к)).

У режимі приховування блок приховування за алгоритмом змішує (сумує за певних правил) П - послідовність, що надходить, з деякою наперед відомою послідовністю символів Пз та генерує послідовність Fп(П, Пз).

У режимі відтворення блок відтворення за алгоритмом виділяє (виділяє шляхом віднімання за певних правил) та генерує П - послідовність (приховану послідовність), обробивши наперед відому послідовність символів Пз та послідовність Fп(П, Пз), у якій приховали.

3.4 Розробка діаграми процесів

Оскільки для опису архітектури інформаційної системи 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 [38].

Діаграма процесів розробленої системи зображена на рисунку 3.3. 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 починається з визначення Автентифікації, після чого вибирається режим та у випадку наявності дозволів на виконання проходить його виконання з поверненням до вибору режиму. Дозвіл на виконання режиму надала Автентифікація. Якщо Автентифікація не пройшла, то перевіряється можливість надання Бета (демо) режиму на виконання; якщо і він не проходить, то запускається Активізація і тоді далі на визначення Автентифікації. Активізація програми може вибиратись із режиму напряму. Крім того, з режиму вибору можуть запускатись довідка або налаштування, після яких система знову повертається до режиму вибору. Також наявний перехід від режимів кодування та декодування до генерації псевдовипадкових послідовностей із поверненням до режиму, з якого було зроблено перехід.

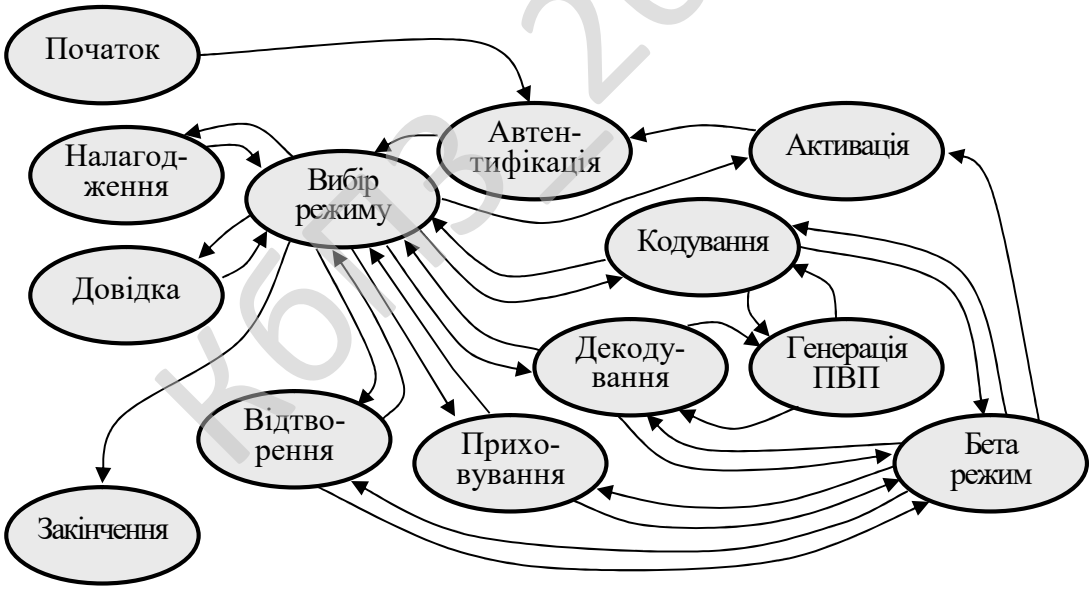

Рисунок 3.3 - Діаграма процесів

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 ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Розглянемо алгоритм роботи основної програми. Його блок-схема зображена на рисунку 4.1.

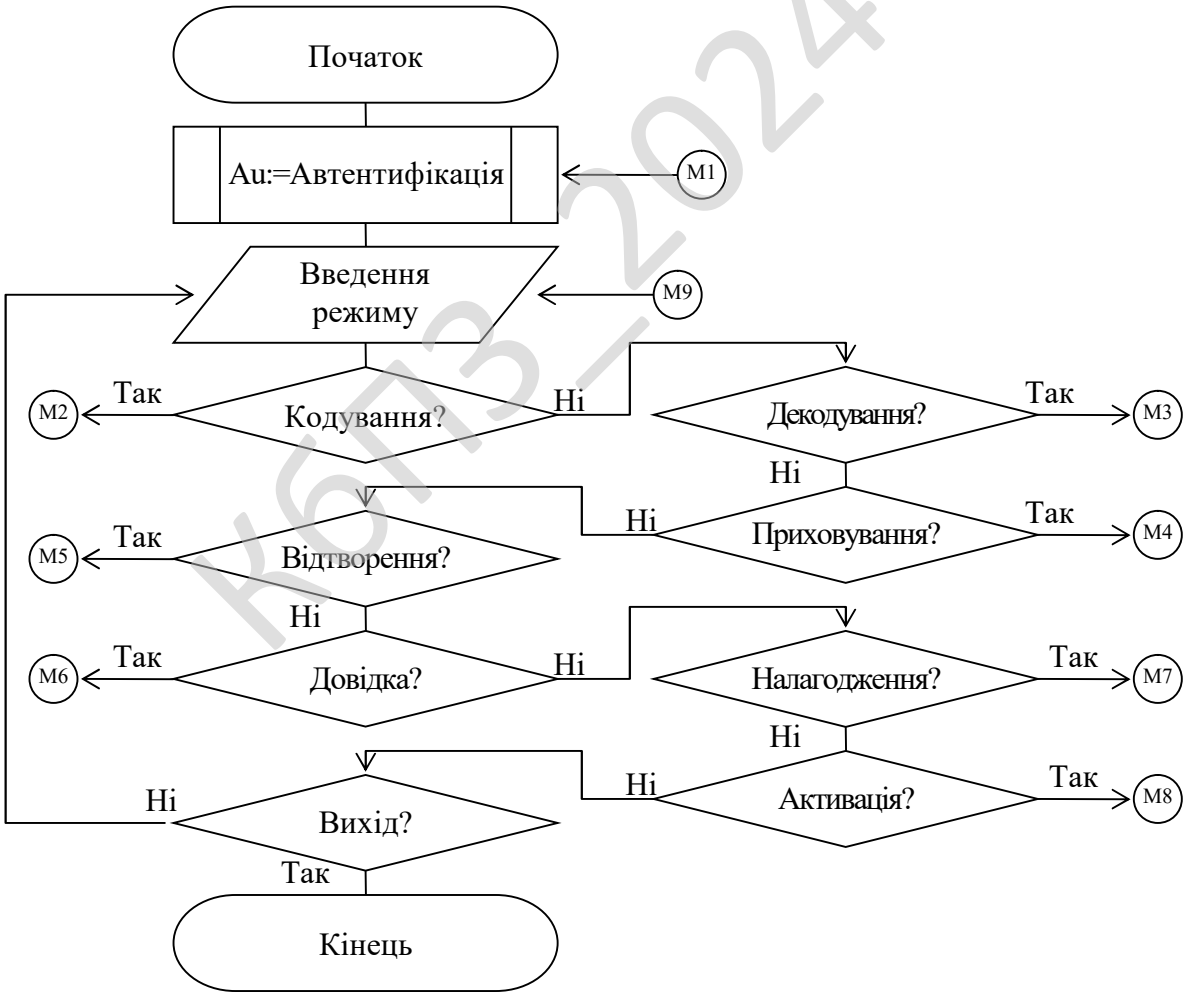

Рисунок 4.1 - Блок-схема роботи основної програми, частина з перевіркою на автентифікацію та вибором режимів роботи

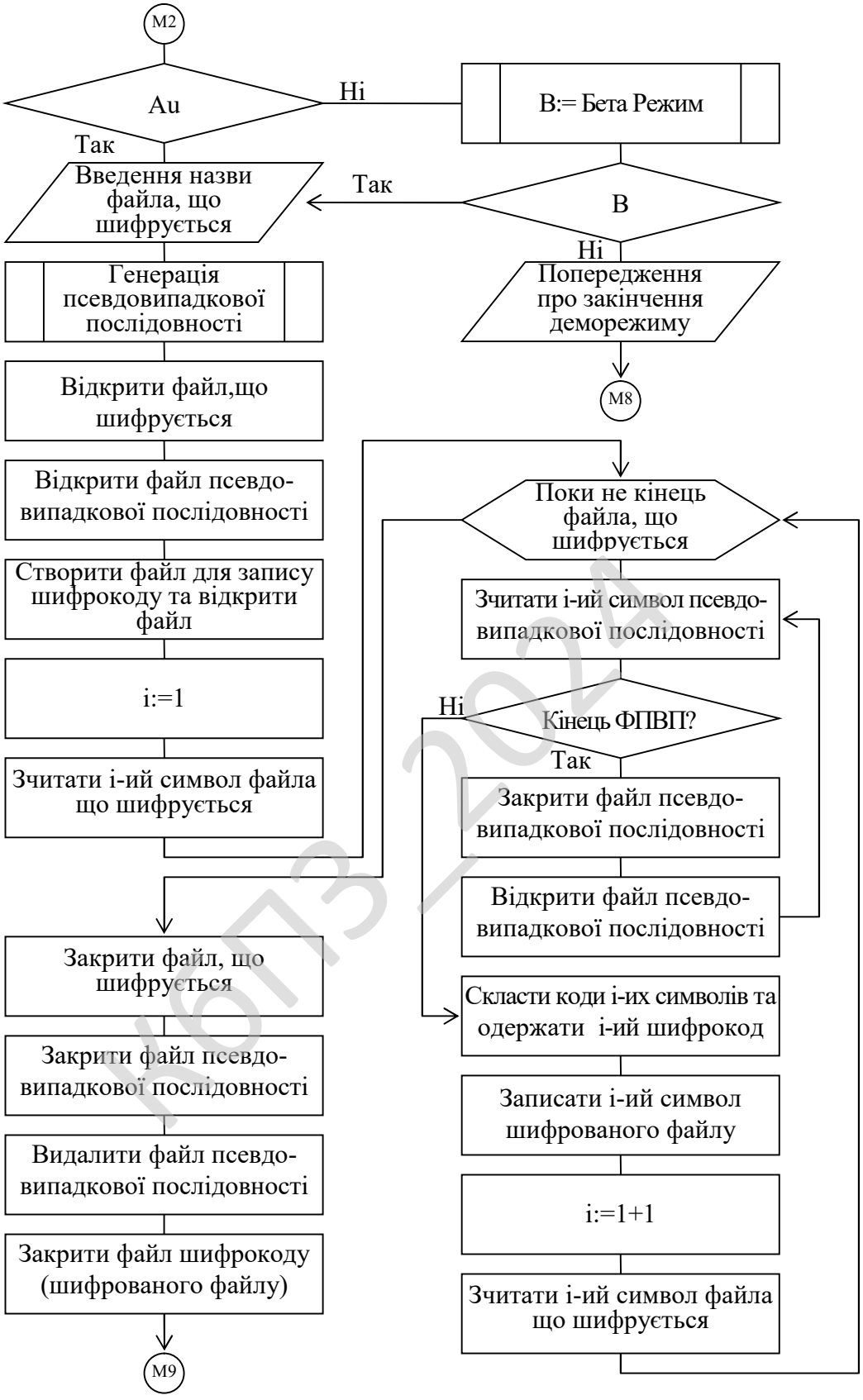

Рисунок 4.2 - Блок-схема роботи основної програми, частина з режимом роботи кодуванням

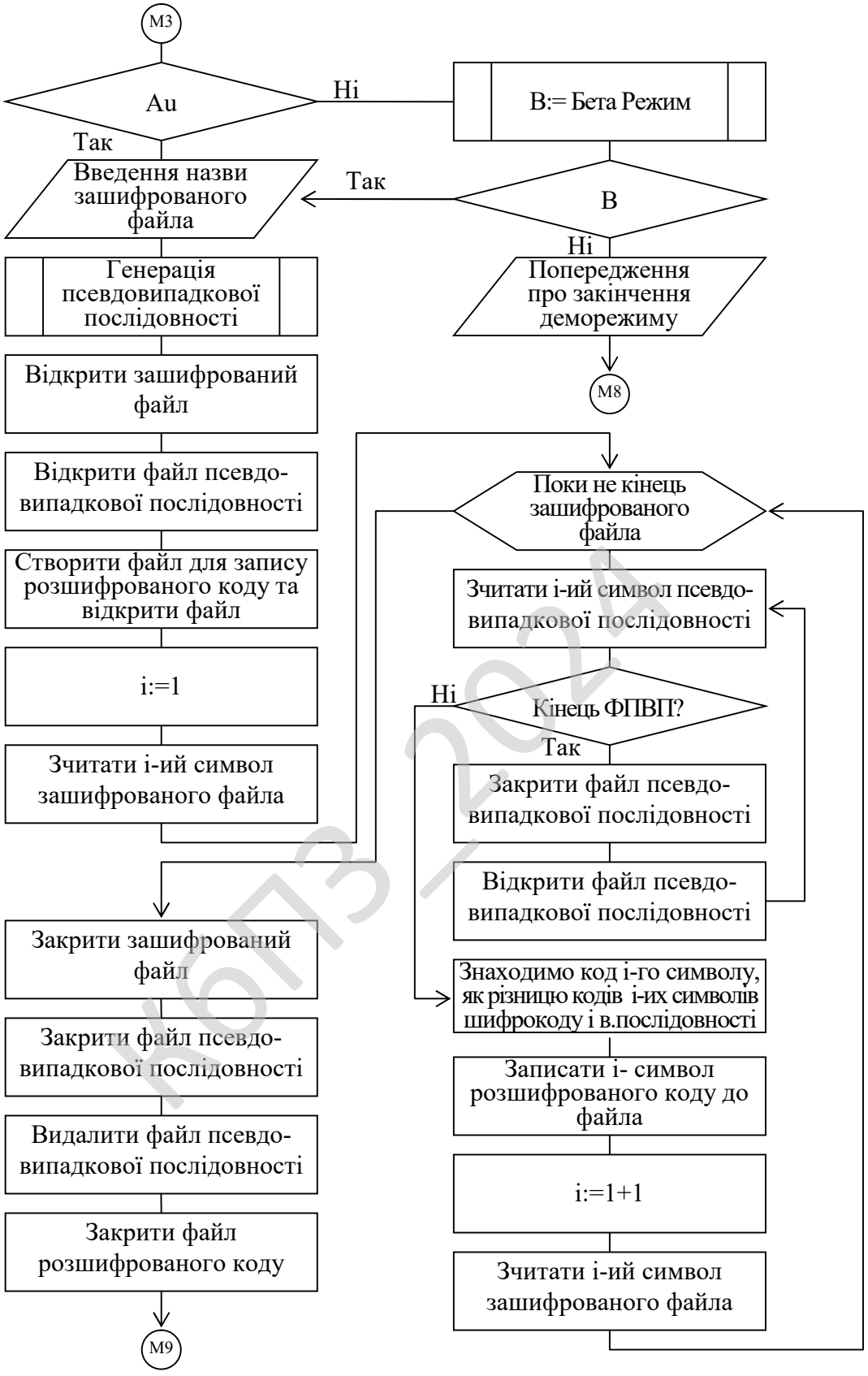

Рисунок 4.3 - Блок-схема роботи основної програми, частина з режимом роботи декодуванням

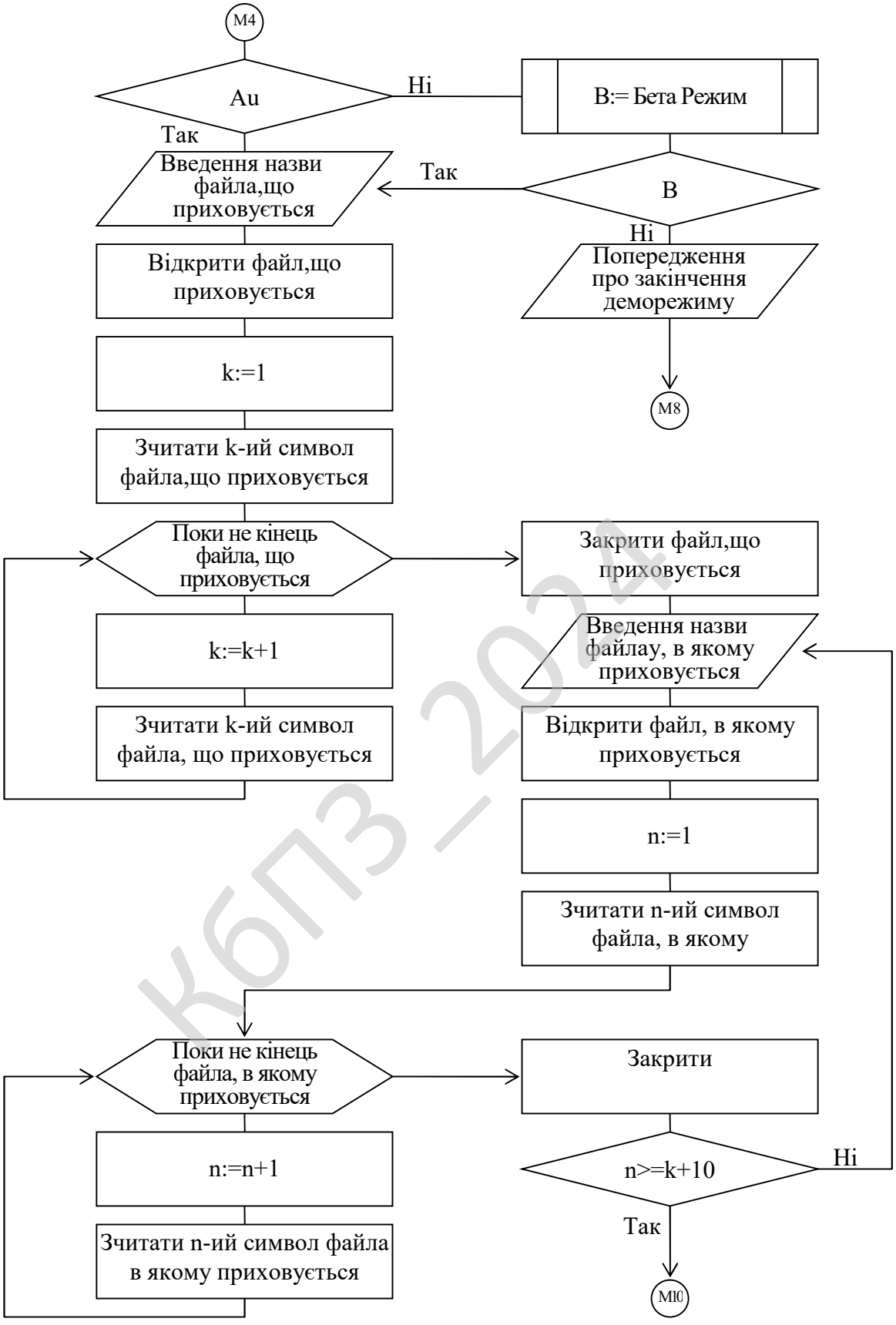

Рисунок 4.4 - Блок-схема роботи основної програми, частина з режимом роботи приховуванням (початкова частина)

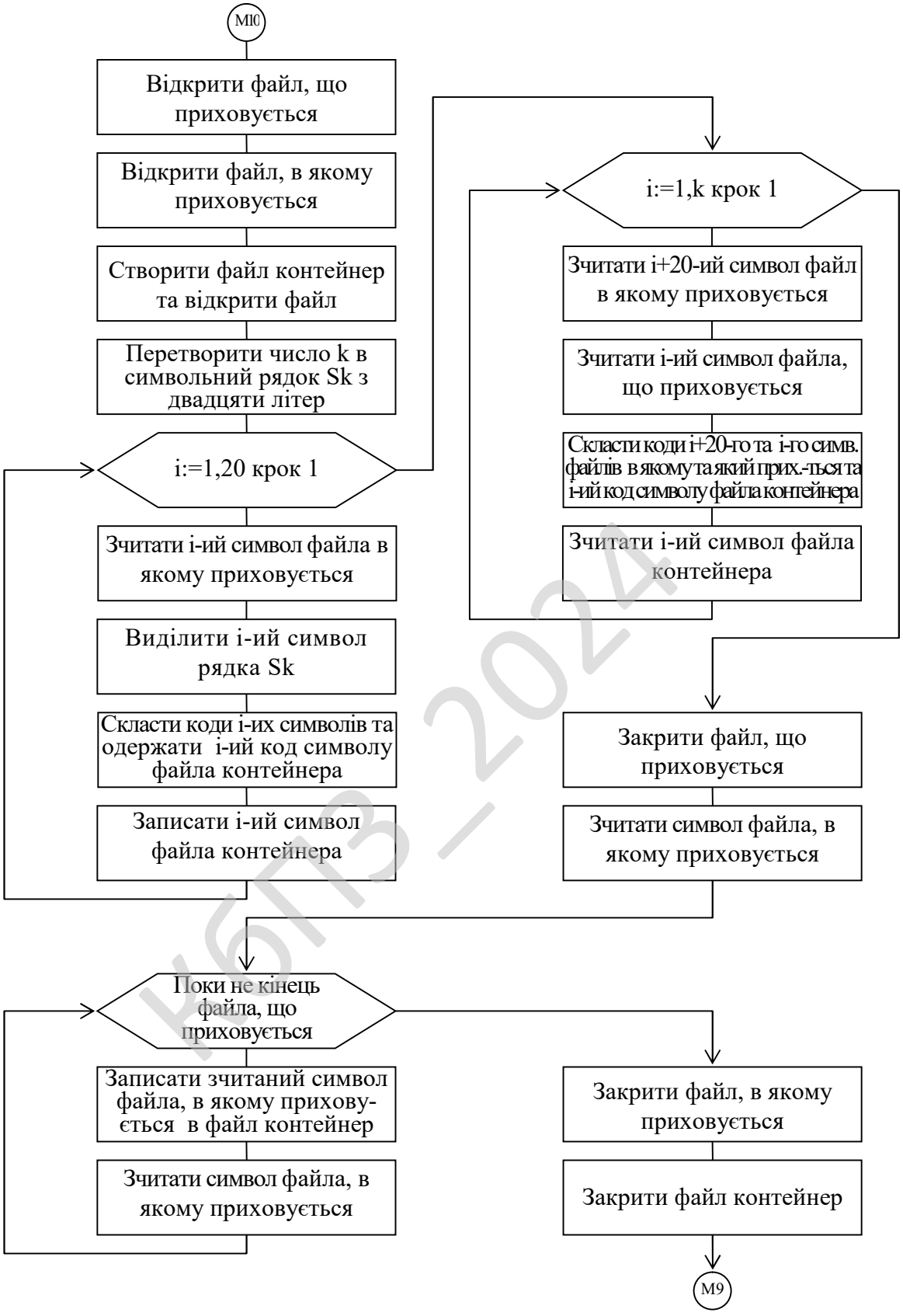

Рисунок 4.5 - Блок-схема роботи основної програми, частина з режимом роботи приховуванням (кінцева частина)

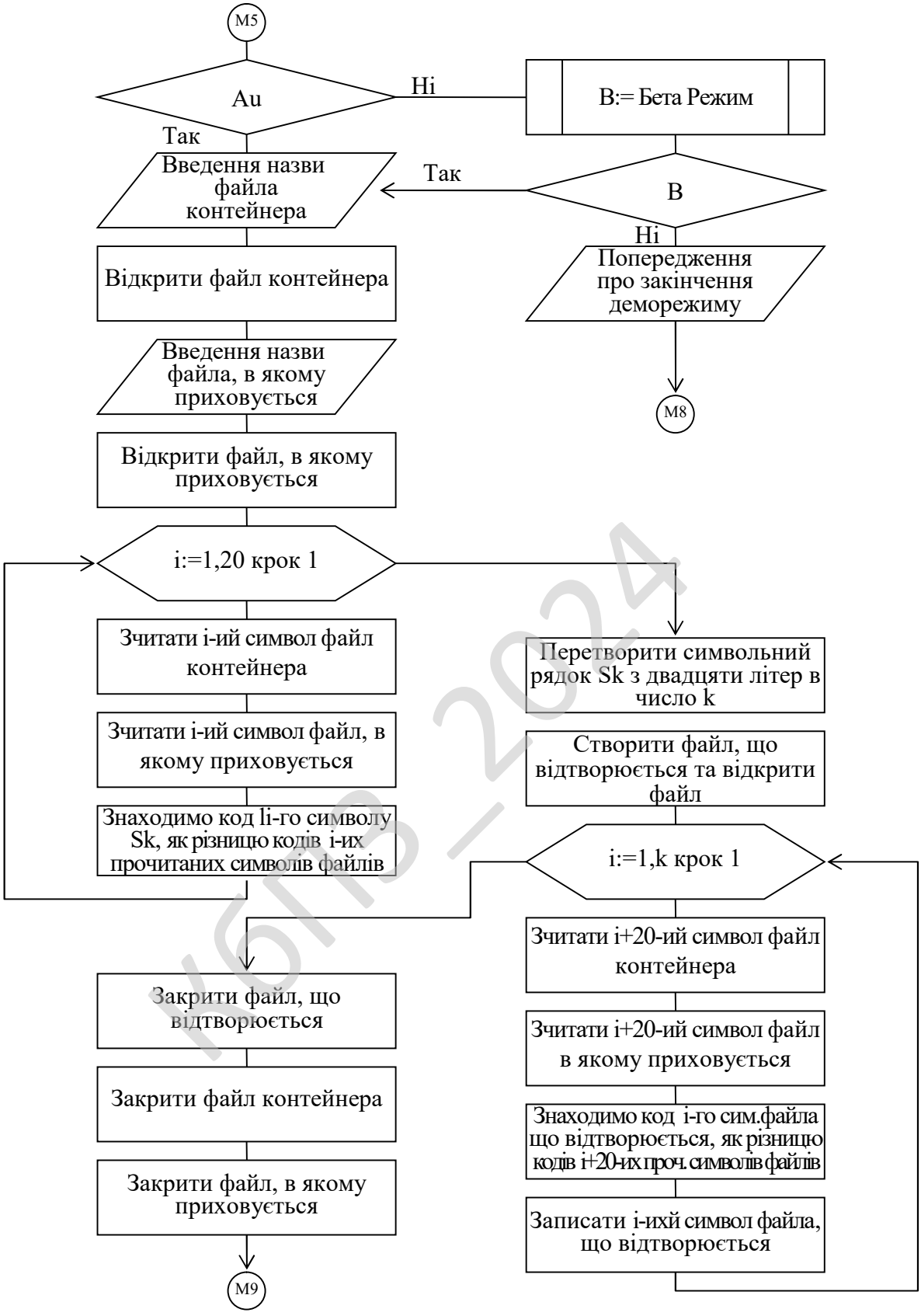

Рисунок 4.6 - Блок-схема роботи основної програми, частина з режимом роботи відтворення (початкова частина)

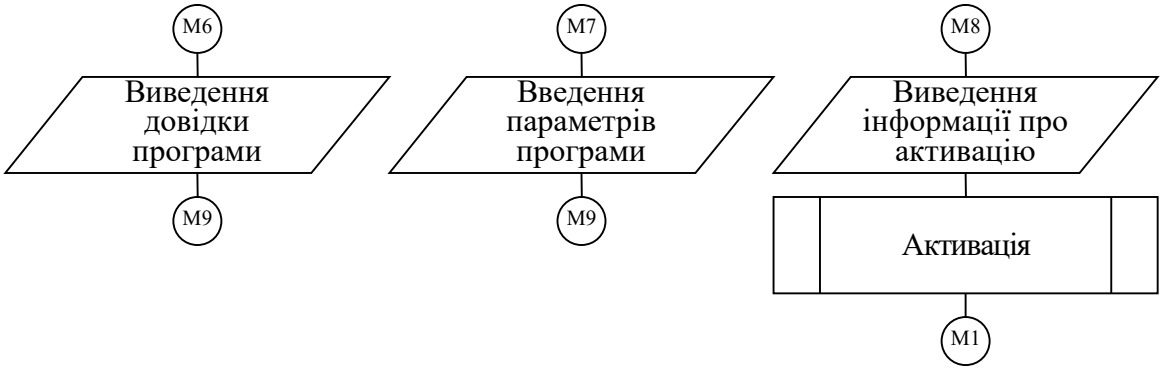

Рисунок 4.7 - Блок-схема роботи основної програми, частина з викликом довідки, налаштування та активації

Рисунок 4.8 - Блок-схема роботи підпрограми активації

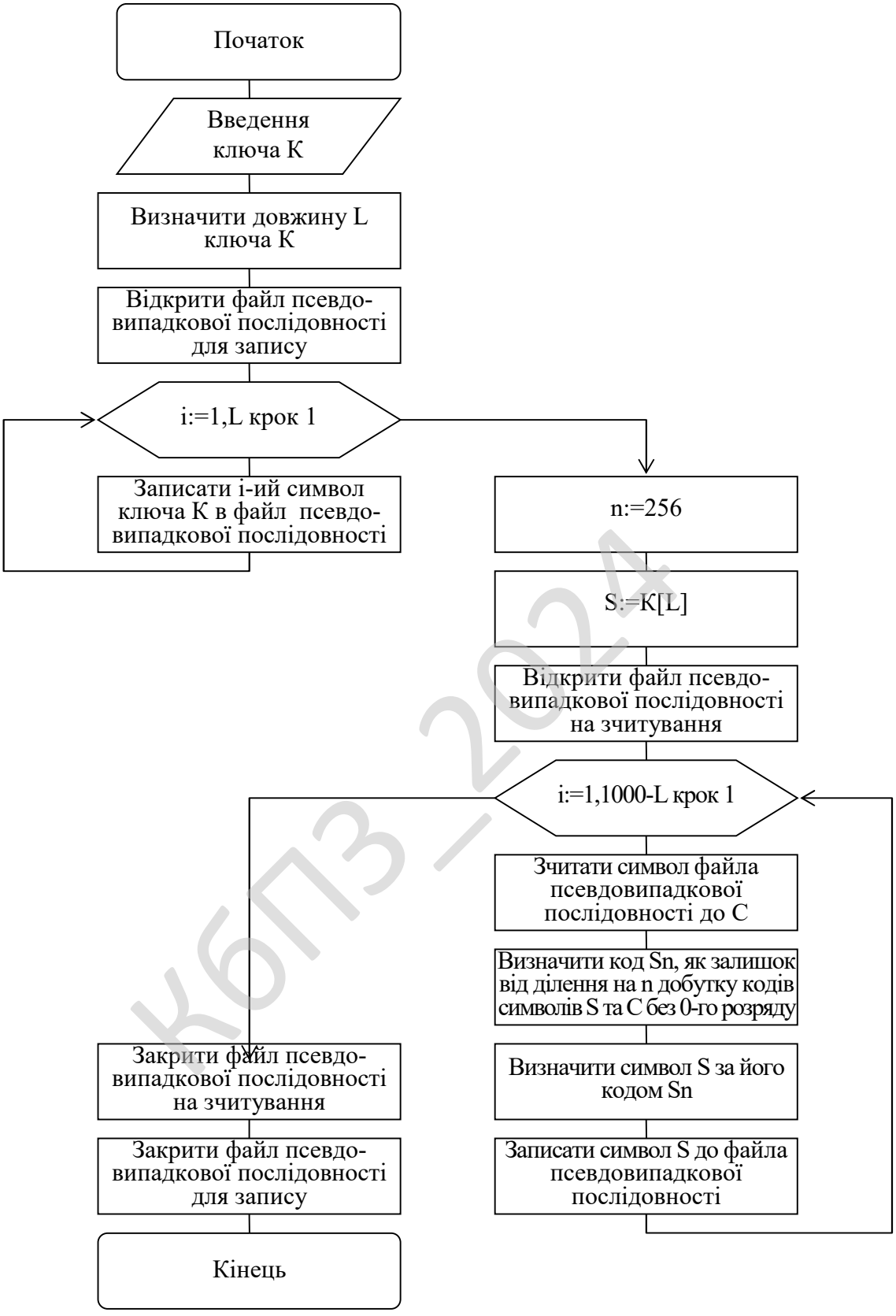

Рисунок 4.9 - Блок-схема роботи підпрограми генерації псевдовипадкової послідовності

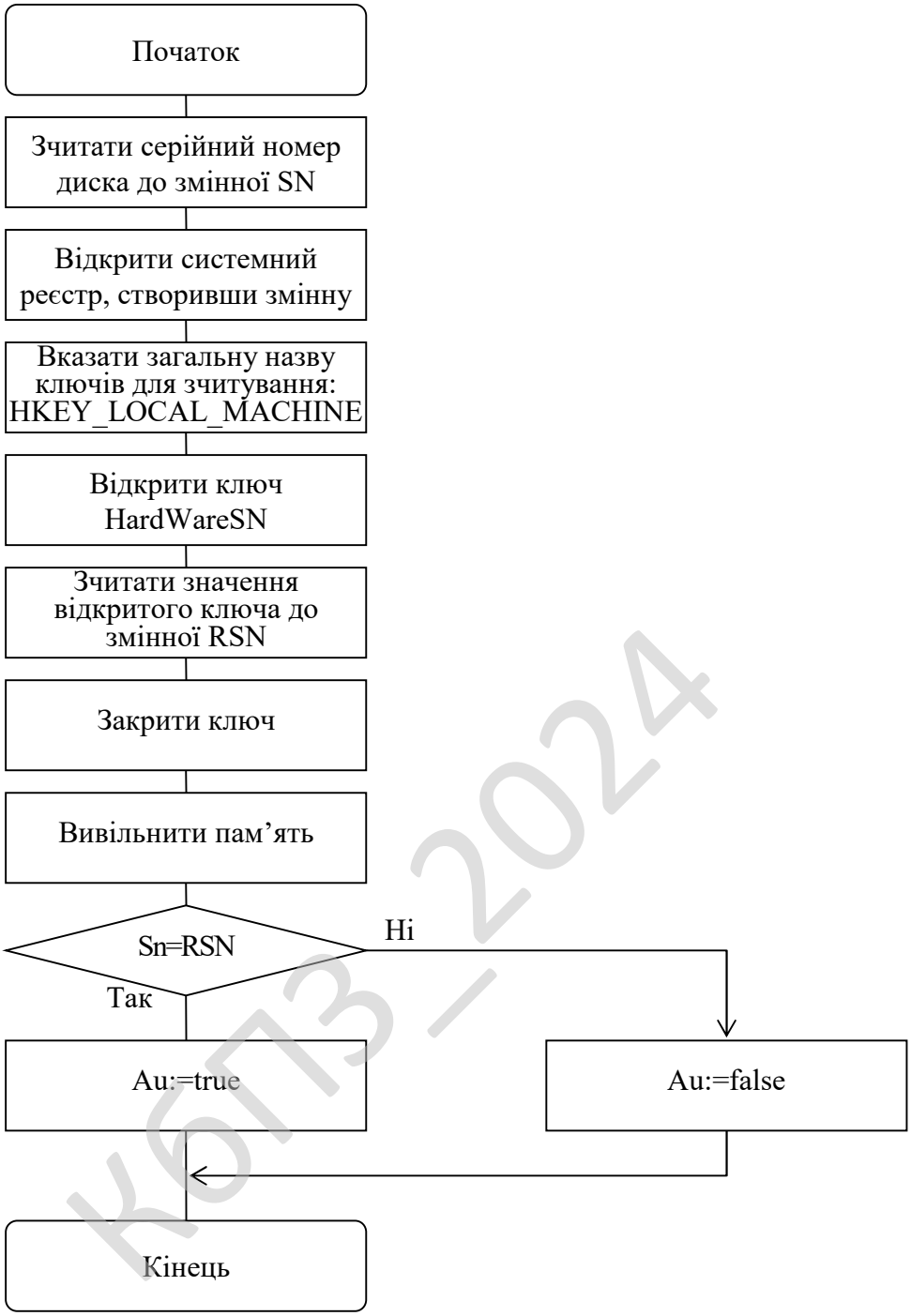

Рисунок 4.10 - Блок-схема роботи підпрограми автентифікації

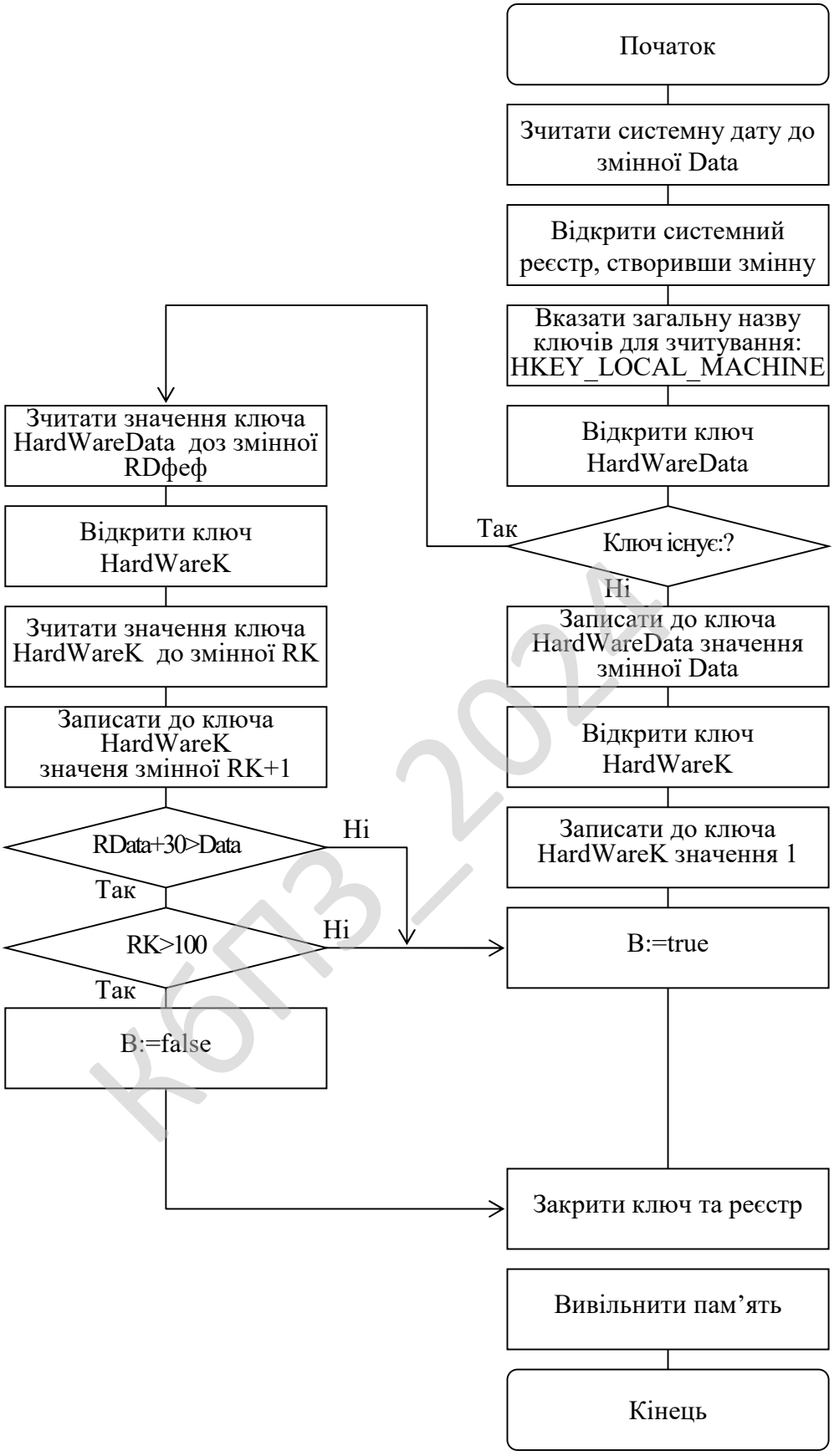

Рисунок 4.11 - Блок-схема роботи підпрограми визначення бета режиму

З представленої блок-схеми роботи програми видно, що спочатку відбувається перевірка на автентифікацію програми (захист на копію програмного забезпечення), яка проводиться окремою підпрограмою.

Потім здійснюється вибір режиму роботи програми, й, в залежності від вибраного режиму, проходить розгалуження виконання роботи програми.

Для режиму кодування у випадку незареєстрованої копії програми включається бета режим (деморежим, у даному випадку повноцінний режим роботи лише з обмеженням в часі та кількістю відпрацювань для стандартного рекламного поширення програмного забезпечення, та проводиться окремою підпрограмою; якщо деморежим не включається, то видається відповідне повідомлення і проходить перехід до вибору режиму). Потім проходить вибір файла для кодування. Після чого генерується псевдовипадкова послідовність та записується до файла. Генерація псевдовипадкова послідовності проводиться окремою підпрограмою для зручної заміни способу генерації самої псевдовипадкової послідовності та можливості генерації псевдовипадкової послідовності в інших режимах роботи програми. Далі проводиться шифрування шляхом сумуванням кодів символів файлів, що шифруються та псевдовипадкової послідовності, з яких проводиться посимвольне зчитування, а результат також посимвольно записується до шифрованого файла. Все це проводиться посимвольно до кінця файла, що шифрується. Якщо файл псевдовипадкової послідовності закінчується раніше, то він перевідкривається. Після чого закриваються всі файли: файл, що щирується, файл псевдовипадкової послідовності, файл зашифрованого коду. Файл псевдовипадкової послідовності в кінці відпрацювання режиму видаляється. По закінченню роботи програма переходить до вибору нового режиму роботи.

Для режиму декодування у випадку незареєстрованої копії програми може включатися бета режим (деморежим, у даному випадку повноцінний

режим роботи лише з обмеженням у часі та кількістю відпрацювань для стандартного рекламного поширення програмного забезпечення) та проводиться окремою підпрограмою. Якщо деморежим не включається, то видається відповідне повідомлення (проходить перехід до вибору режиму.). Потім проходить вибір зашифрованого файла. Після чого генерується псевдовипадкова послідовність та записується до файла. Генерація псевдовипадкової послідовності проводиться окремою підпрограмою для зручної заміни способу генерації самої псевдовипадкової послідовності та можливості генерації псевдовипадкової послідовності в інших режимах роботи програми. Далі проводиться дешифрування шляхом поелементного (для кожного символу окремо) віднімання кодів символів файлів, що розшифрується та псевдовипадкової послідовності, з яких проводиться по-символьне зчитування, а результат також посимвольно записується до дешифрованого файла. Все це проводиться посимвольно до кінця файла, що розшифрується. Якщо файл псевдовипадкової послідовності закінчується раніше, то він перевідкривається. Після чого закриваються всі файли: файл, що розшифровується, файл псевдовипадкової послідовності, дешифрований файл. Файл псевдовипадкової послідовності в кінці відпрацювання режиму видаляється. По закінченню роботи програма переходить до вибору нового режиму роботи.

Для режиму приховування у випадку незареєстрованої копії програми може включатися бета режим (деморежим, у даному випадку повноцінний режим роботи лише з обмеженням у часі та кількістю відпрацювань для стандартного рекламного поширення програмного забезпечення, та проводиться окремою підпрограмо). Якщо деморежим не включається, то видається відповідне повідомленн,я проходить перехід до вибору режиму. Потім проходить вибір файла, що приховується. Відкривається файл, що приховується, та визначається його довжина. Після чого проходить вибір файла, в якому приховується, та визначається його

довжина. Якщо файл, в якому приховується, не приховується і довший за файл, що приховується, на 20 символів, то вибирається інший файл для приховування. І так робиться до тих пір, поки не буде знайдено потрібного файла. Тоді проводиться посимвольне змішування кодів символів. Спочатку в 20 позиціях підмішується довжина файла, що приховується, потім символи самого файла. Змішування проходить простим сумуванням кодів символів (посимвольно). Результат посимвольно записується до файла контейнера. Після закінчення файла, що приховується, до файла контейнера записуються символи файла, в якому приховуються, без змін. Після чого закриваються всі файли, що приховуються, та файл контейнера. По закінченню роботи програма переходить до вибору нового режиму роботи.

Для режиму відтворення у випадку незареєстрованої копії програми може включатися бета режим (деморежим, у даному випадку повноцінний режим роботи лише з обмеженням у часі та кількістю відпрацювань для стандартного рекламного поширення програмного забезпечення, та проводиться окремою підпрограмо; якщо деморежим не включається, то видається відповідне повідомлення, проходить перехід до вибору режиму). Потім проходить вибір файла контейнера і відкривається. Далі проходить вибір файла, до якого приховували, та відкривається. Посимвольно відтворюються шляхом віднімання коду символу файла до якого приховували, від коду символу файла контейнера послідовність в двадцять позицій, що задає довжину файла, що приховували. Потім, також посимвольно, визначаються наступні відомі кількістю символів файла, що приховували, шляхом віднімання коду символу файла, до якого приховували, від коду символу файла контейнера, які посимвольно записуються до файла, що відтворюється. Після чого закриваються всі файли: контейнерний, до якого приховувати, та той, який відтворено. По закінченню роботи програма переходить до вибору нового режиму роботи.

Вихід з програми проходить по вибору вихід (завершення).

З представленої блок-схеми роботи підпрограми активації видно, що спочатку відбувається введення інформації про активацію (серійного номера диска), потім відкривається системний реєстр, встановлюються ключі (загальна назва), відкривається ключ, записується його значення, закривається ключ, вивільнюється пам'ять.

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 остаток 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.

З представленої блок-схеми роботи підпрограми автентифікації видно, що спочатку відбувається зчитування серійного номера диска, потім відкривається системний реєстр, встановлюються ключі (загальна назва), відкривається ключ, зчитується його значення, закривається ключ, вивільнюється пам'ять, після чого перевіряється на співпадання значень зчитаного з пристрою та системного реєстру. У випадку співпадання функція приймає значення істина, неспівпаданні - хиба.

З представленої блок-схеми роботи підпрограми визначення бета режиму, що спочатку відбувається зчитування системної дати, потім відкривається системний реєстр, встановлюються ключі (загальна назва),

відкривається ключ HardWareData/ Якщо він не існує, то створюється й до нього заноситься значення поточної системної дати. Також відкривається (створюється) ключ HardWareK, до якого заноситься 1, і функція приймає значення істина. У видку існування ключа HardWareData він зчитується і також відкривається ключ HardWareK, який зчитується і його значення збільшується на 1. Після чого перевіряються значення ключів, значення ключа HardWareData не перебільшує системну дату на 30 діб або значення ключа HardWareK не перебільшує 100 (відпрацювань). У такому випадку функція приймає значення істина, в іншому випадку функція приймає значення хиба. На завершення підпрограми закриваються ключі та реєстр, вивільняється пам'ять.

4.2 Захист розробленого програмного забезпечення

Для захисту системи (програми) від санкціонованого копіювання пропонується один із загальнорозповсюджених методів - прив'язка до параметрів комп'ютера [39].

У нашому випадку доцільно організувати прив'язку до жорсткого диска. Технічно прив'язка виконується до серійного номера вінчестера. В Delphi безпосередньо серійний номер можна визначати [40].

```
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;
```

Автентифікацію системи пропонується визначати шляхом порівняння зчитаного номера диска з номером, який було попередньо зареєстровано при

встановленні програми, записано до системного реєстру. З метою отримання таких даних може бути використано зчитування з реєстру за допомогою функції [41]:

```
function RegSerialNamder: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 автентифікації проведення процедури прив'язки (активації) системи до обладнання може проводитися записом номера диска до системного реєстру при встановленні програми за допомогою:

```
procedure Sn_HDD;
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;
```

Багато робіт присвячено захисту даних, захисту програмного забезпечення та захисту інформацій в цілому [42. 43, 44, 45, 46, 47, 48, 49, 50. 51], проте просто захист від несанкціонованого копіювання робити недоцільно.

Бажано давати демонстрацію роботи програми. Наприклад, можна давати час та/або кількість відпрацювань, для чого можна використати реєстр.

```
function Beta:boolean;
uses Registry;
var regD,regK:TRegistry;
Rd:TDateTime;
Rk:DWORD;
begin
regD := TRegistry.Create;
regD.RootKey := HKEY_LOCAL_MACHINE;
if reg.OpenKey('HardWareData', false)
then
begin
Rd:=RegKeyGetDateTime(HKEY_LOCAL_MACHINE,'HardWareData');
regK:=TRegistry.Create;
regK.OpenKey('HardWareK', true);
Rk:= RegKeyGetDw(HKEY_LOCAL_MACHINE,'HardWareK');
if  Rd+30>Date and Rk>100
then
Result := false
else
begin
RegKeySettDw(HKEY_LOCAL_MACHINE,'HardWareK',Rk+1);
Result := true;
end;
end
else
begin
regD.OpenKey('HardWareData}',true)
RegKeySetDateTime(HKEY_LOCAL_MACHINE,'HardWareD',Date);
regK := TRegistry.Create;
regK.OpenKey('HardWareK}', true)
RegKeySettDw(HKEY_LOCAL_MACHINE,'HardWareK',1);
result := true;
end;
regD.CloseKey;
regD.Free;
regK.CloseKey;
regK.Free;
end;
```

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Загалом використання програмного забезпечення вимагає розгляду наступних завдань: підготовка користувачів програмного забезпечення, встановлення програмного забезпечення, його запуск та налагодження, контроль працездатності та супровід програмного забезпечення.

Підготовка користувачів програмного забезпечення для системи захисту даних на основі генерації псевдовипадкових послідовностей загалом не вимагає ніяких специфічних навиків. Програма оснащена довідковою системою її користувача, що має загальні навики роботи з комп'ютерними програмами, без будь-яких труднощів він може одержувати необхідну для робити інформацію, щодо функціональних можливостей та порядку їх використання. То ж перед введенням в експлуатацію системи захисту даних на основі генерації псевдовипадкових послідовностей немає необхідності у проведенні спеціальних навчань персоналу, а тому вони і не проводяться .

Порядок встановлення системи захисту даних на основі генерації псевдовипадкових послідовностей не відрізняється від загально прийнятих для програмного забезпечення - для запуску програми необхідно й достатньо організувати доступ до файлів програми, які можуть бути записані на жорсткий диск комп'ютера (перенесені, як звичайні файли, які були скопійовані з іншого комп'ютера із встановленою програмою; інстальовані з дистрибутива - спеціально скомпонованого файлу для завантаження; розпаковано з архіву, скопійовано з мережі, тощо) або принесені (монтовані) на зовнішньому з'ємному носію. Обов'язковою умовою для працездатності системи захисту даних на основі генерації псевдовипадкових послідовностей є наявний доступ до системного реєстру, в якому зберігаються або заносяться дані.

Запуск та налагодження системи захисту даних на основі генерації псевдовипадкових послідовностей є стандартним для програм, написаних для сімейства ОС Windows. Враховуючи те, що файлом програми є Shyfr.exe, запуск програми може проводитися так: з меню «програми» операційної ОС (якщо прописано); з меню «виконати» операційної ОС; з робочого столу ярликом програми (якщо встановлено сам ярлик на робочому столі для програми); з панелі задач (якщо встановлено ярлик на панелі швидкого запуску); шляхом натиснення комбінацій гарячих кнопок для запуску програми (якщо така комбінація прописана для даної програми);автоматично при завантажені ОС (якщо вказано автоматичне завантаження при запуску ОС); автоматично при підключенні зовнішнього з'ємного носія (якщо автоматичний запуск програми прописано); з файлових менеджерів при запуску файла програми тощо. Після запуску програми може проводитися налагодження - вибираються мова інтерфейсу, кольори тощо. Одним із пунктів налагодження програми є активація програмного засобу - пропис автентичності, для цього за допомогою відповідного режиму необхідно задати користувача та одержати договір на використання програмного забезпечення (у випадку комерційного використання програми передбачається оплата, тобто договір підкріплюється оплаченим рахунком; у такому випадку за номером договору і логіна користувача та наявному підтвердженні оплати надається код активації програми, за допомогою якого прописується можливість проходження автентифікації; до вказаної процедури системи захисту даних на основі генерації псевдовипадкових послідовностей в деморежимі визначений час, який фіксує кількість відпрацювань).

Успішність проекту на царині ІТ технологій передбачає обов'язковий супровід програмного забезпечення з контролем працездатності, який призначений забезпечувати постійний прибуток, шляхом підтримання зростаючих вимог до функціональних можливостей програмних засобів.

Зважаючи на зазначене, для успішності розробки передбачена можливість програми для модернізації - додавання нових методів генерації псевдовипадкових послідовностей та розробки нових принципів приховування (методи приховування, які більш ефективні з точки зору непомітності, застосовуватимуть для різних типів контейнерних файлів). Внаслідок виправлень, виявлених у процесі експлуатації помилок, процес підтримки працездатності системи захисту даних на основі генерації псевдовипадкових послідовностей приводить до внесення зміни в програму, що в підсумку виллється в перехід до її нових версій.

6 НАУКОВА НОВИЗНА

У магістерській роботі розроблено програмне забезпечення, яке призначено для системи захисту даних на основі генерації псевдовипадкових послідовностей.

Метою розробки є дослідження та програмна реалізація системи захисту даних .

Об'єктом дослідження є процес захисту.

Предметом дослідження є методи захисту.

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

Наукова новизна отриманих результатів. У процесі вирішення завдань, обумовлених цілями дослідження, отримані наступні результати:

- Удосконалено метод захисту даних на основі генерації псевдовипадкових послідовностей.

- Розроблено вітчизняний продукт захисту даних на основі генерації псевдовипадкових послідовностей, який має більш широкі можливості, на відміну від існуючих аналогів.

7 МАРКЕТИНГОВЕ ТА ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ ІТ-ПРОЄКТУ

7.1 Визначення цільової аудиторії кінцевого готового продукту

Результати дослідження та програмної реалізації системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів можуть бути цікавими для таких груп, представлених на рисунку 7.1.

Спеціалісти з кібербезпеки – професіонали, які займаються захистом інформації, можуть використовувати результати для вдосконалення методів захисту персональних даних від несанкціонованого доступу.
Розробники програмного забезпечення – для інтеграції ефективних алгоритмів шифрування та захисту даних у свої продукти, зокрема для створення додатків або платформ, що потребують високого рівня безпеки.
ІТ-компанії – зокрема ті, які працюють у сфері обробки великих обсягів персональних даних (наприклад, фінансові установи, медичні заклади, компанії електронної комерції), можуть бути зацікавлені у використанні та впровадженні нових технологій для забезпечення конфіденційності даних.
Науковці та дослідники – у галузі інформаційної безпеки, криптографії та алгоритмів захисту даних, які можуть розглядати дослідження як базу для подальших наукових робіт або розробки нових методів шифрування.
Державні установи – особливо регуляторні органи, відповідальні за захист персональних даних та розробку стандартів інформаційної безпеки, для використання найкращих практик і вдосконалення нормативних актів.
Компанії, що займаються консалтингом у сфері безпеки – для надання консультаційних послуг щодо впровадження систем захисту персональних даних у різних організаціях.
Ентузіасти у сфері криптографії – які цікавляться новими підходами до захисту даних і готові вивчати й випробовувати алгоритми на практиці.

Рисунок 7.1 – Цільова аудиторія

Таким чином, широке коло фахівців, що працюють у сфері захисту даних та інформаційної безпеки, можуть виявити інтерес до цього дослідження.

7.2 Оцінка привабливості шляхом застосування методів експертних оцінок

Для оцінки привабливості програмної реалізації системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів можна використати методи експертних оцінок, наприклад, метод парних порівнянь або метод рейтингових оцінок. Зупинимось на використанні методу рейтингових оцінок.

Для оцінки привабливості системи будемо використовувати критерії, наведені на рисунку 7.2.

Рисунок 7.2 – Критерії експертної оцінки

Залучаємо групу експертів (наприклад, 5-10 осіб), що мають досвід у сфері кібербезпеки, криптографії, розробки ПЗ та оцінювання інформаційних систем. Експерти визначають вагу кожного критерію за шкалою від 0 до 1, де 1 означає найбільшу важливість. Сума всіх ваг повинна дорівнювати 1. Встановлюємо наступні ваги для критеріїв: надійність захисту даних -0.25, швидкість роботи -0.20, зручність використання -0.15, масштабованість -0.10, вартість впровадження -0.10, відповідність стандартам -0.10, гнучкість -0.10.

Експерти оцінюють кожен критерій для розробленої системи за шкалою від 1 до 10, де 10 – найкращий показник, а 1 – найгірший. Після оцінок експертів результати зводимо до таблиці 7.1.

Таблиця 7.1. – Зведені дані в процесі експертної оцінки

Критерій	**Вага**	**Оцінка (середнє значення)**
Надійність захисту даних	0.25	8
Швидкість роботи	0.20	7
Зручність використання	0.15	6
Масштабованість	0.10	7
Вартість впровадження	0.10	6
Відповідність стандартам	0.10	9
Гнучкість	0.10	8

Для обчислення загальної оцінки привабливості системи необхідно перемножити вагу кожного критерію на його оцінку, а потім підсумувати результати. Загальна оцінка привабливості становить 7.3 з 10, що свідчить про те, що система має високий рівень привабливості для цільової аудиторії за обраними критеріями. На основі цього можна зробити висновки про

подальші дії, наприклад, визначити, які аспекти потребують покращення. Цей метод дозволяє отримати числову оцінку привабливості системи, що спрощує прийняття рішень щодо впровадження або вдосконалення системи захисту персональних даних.

7.3 Вибір методу оцінки вартості ПЗ

Для оцінки вартості програмної реалізації системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів можна розглянути декілька методів. Найкращий метод залежить від специфіки проекту, обсягу робіт і точності, яка вам потрібна. Для оцінки вартості програмної реалізації системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів рекомендуємо використовувати метод оцінки на основі трудових витрат (Bottom-Up), оскільки він дає точні результати і дозволяє детально розрахувати вартість кожного етапу проекту. Він особливо корисний, якщо проект є складним і включає багато етапів розробки, тестування та впровадження. Якщо у вас є база даних попередніх проектів або приклади схожих систем, можна додатково використовувати метод аналогії для попередньої оцінки, а потім уточнити її за допомогою методу «Bottom-Up».

Загалом, цей метод передбачає оцінку вартості проєкту на основі розрахунку обсягу робіт, які необхідно виконати для реалізації системи. Використовується для детального проєктування і розрахунку витрат на основі часу, необхідного на виконання кожного завдання.

Процедура проведення такої оцінки передбачає: розбити проект на окремі завдання (етапи розробки, тестування, впровадження); оцінити трудові витрати на кожне завдання в годинах або днях; визначити ставки оплати праці для кожного фахівця (розробник, аналітик з безпеки,

тестувальник тощо); підсумувати витрати для кожного завдання, враховуючи оплату праці та інші можливі витрати (обладнання, ліцензії, інфраструктура).

Метод є трудомістким, але при цьому з високою точністю та кращим контролем бюджету.

7.4 Розрахунок економічної ефективності від впровадження реалізованого ПЗ як фактору його привабливості

Економічна ефективність впровадження системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів може бути оцінена через кілька ключових аспектів, які демонструють, як система може знизити витрати, підвищити доходи або зменшити ризики. Нижче наведено ключові фактори на рисунку 7.3.

Підсумковий розрахунок економічної ефективності

Сума економічних вигод від впровадження системи захисту персональних даних може виглядати так:

Зменшення ризиків витоку даних: 450,000 грн

Покращення репутації: 400,000 грн

Зменшення витрат на юридичні послуги: 180,000 грн

Зменшення витрат на штрафи: 270,000 грн

Покращення продуктивності: 50,000 грн

Загальна економічна ефективність = 450,000 + 400,000 + 180,000 + 270,000 + 50,000 = 1,330,000 грн на рік. Впровадження системи захисту персональних даних може мати суттєвий позитивний економічний ефект, що перевищує витрати на її реалізацію.

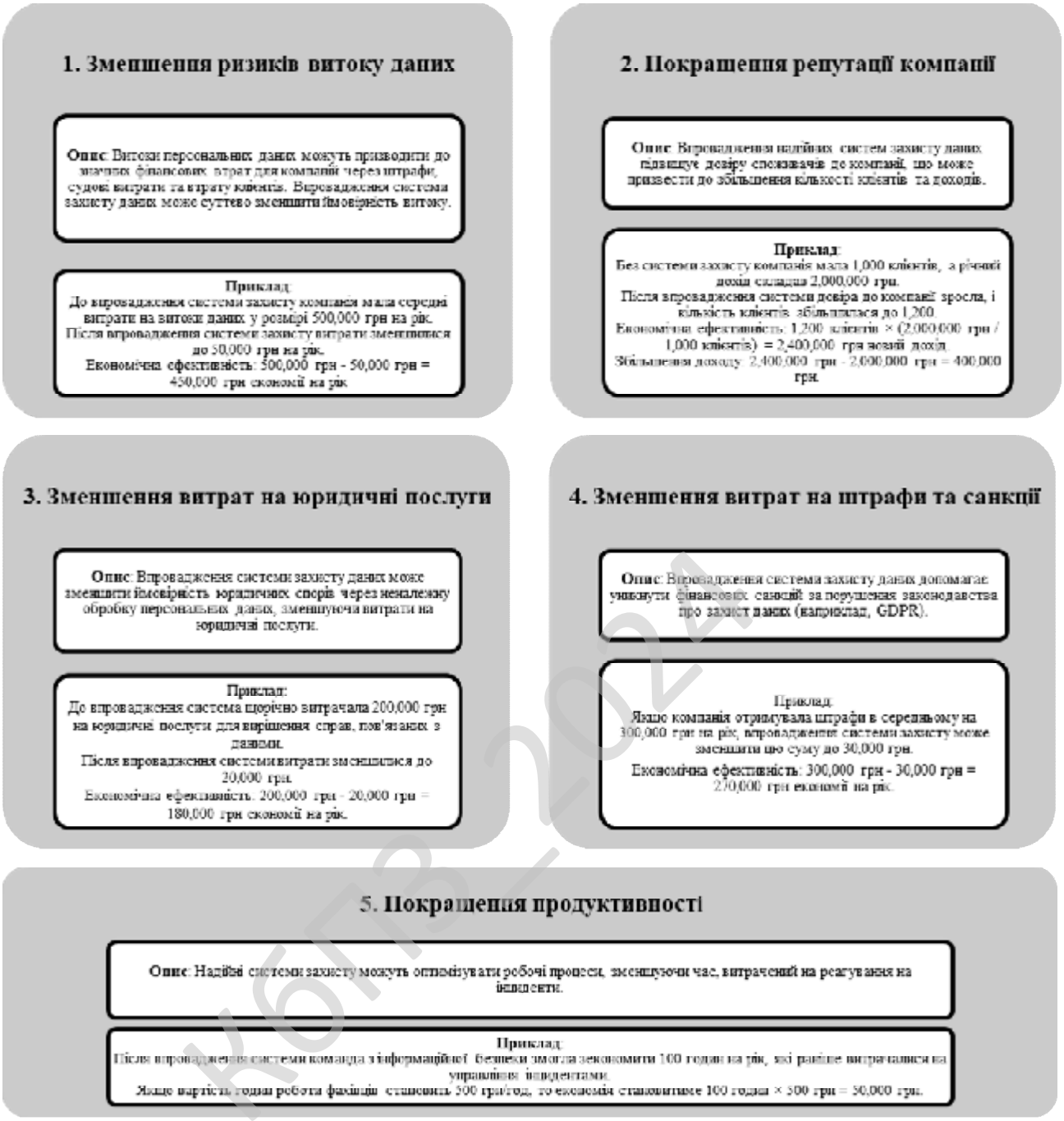

Рисунок 7.3 – Економічна ефективність від реалізації проєкту для клієнта

7.5 Пропозиція алгоритму просування проєкту розробки ПЗ

Алгоритм просування проекту програмної реалізації системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів

представлено на рисунку 7.3. Цей алгоритм може включати кілька етапів, від дослідження ринку до запуску продукту.

Рисунок 7.3 – Алгоритм просування проєкту

Цей алгоритм просування проекту програмної реалізації системи захисту персональних даних допоможе вам стратегічно підходити до маркетингу та продажу вашого продукту. Кожен етап важливий для успіху вашої ініціативи, тому важливо ретельно планувати та реалізовувати всі частини цього процесу.

7.6 Оптимізація каналів збуту та шляхів реалізації ПЗ

Оптимізація каналів збуту та шляхів реалізації проекту програмної реалізації системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів може включати в себе кілька стратегій та дій:

– створити професійний веб-сайт, де будуть представлені можливості системи, переваги та відгуки користувачів. Включити функціонал для онлайн-придбання або замовлення демонстрації;

– активно використовувати платформи, такі як Facebook, LinkedIn, Twitter та Instagram для просування продукту, публікації корисного контенту та залучення нових клієнтів;

– створити список підписників для розсилки новин, акцій та навчальних матеріалів;

– встановити партнерства з компаніями, які надають супутні послуги (наприклад, постачальники IT-інфраструктури, консультанти з безпеки) для спільного просування продукту;

– запустити програми для залучення афілійованих маркетологів, які отримуватимуть комісію за продажі, що здійснюються через їхні канали;

– проводити вебінари та тренінги з теми захисту персональних даних, де можна продемонструвати переваги вашої системи. Це також допоможе підвищити обізнаність про важливість захисту даних;

– залучати відомих експертів для участі в заходах, що збільшить їхню привабливість та надійність продукту;

– запровадити знижки або спеціальні тарифи для малих підприємств і стартапів, які потребують захисту даних, але можуть мати обмежений бюджет;

– пропонувати безкоштовні пробні версії продукту, щоб потенційні клієнти могли оцінити його переваги;

– регулярно проводити оцінку ефективності різних каналів збуту (онлайн, офлайн, партнерських) та адаптувати стратегію відповідно до отриманих результатів;

– збирати відгуки від клієнтів про їхні враження від покупки та використання продукту, щоб виявити області для покращення;

– використовувати платну рекламу для залучення цільової аудиторії, яка може бути зацікавлена в системах захисту даних;

– впровадити ремаркетинг для повернення відвідувачів, які не завершили покупку;

– брати участь у тематичних виставках та конференціях з безпеки даних, де можна продемонструвати систему потенційним клієнтам;

– налагоджувати зв’язки з іншими учасниками галузі для обміну ідеями та можливостями співпраці;

– впровадити CRM-систему для управління контактами, ведення обліку клієнтів, автоматизації процесів продажу та аналізу даних;

– використовувати дані CRM для створення персоналізованих пропозицій на основі історії покупок та вподобань клієнтів.

Оптимізація каналів збуту та шляхів реалізації проекту системи захисту персональних даних вимагатиме комплексного підходу з урахуванням різних стратегій і тактик. Важливо залишатися гнучкими та готовими адаптуватися до змін на ринку та вимог клієнтів, щоб забезпечити успіх вашого проекту.

7.7 Визначення ключових факторів успіху конкретного проєкту

Ключові фактори успіху проекту програмної реалізації системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів включають в себе різноманітні аспекти, які впливають на його ефективність, надійність і прийнятність серед користувачів. Схематично вони подані на рисунку 7.4.

Рисунок 7.4 – Ключові фактори успіху проєкту

Успіх проекту програмної реалізації системи захисту персональних даних залежить від множини факторів, які повинні бути враховані на всіх етапах його реалізації. Комбінація технічних, правових, організаційних та маркетингових аспектів допоможе забезпечити ефективність, надійність і конкурентоспроможність вашої системи.

8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

У наш час продовжують розвиватись різні методи розробки складного програмного забезпечення. Праця програмістів характеризується значною розумовою напругою і нервово-емоційним навантаженням, високою напруженістю зорової роботи і достатньо великим навантаженням на м'язи рук при роботі з клавіатурою.

На робочому місці програміста повинні бути передбачені заходи захисту від можливих шкідливих і небезпечних факторів. Для визначення переліку шкідливих та небезпечних факторів при роботі із комп'ютерною технікою проведемо аналіз умов праці програміста, який зайнятий розробкою дослідження та програмною реалізацією програмного продукту. Якість роботи залежить від багатьох факторів , наприклад, від таких, як:

- освітлення приміщення ;
- вентиляція приміщення ;
- оптимальні параметри повітряного середовища в приміщенні тощо.

Для цього проведемо аналіз існуючих санітарно-гігієнічних умов праці на робочому місці програміста, щоб визначити саме ті фактори, які знижують працездатність людини у її професіональній сфері і можуть викликати професійне захворювання та запропонувати заходи щодо зменшення впливу комп'ютера на організм його користувача.

8.1 Шкідливі та небезпечні чинники на робочому місці програміста

При роботі з використанням персонального комп'ютера відзначають наступні шкідливі та небезпечні фактори, які можуть загрожувати здоров'ю працівників на робочих місцях:

- несприятливі кліматичні (мікрокліматичні) умови у повітрі робочої зони;

- напружені зорові роботи;

- інтелектуальні навантаження;

- монотонність праці;

- нервово-емоційна напруженість праці;

- невідповідність ергономічних показників робочого місця діючим вимогам;

- фізична важкість виконуваної роботи (статичні навантаження на кістково-м'язовий апарат людини);

- шуми;

- електромагнітні випромінювання;

- ризики ураження електричним струмом;

- ризики виникнення пожеж;

- недостатня, або надмірна освітленість робочого місця;

- ризик виникнення надзвичайних ситуацій природного або штучного характеру на об'єкті або території.

8.2 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Приміщення, що буде розглядатися, в якому знаходиться робоче місце програміста має наступні геометричні розміри (площа, обсяг) і кількість працюючих у ньому людей дорівнює двом чоловікам. Вікно (1,5х2,0) кімнати орієнтовано на схід. Розміри аналізованого приміщення приведені в таблиці 8.1.

Таблиця 8.1 - Розміри приміщення

Найменування	Позначення	Значення, м
Довжина	А	6
Ширина	В	3
Висота	Н	3.0

Таблиця 8.2 - Площа та обсяг приміщення, на одного працюючого

Геометрична характеристика	Одиниця виміру	Нормативне значення	Фактичне значення
Площа, S	$м^2$	не менше 6.0	9.0
Обсяг, V	$м^3$	не менше 20.0	27.0

За даними, наведеними у таблиці 8.2., можна зробити висновок, що площа та об'єм для одного комп'ютеризованого робочого місця в даному приміщенні відповідають нормативним вимогам СанПіН 3.3.2 - 007 - 98.

У приміщенні розташовано 2 комп'ютера. Напруга джерела живлення комп'ютерів у приміщенні - 220 В. У приміщенні розміщені 2 комп'ютеризованих робочих місця, одна шафа для зберігання документів, один холодильник.

За небезпекою ураження електричним струмом управлінське приміщення відділу належить до приміщень без підвищеної небезпеки ураження електричним струмом працюючих.

Що стосується санітарно - гігієнічних показників у приміщенні, то умови праці при роботі на комп'ютері характеризуються можливістю впливу на нього наступних виробничих факторів: шуму, тепловиділень, іонізуючих та неіонізуючих випромінювань, специфічних умов зорової роботи.

Повітряне середовище в приміщенні характеризується мікрокліматом, запиленістю повітря та його загазованістю. Мікроклімат приміщення визначається діючим на організм людини поєднанням температури, відносної вологості, швидкості руху повітря та інтенсивності теплового випромінювання. Аналіз мікроклімату складається з виміру зазначених вище параметрів і зіставлення результатів із встановленими нормами.

Температура повітря в приміщенні визначається температурою зовнішнього повітря і тепловою енергією, що виділяється всередині приміщення. Джерелами теплоти в даному приміщенні є люди,

електроустаткування, а також освітлювальні прилади в темний час доби. Зовнішнім джерелом надлишкового тепла є сонячна радіація у світлий час доби. Робота, виконувана в даному приміщенні, відноситься до категорії I-а. Людиною в цьому випадку виділяється до 120 ккал. теплової енергії в годину. Вологість повітря в приміщенні визначається вологістю атмосферного і видихуваного людьми повітря, а також випарами з поверхні шкіри.

У таблиці 8.3 наведені оптимальні значення параметрів мікроклімату для категорії ваги робіт I-а, а також фактичні значення цих параметрів у розглянутому приміщенні. У приміщеннях з використанням обчислювальної техніки рекомендується застосування тільки оптимальних значень показників мікроклімату, тобто таких, при яких людина почуває себе комфортно.

Таблиця 8.3 - Оптимальні і фактичні значення параметрів мікроклімату

Пора року	Оптимальні для Іа			Фактичні		
	Температура, °С	Вологість,%	Швидкість повітря, м/с	Температура,°С	Вологість%	Швидкість повітря, м/с
Тепла	23-25	50-70	0,1	24-25	40-50	0,15
Холодна	22-24	40-60	0,1	23-25	40-50	0,1

Таким чином, показники мікроклімату в приміщенні, загалом, відповідають установленим нормам. У холодний період року використовується індивідуальне опалення, завдяки якому дотримується температурний режим у приміщенні в залежності від температури повітря навколишнього середовища.

Для підтримки температури і вологості повітря в літню пору встановлений кондиціонер, який має достатню потужність по холоду.

Джерелами запиленості повітря в приміщенні є одяг людей і пил, що проникає із вулиці. З метою боротьби з пилом робляться регулярні вологі прибирання і провітрювання.

У приміщенні немає виділення шкідливих газів. Тому що в ньому не проводяться монтажні роботи, пайки чи інші види робіт, при яких виділяються шкідливі гази.

Для нормалізації параметрів повітряного середовища також періодично здійснюється провітрювання приміщення і вологе прибирання. У всьому будинку діє встановлена загально обмінна витяжна вентиляція.

Опалення приміщення - центральне водяне від міських теплових мереж. Можливе використання електричного обігрівача середньої потужності на особливо холодний період часу і під час демісезонного відключення опалення.

Що стосується виробничого шуму, то у приміщенні перебувають такі джерела шуму: електродвигуни внутрішнього вентилятора ЕОМ; працюючий лазерний принтер; працюючі дисководи. Шум, вироблений вентилятором, можна класифікувати як постійний, всі інші джерела шуму - як імпульсні. Дане приміщення має паспорт, згідно з яким рівень звуку, Дб(А), обмірюваний за шкалою (А) шумоміра досяг величини 38,3 Дб(А) при роботі всього устаткування вузла, включаючи й принтер. Допустимий еквівалентний рівень шуму для робочого місця користувача ПК складає 65 дБА. (ДСанПіН 3.3.2-007-98 « Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин»). Допустимі параметри рівнів звуку та звукового тиску представлені в таблиці 8.4.

Таблиця 8.4 - Рівні звукового тиску від різних джерел

Джерело шуму	Рівень шуму, дБА
Жорсткий диск	45
Вентилятор	45
Принтер	55
Сканер	50

Це дозволяє зробити висновок про відповідність рівня звуку в приміщенні вимогам ДСанПіН 3.3.2-007-98.

Раціональне освітлення робочого місця є одним з найважливіших

факторів, що впливають на ефективність трудової діяльності людини, які попереджають травматизм і фахове захворювання програмістів. Освітлення на робочому місці програміста повинно бути таким, щоб працівник міг без напруги зору виконувати свою роботу.

Відповідно ДБН В.2.5 - 28 - 2006 роботу працівника, який постійно працює за комп'ютером, можна віднести до роботи з малою точністю (найменший розмір об'єкта розрізнення від 1 до 5 мм) V-го розряду зорової роботи, з великою контрастністю об'єкта розрізнення (символів на екрані дисплея), з темним тлом (під розряд зорової роботи В). Приміщення вузла можна віднести до 1-ої групи приміщень, у яких проводиться розрізнення об'єктів зорової роботи при фіксованому напрямку лінії зору того, що працює на робочу поверхню. Для такого типу приміщень і розряду зорової роботи нормоване значення коефіцієнта природної освітленості (КПО) робочої поверхні (при сполученому висвітленні), повинен становити не більше 1,5%, освітленість при штучному висвітленні повинна становити 300 лк.

За результатами виміру освітленості відділом охорони праці величина освітленості від системи загального штучного висвітлення лежить у межах 200-250 лк, що не відповідає вимогам, які пред'являються до даного приміщення.

8.3 Заходи профілактики при роботі з комп'ютерною технікою

Умови праці працівників у цілому відповідають існуючим санітарно-гігієнічним нормам. Для того, щоб особи, які працюють з ВДТ, меншою мірою втомлювались і зберігали високий рівень працездатності, потрібно раціонально організовувати їхні робочі місця.

З точки зору забезпечення електробезпеки до цих заходів можна віднести: устаткування розподільних щитів спеціальними розетками з заземлюючими контактами; організація заземлення всіх приладів і пристроїв;

періодична перевірка всіх приладів і пристроїв; щорічна здача іспитів з охорони праці.

З точки зору забезпечення оптимальних умов мікроклімату і освітленості до цих заходів можна віднести: організацію природної вентиляції, за допомогою дефлектора, для забезпечення необхідного повітрообміну в приміщенні; для забезпечення необхідних умов зорової роботи, що відповідають нормативним, оформлення паспорта на приміщення, з занесенням в нього вимірювань освітленості, проведених відділом охорони праці.

Таким чином, умови праці працівників у цілому відповідають існуючим санітарно-гігієнічним нормам. Але у зв'язку з тим, що більшу частину часу працівник займає сидячу позу і мало рухається, то для збереження здоров'я працівників, запобігання професійним захворюванням і підтримки працездатності слід дотримуватися вимог СанПіН 3.3.2.007-98 щодо режиму праці та відпочинку. Для цього призначаються регламентовані перерви для відпочинку.

Протягом робочого дня мають передбачатися:

- перерви для відпочинку і вживання їжі (обідні перерви);

- перерви для відпочинку і особистих потреб (згідно з трудовими нормами);

- додаткові перерви, що вводяться для окремих професій з урахуванням особливостей трудової діяльності.

В окремих випадках, при постійних скаргах на зорове стомлення тих, хто працює перед відеотерміналом, при дотриманні санітарно-гігієнічних вимог до режиму праці та відпочинку, а також вимог щодо застосування індивідуальних засобів локального захисту очей, допускається індивідуальний підхід до обмеження тривалості робіт перед відеотерміналом, зміни змісту роботи, чергування з іншими видами діяльності, не пов'язаними з відеотерміналом.

З метою зниження нервово-емоційного напруження, стомлення зорового аналізатора, поліпшення мозкового кровообігу, подолання несприятливих наслідків гіподинамії, запобігання втоми ДСанПіН 3.3.2.007-98 рекомендується деякі перерви використовувати для психофізіологічного розвантаження.

8.4 Розрахунок та проектування інженерно-технічного заходу захисту від шкідливого (небезпечного) виробничого фактору (освітленість приміщення)

Пропонуються наступні покращення умов праці на даному робочому місці: покращення умов штучного освітлення (ДБН В.2.5-28-2006 - "Природне та штучне освітлення)".

Розрахунок штучного освітлення проведемо для кімнати площею 18 м^2, ширина якої складає 3 м, довжина - 6 м, висота - 3,0 м. Скористаємося методом використання світлового потоку.

Для визначення потрібної кількості світильників, які повинні забезпечити нормований рівень освітленості, визначимо світловий потік, що падає на робочу поверхню за формулою:

$$F = \frac{E \cdot K \cdot S \cdot Z}{n}, \qquad (8.1)$$

де F - світловий потік, що розраховується, Лм;

E - нормована мінімальна освітленість, Лк; E = 300 Лк;

S - площа освітлюваного приміщення (у нашому випадку S=18м^2);

Z - відношення середньої освітленості до мінімальної (зазвичай приймається рівним 1,1...1,2, в нашому випадку Z =1,1);

K - коефіцієнт запасу, що враховує зменшення світлового потоку лампи в результаті забруднення світильників упроцесі експлуатації (його значення залежить від типу приміщення і характеру робіт, що проводяться в ньому, в нашому випадку K = 1,5);

n - коефіцієнт використання світлового потоку, (виражається відношенням світлового потоку, що падає на розрахункову поверхню, до сумарного потоку всіх ламп, і обчислюється в долях одиниці; залежить від характеристик світильника, розмірів приміщення, забарвлення стін і стелі, що характеризуються коефіцієнтами відбиття від стін ($\rho_{СТ.}$) і стелі ($\rho_{СТЕЛІ}$)), значення коефіцієнтів дорівнюють ρ = 40% і ρстелі=60%.

Обчислимо індекс приміщення за формулою:

$$I = \frac{S}{h(A+B)}, \tag{8.2}$$

де S - площа приміщення, S = 18 м^2;

h - розрахункова висота підвісу, h = 3,0 м;

A - ширина приміщення, A = 3 м;

B - довжина приміщення, B = 6 м.

Підставивши значення,отримаємо:

$$I = \frac{18}{3 \cdot (3+6)} = 0{,}66$$

Знаючи індекс приміщення I, за таблицею знаходимо п =0,25. Підставимо всі значення у формулу для визначення світлового потоку F:

$$F = \frac{300 \cdot 1{,}5 \cdot 18 \cdot 1{,}1}{0{,}25} = 35640 \text{ Лм}$$

Для освітлення використані люмінесцентні лампи типу ЛБ 40-1, світловий потік яких F = 4320 Лм.

Розрахуємо необхідну кількість ламп у світильниках за формулою:

$$N = \frac{F}{Fл}, \tag{8.3}$$

де *N* - визначуване число ламп;

F - світловий потік, *F* = 35640 Лм;

Fл - світловий потік лампи, *Fл* = 4320 Лм.

$$N = \frac{35640}{4320} = 8{,}25$$

У приміщенні використовуються світильники типу ОД. Кожен світильник комплектується двома лампами. Тобто необхідно використовувати 4 світильника із 8 працюючими лампами в них. На момент атестації робочого місця оператора працювало 6 ламп, тому рівень штучного освітлення не відповідав санітарним нормам.

Для покращення умов праці рекомендуємо збільшити рівень загальної освітленості приміщення шляхом встановлення 2 додаткових ламп.

Висновки

Професія програміста характеризується наявністю багатьох шкідливих та небезпечних факторів, а саме: недостатній або надмірний рівень освітленості, умови мікроклімату, рівень шуму, випромінювання, особливості фізичного навантаження організму тощо.

Була проведена робота по дослідженню шкідливих і небезпечних умов праці на робочому місці програміста та дослідженню ергономічних умов праці на цьому ж робочому місці.

Розглянувши тему освітлення робочого місця, можна зробити висновок, що працездатність кожного співробітника залежить не тільки від правильно організованого трудового процесу і від внутрішніх відносин у колективі, але і від того, як організовані службові приміщення в цілому і робоче місце даного співробітника, зокрема його раціональне освітлення.

Запропоновані технічні вирішення щодо забезпечення раціонального освітлення приміщень з ЕОМ та проведено розрахунок необхідної кількості світильників у даному приміщенні, щоб забезпечити достатній рівень освітленості на робочому місці програміста.

9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання магістерської роботи, призначене для захисту даних на основі генерації псевдовипадкових послідовностей.

У межах України в недостатній мірі представлені вітчизняні розробки в цій галузі.

У магістерській роботі наведено теоретичне узагальнення й вирішення наукового завдання дослідження методів захисту даних на основі генерації псевдовипадкових послідовностей.

Вирішення даного завдання полягало у здійсненні наступних задач:

- був проведено огляд існуючих систем захисту даних;

- досліджена система захисту даних на основі генерації псевдовипадкових послідовностей;

- на основі отриманих результатів досліджень створена програмна реалізація системи захисту даних на основі генерації псевдовипадкових послідовностей.

Розроблені під час виконання магістерської роботи алгоритми дозволяють успішно вирішувати завдання захисту даних на основі генерації псевдовипадкових послідовностей.

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 Windows XP/Vista/7/8/10.

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.

Розроблена програма має реальний економічний ефект від її впровадження у виробництво, а у користувача програмної продукції величина економічного ефекту становить 11777 грн., період окупності додаткових капітальних вкладень - 0,8 роки.

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. 500 кращих програм для Windows Уваров Сергій Сергійович Шифрування даних [Електронний ресурс] - Режим доступу: https://it.wikireading.ua/42304

4. Порівняння можливостей шифрування архіваторів WinRar та WinZip [Електронний ресурс] - Режим доступу: https://studfile.net/preview/5470392/page:12/

5. Архіватори WinRar і WinZip[Електронний ресурс] - Режим доступу: http://pro-computer.pp.ua/5779-arhvatori-winrar-winzip.html

6. Security Software for Windows OS [Електронний ресурс] - Режим доступу: www.secureaction.com

7. Обзор Advanced Encryption Package [Електронний ресурс] - Режим доступу: https://soft.mydiv.net/win/download-Advanced-Encryption-Package.html

8. CryptoExpert [Електронний ресурс] - Режим доступу: https://www.yourbestsoft.com/cryptoexpert/

9. Crypto Systems [Електронний ресурс] - Режим доступу: www.crypto-systems.com

10. softeza [Електронний ресурс] - Режим доступу: www.softeza.com

11. Max File Encryption [Електронний ресурс] - Режим доступу: https://www.softportal.com/software-4325-max-file-encryption.html

12. Огляд утиліти Dekart Private Disk Multifactor [Електронний ресурс] - Режим доступу: https://www.ixbt.com/soft/dekart-private-disk-multifactor.shtml

13. Universal Shield [Електронний ресурс] - Режим доступу: https://www.ixbt.com/news/soft/index.shtml?02/33/17

14. Folder Guard Professional [Електронний ресурс] - Режим доступу: https://1progs.ua/folder-guard/

15. Folder Guard [Електронний ресурс] - Режим доступу: https://soft.sibnet.ua/soft/16702-folder-guard-8-1/

16. Folder Guard [Електронний ресурс] - Режим доступу: https://manshet.org/168-folder-guard-18102425.html

17. Методи сучасної криптографії / Дегтярьов А. К. [Електронний ресурс] - Режим доступу: -https://dehtyarov09.wordpress.com/

18. Криптографія: загальні визначення, класифікація. асиметричні та симетричні криптоалгоритми. [Електронний ресурс] - Режим доступу: - https://dehtyarov09.wordpress.com/2014/03/16/криптографія-загальні-визначення-кл-2/

19. Інформатика для економістів/ Криптографічні методи захисту інформації [Електронний ресурс] - Режим доступу: - https://stud.com.ua/53399/informatika/kriptografichni_metodi_zahistu_informatsiyi

20. Телекомунікаційні системи та мережі. Том 1. Структура й основні функції. -[Електронний ресурс] - Режим доступу: http://www.znanius.com/3851.html

21. Електронний підручник з інформатики [Електронний ресурс] - Режим доступу: http://informatics.dp.ua/struktura-programy-delphi-elementy-progra

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

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

24. Захист програмного забезпечення [Електронний ресурс] - Режим доступу: http://ua-referat.com/Захист_програмного_забезпечення

25. Захист програм від злому [Електронний ресурс] - Режим доступу: http://easy-code.com.ua/2011/04/zaxist-program-vid-zlomu/

26. Сайт фірми Brain [Електронний ресурс] - Режим доступу: http://brain.com.ua

27. Буняк В. М., Основи криптографії: навчальний посібник. — К.: "Мета", 2020.

28. General Data Protection Regulation (GDPR). Офіційний сайт ЄС. [Електронний ресурс] - Режим доступу: https://gdpr.eu.

29. Bruce Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C. - John Wiley & Sons, 2015.

30. Stinson, D. R. Cryptography: Theory and Practice. - CRC Press, 2006.

31. Литвиненко І. В. Методи шифрування даних в інформаційних системах. - Одеса: ОНУ ім. І. І. Мечникова, 2021.

32. Кравець М. О. Розробка програмного забезпечення для шифрування даних. - Вінниця: ВНТУ, 2022.

33. Захист програмного забезпечення [Електронний ресурс] - Режим доступу: http://ua-referat.com/Захист_програмного_забезпечення

34. Сайт фірми Brain [Електронний ресурс] - Режим доступу: http://brain.com.ua

35. Мелешко Є.В., Якименко М.С., Поліщук Л.І. Алгоритми та структури даних: Навчальний посібник для студентів технічних спеціальностей денної та заочної форми навчання. – Кропивницький: Видавець – Лисенко В.Ф., 2019. – 156 стор.

36. Власій О.О. Алгоритми та структури даних: Лабораторний практикум. – Івано-Франківськ: ДВНЗ «Прикарпатський національний університет імені Василя Стефаника», 2015. – 68 с.

37. Захист програм від злому [Електронний ресурс] - http://easy-code.com.ua/2011/04/zaxist-program-vid-zlomu/

38. Alexander Osterwalder, Yves Pigneur – Business Model Generation. Wiley, 2010. – 288 стор.

39. Steve Blank, Bob Dorf – The Startup Owner's Manual: The Step-by-Step Guide for Building a Great Company. K&S Ranch, 2012. – 608 стор.

40. Philip Kotler – Marketing Management (15th Edition). Pearson, 2015. –

832 стор.

41. Kenneth C. Laudon, Jane P. Laudon – Management Information Systems: Managing the Digital Firm (16th Edition). Pearson, 2020. – 688 стор.

42. Жидецький В.Ц. Основи охорони праці: підруч. 3-є вид., перероб і доп. Львів : УАД, 2006. 336 стор.

43. Босов Є.П., Жесан Р.В., Каліч В.М., Голик О.П., Зубенко В.О. Охорона праці при проектуванні систем автоматизації виробництва : навч. посіб. 2-е вид., перероб. і доп. Кропивницький : ЦНТУ, 2022. – 208 стор.

44. Конституція України. [Електронний ресурс] / Режим доступу: https://zakon.rada.gov.ua/laws/main/254%D0%BA/96-%D0%B2%D1%80

45. Про охорону праці : Закон України. [Електронний ресурс] / Режим доступу: https://zakon.rada.gov.ua/laws/main/2694-12#Text.

46. Основи законодавства України про охорону здоров'я : Закон України. [Електронний ресурс] / Режим доступу: https://zakon.rada.gov.ua/laws/show/2801-12#Text.

47. Закон України «Про захист інформації в інформаційно-телекомунікаційних системах» [Електроний ресурс]. – Режим доступу до ресурсу: http://zakon4.rada.gov.ua/laws/show/2594-15

48. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин: ДСанПІН 3.3.2-007-98. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/rada/show/v0007282-98

49. Захист програмного забезпечення [Електронний ресурс] - Режим доступу: http://ua-referat.com/Захист_програмного_забезпечення

50. Захист програм від злому [Електронний ресурс] - Режим доступу: http://easy-code.com.ua/2011/04/zaxist-program-vid-zlomu/

Додаток А
(обов'язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на дослідження та програмну реалізацію системи реалізація системи захисту даних.

2 Підстава для розробки

Підставою для розробки служить завдання на магістерську роботу, видане на кафедрі програмування та захисту інформації (нак. №19-13 від 07.08.2024 року).

3 Мета та призначення розробки

Метою магістерської роботи є дослідження та програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів.

4 Джерела розробки

Джерелом цієї магістерської роботи є стосовна до теми література й 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;

- техніко-економічне обґрунтування доцільності прийнятого до розробки програмного забезпечення;

- аналіз умов праці;

- розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

- системний захист даних на основі генерації псевдовипадкових;

- 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 найбільш поширені.

5.5 Вимоги до надійності

Програмні модулі написані за всіма правилами, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні відповідати 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 сімейства Windows 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 сімейства Windows.

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

Середовище Delphi версії 7.

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 Економічні вимоги

7.1 Для ПЗ необхідно виробити функціонально-вартісний аналіз варіантів розробки.

7.2 Виконати розрахунок витрат показників економічного ефекту з урахуванням цін на 3 вересня 2024 року.

8 Вимоги щодо охорони праці

У частині охорони праці магістерської роботи повинні бути розглянуті рекомендації, щодо зменшення шкідливого впливу периферійної техніки, яка за звичай знаходиться в приміщенні, де працює програміст.

9 Перелік документів, що розробляються

- Наукова новизна - 1 аркуш.
- Структурна схема системи - 1 аркуш.
- Функціональна схема системи - 1 аркуш.
- Діаграма процесів - 1 аркуш.
- Блок-схема алгоритму роботи програми - 3 аркуша.
- Показники економічної ефективності - 1 аркуш.
- Пояснювальна записка - 90 аркушів.

10 Етапи розробки

10.1 Збір і обробка інформації з теми магістерської роботи. Постановка задачі на виконання магістерської роботи (складання ТЗ).

10.2 Проведення досліджень або експериментальних робіт для уточнення основних положень магістерської роботи.

10.3 Розробка функціональних схем, блок - схем алгоритмів роботи програмного забезпечення.

10.4 Побудова схем взаємодії даних.

10.5 Створення прототипу ПЗ.

10.6 Віднаходження ПЗ, аналіз отриманих результатів.

10.7 Робота над питанням охорони праці і техніки безпеки.

10.8 Розрахунок з техніко-економічного обґрунтування.

10.9 Оформлення пояснювальної записки і виконання робіт по графічній частині.

11 Порядок контролю та приймання

11.1 Подання магістерської роботи на попередній захист 02.12.2024 р.

11.2 Подання магістерської роботи на захист _____20__ р.

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник магістерської роботи

_____ Кислун О.А.

Дослідження та програмна реалізація системи захисту персональних даних із застосуванням псевдовипадкових алгоритмів

Лістинг програми

Код документу 12

Носій: DVD-RW диск

Загальна кількість аркушів: 21

Літера: РП

Кропивницький 2024

```
// -------------------- u.pas -------------------- Програмний модуль
unit u;

interface

uses

  Windows, Messages,
  SysUtils, Variants,
  Classes, Graphics,
  Controls, Forms,
  Dialogs,  StdCtrls,
  ComCtrls, Registry;

type

  TForm1 = class(TForm)

    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Button1: TButton;
    TabSheet2: TTabSheet;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Edit5: TEdit;
    Edit6: TEdit;
    Edit7: TEdit;
    Button2: TButton;
    TabSheet3: TTabSheet;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Edit8: TEdit;
    Edit9: TEdit;
    Edit10: TEdit;
    Button3: TButton;
    TabSheet4: TTabSheet;
    Edit11: TEdit;
    Edit13: TEdit;
    Edit12: TEdit;
    Button4: TButton;
    TabSheet5: TTabSheet;
    TabSheet7: TTabSheet;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    OpenDialog2: TOpenDialog;
    SaveDialog2: TSaveDialog;
    OpenDialog3: TOpenDialog;
    OpenDialog4: TOpenDialog;
    SaveDialog3: TSaveDialog;
    OpenDialog5: TOpenDialog;
    OpenDialog6: TOpenDialog;
    SaveDialog4: TSaveDialog;
    Label10: TLabel;
    Label11: TLabel;
```

```
    Label12: TLabel;
    Memo1: TMemo;
    Button5: TButton;
    Edit4: TEdit;
    Button6: TButton;

    procedure FormCreate(Sender: TObject);
    procedure O1(Sender: TObject);
    procedure S1(Sender: TObject);
    procedure E3(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure O2(Sender: TObject);
    procedure S2(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure O3(Sender: TObject);
    procedure O4(Sender: TObject);
    procedure S3(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure O5(Sender: TObject);
    procedure O6(Sender: TObject);
    procedure S4(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure SIZ(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure Button6Click(Sender: TObject);

   private

   { Private declarations }

  public

    { Public declarations }

  end;

var

  Form1: TForm1;
  pasword:string;
  SIZE,language:integer;
  AU,B:boolean;

implementation

{$R *.dfm}

//////////////////////////////////////////
procedure TForm1.SIZ(Sender: TObject);
begin
          pAGEcONTROL1.width:=form1.width;

          edit1.width:=form1.width-70;
          edit2.width:=form1.width-70;
          edit3.width:=form1.width-70;
          edit5.width:=form1.width-70;
          edit6.width:=form1.width-70;
          edit7.width:=form1.width-70;
          edit8.width:=form1.width-70;
          edit9.width:=form1.width-70;
          edit10.width:=form1.width-70;
          edit11.width:=form1.width-70;
          edit12.width:=form1.width-70;
```

```
memo1.width:=form1.width-60;
memo1.Height:=form1.Height-100;
memo1.Top:=5;

memo1.left:=20;

edit13.width:=form1.width-70;

edit1.Height:=trunc(0.1*form1.Height);
edit2.Height:=trunc(0.1*form1.Height);
edit3.Height:=trunc(0.1*form1.Height);
edit5.Height:=trunc(0.1*form1.Height);
edit6.Height:=trunc(0.1*form1.Height);
edit7.Height:=trunc(0.1*form1.Height);
edit8.Height:=trunc(0.1*form1.Height);
edit9.Height:=trunc(0.1*form1.Height);
edit10.Height:=trunc(0.1*form1.Height);
edit11.Height:=trunc(0.1*form1.Height);
edit12.Height:=trunc(0.1*form1.Height);
edit13.Height:=trunc(0.1*form1.Height);

edit1.Top:=trunc(0.1*form1.Height-10);
edit2.Top:=trunc(0.30*form1.Height-10);
edit3.Top:=trunc(0.50*form1.Height-10);
edit5.Top:=trunc(0.1*form1.Height-10);
edit6.Top:=trunc(0.3*form1.Height-10);
edit7.Top:=trunc(0.5*form1.Height-10);
edit8.Top:=trunc(0.1*form1.Height-10);
edit9.Top:=trunc(0.3*form1.Height-10);
edit10.Top:=trunc(0.50*form1.Height-10);
edit11.Top:=trunc(0.1*form1.Height-10);
edit12.Top:=trunc(0.3*form1.Height-10);
edit13.Top:=trunc(0.5*form1.Height-10);

pAGEcONTROL1.Height:=form1.Height;

Button1.Top:=trunc(0.6*form1.Height-10);
Button2.Top:=trunc(0.6*(form1.Height-10));
Button3.Top:=trunc(0.6*(form1.Height-0));
Button4.Top:=trunc(0.6*(form1.Height-10));

Button1.Height:=trunc(0.1*form1.Height);
Button2.Height:=trunc(0.1*form1.Height);
Button3.Height:=trunc(0.1*form1.Height);
Button4.Height:=trunc(0.1*form1.Height);

Button1.width:=form1.width-70;
Button2.width:=form1.width-70;
Button3.width:=form1.width-70;
Button4.width:=form1.width-70;

LABEL1.Top:=trunc(0.01*form1.Height-3);
LABEL2.Top:=trunc(0.215*form1.Height-3);
LABEL3.Top:=trunc(0.41*form1.Height-3);
LABEL4.Top:=trunc(0.01*form1.Height-3);
LABEL5.Top:=trunc(0.215*form1.Height-3);
LABEL6.Top:=trunc(0.41*form1.Height-3);
LABEL7.Top:=trunc(0.01*form1.Height-3);
LABEL8.Top:=trunc(0.215*form1.Height-3);
LABEL9.Top:=trunc(0.41*form1.Height-3);
LABEL10.Top:=trunc(0.01*form1.Height-3);
LABEL11.Top:=trunc(0.215*form1.Height-3);
LABEL12.Top:=trunc(0.41*form1.Height-3);
```

```
          LABEL1.Left:=25;
          LABEL2.Left:=25;
          LABEL3.Left:=25;
          LABEL4.Left:=25;
          LABEL5.Left:=25;
          LABEL6.Left:=25;
          LABEL7.Left:=25;
          LABEL8.Left:=25;
          LABEL9.Left:=25;
          LABEL10.Left:=25;
          LABEL11.Left:=25;
          LABEL12.Left:=25;
END;
////////////////////////////////////////////

////////////////////////////////////////////при открытии формы
procedure TForm1.FormCreate(Sender: TObject);

var
  VolumeName,
  FileSystemName : array [0..MAX_PATH-1] of Char;
  VolumeSerialNo,snr : DWord;
  MaxComponentLength,FileSystemFlags: Cardinal;
  reg:TRegistry;
  RDate:tDate;
  j:integer;

begin
//чтение номера диска
GetVolumeInformation('C:\',VolumeName,MAX_PATH,@VolumeSerialNo,
MaxComponentLength,FileSystemFlags, FileSystemName,MAX_PATH);
reg:=TRegistry.Create;
reg.RootKey:=HKEY_LOCAL_MACHINE;
reg.OpenKey('HardWare', true);
snr:=REG.readInteger('SN');
if reg.OpenKey('HardWare/data', true)
then
begin
 reg.writeDate('HardWare/data',date);
 reg.writeInteger('HardWare/K',1);
end;
if  snr=VolumeSerialNo
then
begin
 Au:=true;
 Button5.caption:=' А К Т И В О В А Н О ';
 Button4.caption:=' А К Т И В О В А Н О ';
end
else
 begin
     Au:=false;
    Form1.caption:='СИСТЕМА ЗАХИСТУ ДАНИХ - незареєстрована версія';
    reg.CloseKey;
    reg.Free;
     reg:=TRegistry.Create;
     reg.RootKey:=HKEY_LOCAL_MACHINE;
     reg.OpenKey('HardWare', false);
     //RDate:=reg.readdate('data');
    // j:=reg.readinteger('K');
if (Rdate>date+30) or (j<20)
     then
     begin
     b:=true;
     Form1.caption:='СИСТЕМА ЗАХИСТУ ДАНИХ - БЕТАРЕЖИМ';
```

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

// назва вкладок
          edit1.text:='';
          edit2.text:='';
          edit3.text:='';
          edit5.text:='';
          edit6.text:='';
          edit7.text:='';
          edit8.text:='';
          edit9.text:='';
          edit10.text:='';
          edit11.text:='';
          edit12.text:='';
          edit13.text:='';

          edit1.left:=25;
          edit2.left:=25;
          edit3.left:=25;
          edit5.left:=25;
          edit6.left:=25;
          edit7.left:=25;
          edit8.left:=25;
          edit9.left:=25;
          edit10.left:=25;
          edit11.left:=25;
          edit12.left:=25;
          edit13.left:=25;

          form1.width:=500;
          edit1.width:=400;
          edit2.width:=800;
          edit3.width:=400;

      if language=1
   then
       begin
          TabSheet1.caption:=' Кодирование ';
          TabSheet2.caption:=' Декодирование ';
          TabSheet3.caption:=' Утаивание ';
          TabSheet4.caption:=' Извлечение ';
          TabSheet5.caption:=' Справка ';
          TabSheet7.caption:=' Активация ';
// кнопки
          Button1.caption:=' Кодировать ';
          Button2.caption:=' Декодировать ';
          Button3.caption:=' Спрятать ';
          Button4.caption:=' Извлечь ';
// мітки
          label1.caption:=' Кодируемый файл:';
          label2.caption:=' Закодированный файл:';
          label3.caption:=' Пароль:';
          label4.caption:=' Декодируемый файл:';
          label5.caption:=' Расшифрованный файл:';
          label6.caption:=' Пароль:';
          label7.caption:=' Секретный файл:';
          label8.caption:=' Файл, я котором прячется';
          label9.caption:=' Контейнерный файл:';
          label10.caption:=' Контейнерный файл:';
          label11.caption:=' Файл, в котором прячется:';
          label12.caption:=' Файл, который прячется';
```

```
       end
 else
 if language=2
    then begin end
    else
         begin
          TabSheet1.caption:=' Кодування ';
          TabSheet2.caption:=' Декодування ';
          TabSheet3.caption:=' Приховування ';
          TabSheet4.caption:=' Відтворення ';
          TabSheet5.caption:=' Довідка ';
          TabSheet7.caption:=' Активація ';
// кнопки
          Button1.caption:=' Кодувати ';
          Button2.caption:=' Декодувати ';
          Button3.caption:=' Приховати ';
          Button4.caption:=' Відтворити ';
// мітки
          label1.caption:=' Файл, що кодується:';
          label2.caption:=' Закодований файл:';
          label3.caption:=' Пароль:';
          label4.caption:=' Файл, що декодується:';
          label5.caption:=' Розшифровний файл:';
          label6.caption:=' Пароль:';
          label7.caption:=' Файд, що приховується';
          label8.caption:=' Файд, я якому приховується';
          label9.caption:=' Контейнерний файл в містом';
          label10.caption:=' Контейнернпй файл в містом';
          label11.caption:=' Файл, я якому приховується';
          label12.caption:=' Файл, що приховували';

          Memo1.Lines[0]:='До методів захисту інформації відносятьметоди
захисту даних із використанням шифрування та /або приховування.';
          Memo1.Lines.Add('Мета шифрування досягається за рахунок системи
криптографічного захисту - зміною кодування даних, що перешкоджає
безпосередньому доступі до інформації.');
          Memo1.Lines.Add('Приховування - стеганографія - тайнопис, при якому
повідомлення, закодоване таким чином, що не виглядає як повідомлення. То ж
криптографія приховує зміст повідомлення, то стеганографія приховує сам факт
існування повідомлення.');
         Memo1.Lines.Add('Дана програма нажахає можливість скористатися обома
методами.');
         Memo1.Lines.Add('Технічно для кожного з методів наявні два
процеси/');
         Memo1.Lines.Add('Шифруванню: кодування та декодування, для чого
необхідно задати ключ (пароль); стеганогра?фії : приховування та відтворення
для чого необхідний файл контейнер, в якому приховується файл з інформацією .
що підлягає  збереженню.');
         Memo1.Lines.Add('Даний програмний продукт забезпечує  всіх чотири
процеси.');
          Memo1.Lines.Add('Програмне забезпечення функціонувати під під
управлінням ОС сімейства Windows.');
           end;
 end;
//////////////////////////////////////////////

//////////////////////////////////////////////
procedure TForm1.O1(Sender: TObject);

begin
Button1.caption:=' Кодувати ';
with opendialog1 do
begin
if  not execute
```

```
then exit;
end;
edit1.
text:=opendialog1.FileName;
end;
//////////////////////////////////////////////

//////////////////////////////////////////////
procedure TForm1.S1(Sender: TObject);

begin
with savedialog1 do
begin
if  not execute
then
exit;
end;
edit2.text:=savedialog1.FileName;
end;
//////////////////////////////////////////////

////////////////////////////////////////////// генерація  пслдовності
procedure GP;

var
  PvPFile : TextFile;
  i,j:integer;
  ch:char;
  p:string;

begin
  p:=pasword;
  AssignFile(PvPFile, '\PVP.PVP');
  ReWrite(PvPFile);
  for j:=1 to 500 do
  begin
  ch:=p[1];
  for i:=1 to Length(pasword)-1 do
       p[i]:=p[i+1];
       p[Length(pasword)]:=chr((ord(ch)+1)*(ord(p[Length(pasword)-1])+1) mod
255);
   Write(PvPFile,ch);
   end;
  CloseFile(PvPFile);
 end;
//////////////////////////////////////////////

//////////////////////////////////////////////
procedure TForm1.Button1Click(Sender: TObject);

var
KodFile,Pfile,PvPFile: TextFile;
  p,pvp:char;
  i:integer;

begin
if (Au) or (B)
then
begin
pasword:= edit3.text;
if (edit3.text='') or (Length(pasword)<2)
then
Button1.caption:=' Введіть пароль довший за один символ '
else
```

```
begin
  AssignFile(PFile, edit1.text);
  if FileExists(edit1.text)=false
    then
      Button1.caption:=' Файл що кодується відсутній '
    else
       begin
       ReSet(PFile);
       gp;
       AssignFile(KodFile, edit2.text);
       ReWrite(KodFile);
       AssignFile(PvPFile, 'D:\PVP.PVP');
       ReSet(PvPFile);
       i:=1;
       while (not EOF(PFile))
          do
          begin
          Read(PFile,p);
          Read(PvpFile,pvp);
          Write(KodFile,chr((ord(p)+ord(pvp))));
          i:=i+1;
          if i>500
          then
          begin
       ReSet(PvPFile);
       i:=1;
        end;
     end;
  CloseFile(pvPFile);
  CloseFile(PFile);
  CloseFile(KodFile);
  DeleteFile('D:\PVP.PVP');
  end;
 end;
end;
end;
//////////////////////////////////////////

//////////////////////////////////////////
procedure TForm1.E3(Sender: TObject);

begin
Button1.caption:=' Шифрувати ';
end;
//////////////////////////////////////////

////////////////////////////////////////// декодування

//////////////////////////////////////////
procedure TForm1.O2(Sender: TObject);

begin
with opendialog2 do
begin
if not execute
then
exit;
end;
edit5.text:=opendialog2.FileName;
end;
//////////////////////////////////////////

//////////////////////////////////////////
procedure TForm1.S2(Sender: TObject);
```

```
begin
with savedialog2 do
begin
if  not execute
then
exit;
end;
edit6.text:=savedialog2.FileName;
end;
//////////////////////////////////////////

//////////////////////////////////////////
procedure TForm1.Button2Click(Sender: TObject);

var
KodFile,Pfile,PvPFile: TextFile;
  p,pvp:char;
  i:integer;

begin
if (Au) or (B)
then
begin
pasword:= edit7.text;
if (edit7.text='') or (Length(pasword)<2)
then
Button2.caption:=' Введіть пароль довший за один символ '
else
begin
gp;
begin
  AssignFile(KodFile, edit6.text);
  ReWrite(KodFile);
  AssignFile(PFile, edit5.text);
  ReSet(PFile);
  AssignFile(PvPFile, 'PVP.PVP');
  ReSet(PvPFile);
  i:=1;
while (not EOF(PFile))
  do
  begin
  Read(PFile,p);
  Read(PvpFile,pvp);
  Write(KodFile,chr((ord(p)-ord(pvp))));
  i:=i+1;
  if i>500
   then
    begin
    ReSet(PvPFile);
    i:=1;
    end;
  end;
  CloseFile(pvPFile);
  CloseFile(PFile);
  CloseFile(KodFile);
  DeleteFile('D:\PVP.PVP');
  end;
end;
end;
end;
//////////////////////////////////////////

//////////////////////////////////////////   Приховування
```

```
//////////////////////////////////////////////
procedure TForm1.O3(Sender: TObject);

begin
with opendialog3
do
begin
if  not execute
then
exit;
end;
edit8.text:=opendialog3.FileName;
end;
//////////////////////////////////////////////

//////////////////////////////////////////////
procedure TForm1.O4(Sender: TObject);

begin
with opendialog4 do
begin
if  not execute
then
exit;
end;
edit9.text:=opendialog4.FileName;
end;
//////////////////////////////////////////////

//////////////////////////////////////////////
procedure TForm1.S3(Sender: TObject);

begin
with savedialog3 do
begin
if  not execute
then
exit;
end;
edit10.text:=savedialog3.FileName;
end;
//////////////////////////////////////////////

//////////////////////////////////////////////
procedure TForm1.Button3Click(Sender: TObject);

var
Pfile,KodFile,KontFile: TextFile;
KPfile,KKodFile,i:integer;
p,kod:char;
Dp:string;

begin
if (Au) or (B)
then
begin
  AssignFile(PFile,edit8.text);
  ReSet(PFile);
  KPfile:=1;
  while (not EOF(PFile))
  do
  begin
  read(PFile,p);
```

```
  KPfile:=KPfile+1;
  end;
  AssignFile(KodFile,edit9.text);
  ReSet(KodFile);
  KKodfile:=1;
while (not EOF(KodFile))
  do
  begin
  read(KodFile,Kod);
  KKodFile:=KKodFile+1;
  end;
  if KPfile+20>KKodFile
   then
   Button3.caption:=' Файл, що приховується не поміщається в контейнер'
  else
  begin
  KPfile:=KPfile-1;
  Dp:=IntToStr(KPfile);
  Dp:=Dp+'                    ';
  AssignFile(KontFile,edit10.text);
  ReWrite(KontFile);
  ReSet(KodFile);
  ReSet(PFile);
  i:=1;
  while (not EOF(KodFile)
) do
  begin
  read(KodFile,Kod);
 if i<=20
   then
     Write(KontFile,chr(ord(kod)+ord(Dp[i]))) ;
if (i>20) and (i<=KPfile+20)
   then
    begin
    read(pFile,p);
    Write(KontFile,chr(ord(kod)+ord(p) )) ;
    end;
  if i>KPfile+20 then Write(KontFile,kod);
  i:=i+1;
  end;
   Button3.caption:=Dp;
   CloseFile(KontFile);
 end;
  CloseFile(PFile);
  CloseFile(KodFile);
end;
end;
////////////////////////////////////////////

////////////////////////////////////////////  Відтворення

////////////////////////////////////////////
procedure TForm1.O5(Sender: TObject);

begin
with opendialog5 do
begin
if  not execute
then
exit;
end;
edit11.text:=opendialog5.FileName;
end;
////////////////////////////////////////////
```

```
//////////////////////////////////////////
procedure TForm1.O6(Sender: TObject);

begin
with opendialog6
do
begin
if  not execute
then
exit;
end;
edit12.text:=opendialog6.FileName;
end;
//////////////////////////////////////////

//////////////////////////////////////////
procedure TForm1.S4(Sender: TObject);

begin
with savedialog4
do
begin
if  not execute
then
exit;
end;
edit13.text:=savedialog4.FileName;
end;
//////////////////////////////////////////

//////////////////////////////////////////
procedure TForm1.Button4Click(Sender: TObject);

var
Pfile,KodFile,KontFile: TextFile;
 Dp:string[20];
 i,j:integer;
 kont,kod:char;

begin
if (Au) or (B)
then
begin
  AssignFile(KontFile,edit11.text);
  ReSet(KontFile);
  AssignFile(KodFile,edit12.text);
  ReSet(KodFile);
  Dp:='';
  for i:=1 to 20
  do
  begin
  read(KontFile,Kont);
  read(KodFile,Kod);
  Dp:=Dp+chr(ord(Kont)-ord(kod));
  end;
   Dp:=Trim(Dp);
   j:=StrToInt(Dp);
   Dp:='';
   AssignFile(PFile,edit13.text);
   ReWrite(PFile);
   Button4.caption:=Dp;
  for i:=1 to j
  do
```

```
  begin
  read(KontFile,Kont);
  read(KodFile,Kod);
  Dp:=Dp+ chr(ord(Kont)-ord(kod));
   Write(PFile,chr(ord(Kont)-ord(kod)));
   end;
   CloseFile(PFile);
   CloseFile(KontFile);
   CloseFile(KodFile);
   Button4.caption:=Dp;
 end;
 end;
//////////////////////////////////////////

//////////////////////////////////////////
function Sn: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;
//////////////////////////////////////////

//////////////////////////////////////////
procedure TForm1.Button5Click(Sender: TObject);

var
   VolumeName,
  FileSystemName : array [0..MAX_PATH-1] of Char;
  VolumeSerialNo,snr : DWord;
  MaxComponentLength,FileSystemFlags: Cardinal;
  reg:TRegistry;

begin
if Au
 then begin   end
 else
 begin
GetVolumeInformation('C:\',VolumeName,MAX_PATH,@VolumeSerialNo,
MaxComponentLength,FileSystemFlags, FileSystemName,MAX_PATH);
reg:=TRegistry.Create;
reg.RootKey:=HKEY_LOCAL_MACHINE;
reg.OpenKey('HardWare', true);
//REG.WriteInteger('SN',StrToInt((Edit4.Text)));
//reg.OpenKey('HardWare', true);
snr:=REG.readInteger('SN');
if  snr=VolumeSerialNo
  then
  begin
     Au:=true;
     Form1.caption:='СИСТЕМА ЗАХИСТУ ДАНИХ';
  end
  else
  begin
     Au:=false;
    Form1.caption:='СИСТЕМА ЗАХИСТУ ДАНИХ - незареєстрована версія';
    Edit4.Text :=IntToHex(VolumeSerialNo+1,8);
```

```
   Button5.caption:=' А К Т И В У В А Т И   '+ datetostr(date);
  end;
reg.CloseKey;
reg.Free;
end;
end;
//////////////////////////////////////////

//////////////////////////////////////////
procedure TForm1.Button6Click(Sender: TObject);

var
  VolumeName,FileSystemName : array [0..MAX_PATH-1] of Char;
  VolumeSerialNo,snr : DWord;
  MaxComponentLength,FileSystemFlags: Cardinal;
  reg:TRegistry;

begin
 if Au
 then begin   end
 else
 begin
GetVolumeInformation('C:\',VolumeName,MAX_PATH,@VolumeSerialNo,
MaxComponentLength,FileSystemFlags, FileSystemName,MAX_PATH);
reg:=TRegistry.Create;
reg.RootKey:=HKEY_LOCAL_MACHINE;
reg.OpenKey('HardWare', true);
REG.WriteInteger('SN',VolumeSerialNo);
Au:=true;
Form1.caption:='СИСТЕМА ЗАХИСТУ ДАНИХ';
Button5.caption:=' А К Т И В О В А Н О ';
reg.CloseKey;
reg.Free;
end;
end;
end.
//////////////////////////////////////
```

```
// -------------------- Form1.dfm -------------------- Опис Форми Form1
bject Form1: TForm1
  Left = 360
  Top = 154
  Width = 574
  Height = 417
  Caption = 'ÑÈÑÒÅÌÀ ÇÀÕÈÑÒÓ ÄÀÍÈÕ'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  OnActivate = FormCreate
  OnCreate = FormCreate
  OnResize = SIZ
  PixelsPerInch = 96
  TextHeight = 13
  object PageControl1: TPageControl
    Left = -8
    Top = 8
    Width = 729
    Height = 345
    ActivePage = TabSheet7
    TabOrder = 0
    object TabSheet1: TTabSheet
      Caption = 'TabSheet1'
      object Label1: TLabel
        Left = 33
        Top = 8
        Width = 32
        Height = 13
        Caption = 'Label1'
      end
      object Label2: TLabel
        Left = 25
        Top = 96
        Width = 32
        Height = 13
        Caption = 'Label2'
      end
      object Label3: TLabel
        Left = 16
        Top = 184
        Width = 32
        Height = 13
        Caption = 'Label3'
      end
      object Edit1: TEdit
        Left = 25
        Top = 40
        Width = 440
        Height = 21
        TabOrder = 0
        Text = 'Edit1'
        OnClick = O1
      end
      object Edit2: TEdit
        Left = 25
        Top = 136
        Width = 448
        Height = 21
        TabOrder = 1
```

```
    Text = 'Edit2'
    OnClick = S1
  end
  object Edit3: TEdit
    Left = 24
    Top = 224
    Width = 441
    Height = 21
    TabOrder = 2
    Text = 'Edit3'
  end
  object Button1: TButton
    Left = 24
    Top = 256
    Width = 449
    Height = 49
    Caption = 'Button1'
    TabOrder = 3
    OnClick = Button1Click
  end
end
object TabSheet2: TTabSheet
  Caption = 'TabSheet2'
  ImageIndex = 1
  object Label4: TLabel
    Left = 24
    Top = 16
    Width = 32
    Height = 13
    Caption = 'Label4'
  end
  object Label5: TLabel
    Left = 24
    Top = 96
    Width = 32
    Height = 13
    Caption = 'Label5'
  end
  object Label6: TLabel
    Left = 24
    Top = 176
    Width = 32
    Height = 13
    Caption = 'Label6'
  end
  object Edit5: TEdit
    Left = 24
    Top = 56
    Width = 433
    Height = 21
    TabOrder = 0
    Text = 'Edit5'
    OnClick = O2
  end
  object Edit6: TEdit
    Left = 24
    Top = 136
    Width = 441
    Height = 21
    TabOrder = 1
    Text = 'Edit6'
    OnClick = S2
  end
  object Edit7: TEdit
```

```
      Left = 24
      Top = 216
      Width = 449
      Height = 21
      TabOrder = 2
      Text = 'Edit7'
    end
    object Button2: TButton
      Left = 24
      Top = 272
      Width = 449
      Height = 41
      Caption = 'Button2'
      TabOrder = 3
      OnClick = Button2Click
    end
  end
  object TabSheet3: TTabSheet
    Caption = 'TabSheet3'
    ImageIndex = 2
    object Label7: TLabel
      Left = 24
      Top = 8
      Width = 32
      Height = 13
      Caption = 'Label7'
    end
    object Label8: TLabel
      Left = 24
      Top = 80
      Width = 32
      Height = 13
      Caption = 'Label8'
    end
    object Label9: TLabel
      Left = 24
      Top = 160
      Width = 32
      Height = 13
      Caption = 'Label9'
    end
    object Edit8: TEdit
      Left = 24
      Top = 40
      Width = 457
      Height = 21
      TabOrder = 0
      Text = 'Edit8'
      OnClick = O3
    end
    object Edit9: TEdit
      Left = 24
      Top = 112
      Width = 457
      Height = 21
      TabOrder = 1
      Text = 'Edit9'
      OnClick = O4
    end
    object Edit10: TEdit
      Left = 24
      Top = 200
      Width = 449
      Height = 21
```

```
    TabOrder = 2
    Text = 'Edit10'
    OnClick = S3
  end
  object Button3: TButton
    Left = 24
    Top = 264
    Width = 449
    Height = 49
    Caption = 'Button3'
    TabOrder = 3
    OnClick = Button3Click
  end
end
object TabSheet4: TTabSheet
  Caption = 'TabSheet4'
  ImageIndex = 3
  object Label10: TLabel
    Left = 32
    Top = 8
    Width = 38
    Height = 13
    Caption = 'Label10'
  end
  object Label11: TLabel
    Left = 32
    Top = 80
    Width = 38
    Height = 13
    Caption = 'Label11'
  end
  object Label12: TLabel
    Left = 32
    Top = 160
    Width = 38
    Height = 13
    Caption = 'Label12'
  end
  object Edit11: TEdit
    Left = 32
    Top = 40
    Width = 433
    Height = 21
    TabOrder = 0
    Text = 'Edit11'
    OnClick = O5
  end
  object Edit13: TEdit
    Left = 24
    Top = 208
    Width = 441
    Height = 21
    TabOrder = 1
    Text = 'Edit13'
    OnClick = S4
  end
  object Edit12: TEdit
    Left = 32
    Top = 112
    Width = 433
    Height = 21
    TabOrder = 2
    Text = 'Edit12'
    OnClick = O6
```

```
    end
    object Button4: TButton
      Left = 32
      Top = 264
      Width = 425
      Height = 57
      Caption = 'Button4'
      TabOrder = 3
      OnClick = Button4Click
    end
  end
  object TabSheet5: TTabSheet
    Caption = 'TabSheet5'
    ImageIndex = 4
    object Memo1: TMemo
      Left = 24
      Top = 8
      Width = 393
      Height = 305
      Lines.Strings = (
        'Memo1')
      TabOrder = 0
    end
  end
  object TabSheet7: TTabSheet
    Caption = 'TabSheet7'
    ImageIndex = 6
    object Button5: TButton
      Left = 16
      Top = 24
      Width = 449
      Height = 145
      Caption = 'Button5'
      TabOrder = 0
      OnClick = Button5Click
    end
    object Edit4: TEdit
      Left = 272
      Top = 168
      Width = 121
      Height = 21
      TabOrder = 1
    end
    object Button6: TButton
      Left = 16
      Top = 184
      Width = 449
      Height = 121
      Caption = 'Çàðåãèñòðèðîâàòü'
      TabOrder = 2
      OnClick = Button6Click
    end
  end
end
object OpenDialog1: TOpenDialog
  Left = 4
  Top = 304
end
object SaveDialog1: TSaveDialog
  Left = 52
  Top = 304
end
object OpenDialog2: TOpenDialog
  Left = 100
```

```
    Top = 304
  end
  object SaveDialog2: TSaveDialog
    Left = 152
    Top = 304
  end
  object OpenDialog3: TOpenDialog
    Left = 196
    Top = 304
  end
  object OpenDialog4: TOpenDialog
    Left = 252
    Top = 304
  end
  object SaveDialog3: TSaveDialog
    Left = 308
    Top = 304
  end
  object OpenDialog5: TOpenDialog
    Left = 364
    Top = 304
  end
  object OpenDialog6: TOpenDialog
    Left = 408
    Top = 304
  end
  object SaveDialog4: TSaveDialog
    Left = 452
    Top = 304
  end
end
```