НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ
«ФАХОВИЙ КОЛЕДЖ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проєкту

Фаховий молодший бакалавр
(освітньо-професійний ступінь)

на тему: *Розробка програмних засобів для криптографічного захисту даних на базі матричного шифру перестановок*

Виконав студент IV курсу, групи ***ОК-42***
ОПП *«Обслуговування комп'ютерних систем та мереж»*
Спеціальності *123 Комп'ютерна інженерія*
Головенко Андрій Михайлович
(прізвище, ім'я по батькові)

Керівник		*Любомира Кужій*
	(підпис)	(ім'я прізвище)
Нормоконтролер		*Любомира Кужій*
	(підпис)	(ім'я прізвище)
Рецензент		
	(підпис)	(ім'я прізвище)
Голова ЕК		*Олег Гіщак*
	(підпис)	(ім'я прізвище)
Члени ЕК		*Любомира Кужій*
	(підпис)	(ім'я прізвище)
		Андрій Селемонавічус
	(підпис)	(ім'я прізвище)

Дипломний проєкт захищений в ЕК « __ » _____2025 р.

з оцінкою « _____ »

Львів 2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ
«ФАХОВИЙ КОЛЕДЖ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Циклова комісія	***Комп'ютерних систем і мереж***
Освітньо-професійний ступінь	***Фаховий молодший бакалавр***
Освітньо-професійна програма	***Обслуговування комп'ютерних систем та мереж***
Спеціальність	***123 Комп'ютерна інженерія***

ЗАТВЕРДЖУЮ
Завідувач відділення
«Комп'ютерних систем і мереж»
_____ Володимир СТАХІВ
«____» _____ 2025 року

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЄКТ СТУДЕНТУ

Головенку Андрію Михайловичу
(прізвище, ім'я та по батькові)

1. Тема проєкту *Розробка програмних засобів для криптографічного захисту даних на базі матричного шифру перестановок*

керівник проєкту *Кужій Любомира Іванівна, к.т.н.*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом директора від «*20*» *березня 2025* року № *20 - ст*

2. Строк подання студентом проєкту «*10*» *червня* 2025 року

3. Вихідні дані до проєкту

3.1 Програмне середовище мови програмування С

3.2 Матричний шифр перестановок для шифрування та розшифрування інформації

3.3 Відкритий текст для криптографічного перетворення на основі матричного шифру перестановок

4. Зміст розрахунково-пояснювальної записки

4.1 Основні аспекти захисту інформації

4.2 Криптографічна система захисту даних на основі матричного шифру перестановок

4.3 Розробка алгоритмів і програмного забезпечення для шифрування та дешифрування текстів на основі матричного шифру перестановок

4.4 Техніко-економічне обґрунтування

4.5 Охорона праці та безпека життєдіяльності

5. Перелік графічного матеріалу

5.1.	Схематичне представлення процесу шифрування і дешифрування інформації
5.2.	Криптографічна система шифрування повідомлення на основі матричного шифру перекстановок
5.3.	Структурна схема алгоритму шифрування текстів на основі матричного шифру перестановок
5.4.	Витрати на розробку та впровадження проєктного рішення

6 Консультанти розділів проєкту

Розділ	Ім'я, прізвище та посада консультанта	Підпис, дата	
		Завдання видав	Завдання отримав
Техніко-економічне обґрунтовування	*Тетяна Підкуймуха*		
Охорона праці та безпека життєдіяльності	*Роман Томків*		

7. Дата видачі завдання « *01*»*квітня 2025* року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту	Термін виконання	Примітка
1	*Основні аспекти захисту інформації*	20.04.2025	
2	*Використання матричного шифру перестановок для криптографічного захисту даних*	24.04.2025	
3	*Розробка блок-схем алгоритмів шифрування та дешифрування текстів на основі матричного шифру перестановок*	1.05.2025	
4	*Розробка програмного забезпечення для захисту даних за допомогою матричного шифру перестановок*	7.05.2025	
5	*Техніко-економічне обґрунтування*	10.05.2025	
6	*Охорона праці та безпека життєдіяльності*	20.05.2025	
7	*Вступ, реферат, висновки, зміст*	25.05.2025	
8	*Розробка демонстраційних креслень*	01.06.2025	

Студент _____ *Андрій Головенко*
(підпис) (ім'я, прізвище)

Керівник проєкту _____ *Любомира Кужій*
(підпис) (ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка дипломного проєкту: 81 сторінка., 21 рисунок, 5 таблиць, 25 джерел, 2 додатки.

Об’єкт дослідження – захист інформації при її передачі по каналах зв’язку.

Мета проєкту – розробка програмного забезпечення для захисту інформації на основі матричного шифру.

Метод дослідження – алгоритмічно-програмний.

У дипломному проєкті проаналізовано криптографічні алгоритми з використанням матричного шифру, спрямовані на перетворення змісту початкових повідомлень в незрозумілу форму. Розглянуто загальні вимоги до шифрування і дешифрування текстів, програмно реалізовані алгоритми для захисту інформації.

Розглянуто методи побудови алгоритмів для шифрування і дешифрування інформації на основі матричних шифрів.

Проведено шифрування комп’ютерних текстів, заданих у вигляді масиву символів, їх розшифрування і одержання початкового тексту.

Розраховано параметри для забезпечення сприятливих умов роботи на комп’ютерах з точки зору норм та правил охорони праці при розробці програмного забезпечення

КРИПТОГРАФІЯ, КРИТОГРАФІЧНІ МЕТОДИ, ЗАХИСТ ІНФОРМАЦІЇ, ШИФРУВАННЯ, ШИФРОТЕКСТ, ДЕШИФРУВАННЯ, МАТРИЧНИЙ ШИФР, КРИПТОАНАЛІЗ, СИМЕТРИЧНА КРИПТОГРАФІЯ, КЛЮЧ ШИФРУВАННЯ

ЗМІСТ

ВСТУП

Передача конфіденційних даних в мережах зв'язку може привести до втрати переданої інформації та її компрометації, тобто розголошення секретності. Інформація може стати відомою сторонній особі, яка не має права доступу до неї. На даний час існує велика кількість класичних методів захисту, які надійно захищають дані від втручання сторонніх осіб, і можуть бути застосовані для захисту інформації від зловмисників.

Особливе значення в автоматизованих системах приділяється питанням захисту інформації. Актуальність проблеми захисту інформації полягає в тому, що в сучасних умовах забезпечення інформаційної безпеки різних систем ефективно розв'язується за допомогою криптографічних алгоритмів. Це зумовлює посилення роботи криптоаналітиків по реалізації атак на одержання доступу до секретної інформації. Тому необхідно підвищити надійність і безпеку використовуваних методів захисту даних.

Існує багато криптографічних алгоритмів та протоколів, що використовуються для захисту інформації, але більшість із них орієнтовані на послідовну обробку скалярних даних. Водночас в мережі Інтернет та інших локальних та глобальних мережах і системах зв'язку поширені двовимірні масиви та зображення. Тому доцільною є розробка криптографічних алгоритмів для захисту даних матричним методом, коли процес шифрування і дешифрування представляються у вигляді багатовимірних, наприклад, матричних масивів.

В дипломному проєкті розглянуто спосіб шифруванні відкритих текстів з використанням матричних шифрів. Розроблено алгоритм і програмне забезпечення для шифрування і дешифрування даних. Продемонстровано процес написання програм на мові С для захисту інформації на основі матричного шифру., наведено структурні схеми алгоритмів і описано розроблені програми.

Метою дипломного проєкту є розроблення програмного забезпечення для криптографічного захисту даних, що базується на використанні матричного

шифру. Перевагою програмного способу захисту інформації є висока продуктивність, захищеність та практичність при використанні.

Для досягнення поставленої мети необхідно розв’язати наступні задачі:

– Провести огляд криптографічних алгоритмів для захисту даних;

– Дослідити симетричні алгоритми шифрування та дешифрування відкритих текстів;

– Розробити схему алгоритму для шифрування відкритих текстів з використанням матричного шифру;

– Розробити програмне забезпечення для шифрування відкритих текстів з використанням матричного шифру;

– Розробити схему алгоритму для дешифрування шифротекстів, зашифрованих матричним шифром;

– Розробити програмне забезпечення для дешифрування шифротекстів, зашифрованих матричним шифром.

Дипломний проєкт складається з вступу, п’ятьох розділів, висновків, списку використаних джерел та додатків.

У вступі сформульовано актуальність проблеми захисту даних, визначено мету проєкту та завдання, показано практичну цінність одержаних результатів.

У першому розділі описані основні напрями та аспекти проблеми захисту інформації. Проведено аналіз законів та нормативних актів, які регламентують правове забезпечення по організації заходів щодо захисту даних. Зроблено огляд сучасних криптографічних алгоритмів захисту інформації, наведено класифікацію криптографічних методів шифрування і дешифрування даних за різними критеріями.

Другий розділ присвячено аналізу криптографічної системи захисту даних на основі матричного шифру перестановок.

У третьому розділі описані розроблені алгоритми та програмне забезпечення для перетворення відкритих текстів в шифротексти та дешифрування зашифрованих текстів на базі матричного шифру перестановок.

Проаналізовані одержані результати та наведені блок-схеми розроблених алгоритмів.

У четвертому розділі розраховано витрати на розробку та впровадження проєктного рішення, зроблнно кошторис цих витрат, обчислено економічний ефект від використання розробленого програмного продукту.

Розрахунок показників сприятливих умов виконання практичної частини дипломного проєкту проведено в п’ятому розділі. Ці показники розраховано згідно чинного законодавства та норм і правил з охорони праці.

У висновках описані одержані результати криптографічного захисту даних на основі матричного шифру перестановок та показано їх практичну цінність.

У додатках наведені тексти розроблених програмних модулів на мові програмування С для шифрування та дешифрування текстів на базі матричного шифру перестановок.

1 ОСНОВНІ АСПЕКТИ ЗАХИСТУ ІНФОРМАЦІЇ

1.1 Актуальність проблеми інформаційної безпеки

Актуальність вивчення аспектів інформаційної безпеки пов’язана із входженням України в глобальні процеси, в яких постійно зростає значення інформації та її перетворення на найцінніший товар і продукт. В інформаційному суспільстві інформаційний вплив на суспільство та громадянина є надзвичайно важливим. Значення інформації зростає в міру зникнення національних кордонів між державами. Але суспільство повинно також турбувати проблема інформаційного перенасичення, недостовірної та шкідливої інформації, загрози національній безпеці держави через інформаційну агресію іноземних держав. [12]

Інформаційна безпека – це захист життєво важливих інтересів суспільства, держави і особи від заподіяння шкоди через негативні наслідки функціонування інформаційних технологій або внаслідок розповсюдження інформації, забороненої для розповсюдження законами України. Основними характеристиками інформаційної безпеки є [20]:

– Балансування на стику національної безпеки та інформаційної функції держави;

– Не замикання на національних кордонах;

– Протистояння між бажанням держави засекретити як найбільший масив інформації і невід’ємним правом громадянина мати вільний доступ до неї;

– Державне регулювання інформаційної сфери на правовій основі.

Інформаційна безпека має важливе значення для функціонування суспільства. Невід’ємною частиною загальнолюдських прав є інформаційні стосунки між особою, державою та суспільством [4].

Програмні засоби призначені для перетворення відкритих текстів до незрозумілого вигляду, шляхом розробки відповідного програмного забезпечення. Зміна вигляду відкритого тексту для захования його змісту називається шифруванням. Відкритим текстом може бути текстовий файл або

бітове зображення. Зашифроване повідомлення називається шифротекстом. Операція перетворення зашифровано тексту у початковий називається дешифруванням або розшифруванням**.** Шифруванням і дешифруванням текстів займається криптографія Розшифруванням шифротекстів називається криптоаналізом. Галузь, що охоплює криптографію і криптоаналіз, називається криптологією. а люди, які нею займаються, називаються криптологами.

1.2 Правовий захист інформації в комп'ютерних системах

Широке впровадження інформаційних технологій у сфері державної діяльності, економіки, фінансів зумовило підвищення вимог до забезпечення безпеки інформації. Особливо гостро це питання виникло з появою комп'ютерної техніки та автоматизованих систем опрацювання інформації. Карний Кодекс України здійснює правову охорону інформації в автоматизованих систем, а саме:

– умисне втручання в роботу автоматизованих систем, що приводить до перекручення чи знищення інформації;

– поширення програмних і технічних засобів, призначених для незаконного проникнення в автоматизовані системи, здатних до перекручення або знищення інформації [7].

Комп'ютерні системи здійснюють автоматизовану обробку даних. До її складу входять технічні засоби обробки, а також методи, процедури та програмне забезпечення. Захист інформації в автоматизованих системах − це сукупність організаційно-технічних заходів і правових норм для запобігання шкоди інтересам власників даних. Право власності на інформацію встановлюється з урахуванням норм авторського права на підставі угоди між власником вхідної інформації і користувачем автоматизованих систем. Користувач може обробляти інформацію лише за згодою власника.

Власник автоматизованих систем повинен забезпечити захист інформації згідно угоди з власником інформації. Захист інформації здійснюється шляхом:

– дотримання суб’єктами правових відносин, норм, вимог та правил організаційно-технічного характеру щодо захисту інформації;

– перевірки відповідності засобів автоматизованих систем встановленим вимогам захисту інформації.

Інформація, що є власністю держави, повинна оброблятися в автоматизованих систем, що має відповідний сертифікат. Закон встановлює відповідальність за порушення порядку і правил захисту інформації, механізм відшкодування нанесеної шкоди, гарантує забезпечення інформаційних прав.

Особливість регулювання інформаційних відносин в Інтернеті визначається особливістю фізичного представлення інформації в мережі в електронному вигляді. При передачі інформації відсутній носій інформації, що ускладнює оформлення і представлення документованої інформації у віртуальному середовищі. Виникає проблема закріплення і захисту правового режиму електронного документа, який би гарантував достовірність. Тому актуальною є проблема електронного підпису. Основними об’єктами, з приводу яких виникають інформаційні відносини в Інтернеті, є:

– програмно-технічні комплекси, інформаційні системи;

– інформація, інформаційні ресурси;

– інформаційні права та свободи;

– інтереси особистості, суспільства та держави;

– інформаційний суверенітет держави;

– інформаційна безпека.

З поширенням електронних документів виникають такі проблеми:

– визначення поняття “електронний документ“;

– підтвердження юридичної сили електронного документа;

– встановлення факту і дати введення документа в мережу;

– ідентифікація змісту електронного документа з його власником;

– доведення права авторства електронного документа.

З позицій інформаційної безпеки Інтернет може використовуватися зі злочинною метою. Тому інформаційна безпека в Інтернеті спрямована на захист:

– національної безпеки;

– людської гідності, репутації, прав неповнолітніх;

– інформації (несанкціонований доступ);

– таємниці особистого життя;

– інтелектуальної власності (незаконне поширення творів, програмного забезпечення, музики тощо).

1.3 Засоби захисту інформації

Засоби захисту інформації діляться на технічні та програмні. Вся сукупність технічних засобів ділиться на фізичні й апаратні. Фізичні засоби реалізуються у виді автономних пристроїв і систем і виконують функції загального захисту об'єктів, на яких опрацьовується інформація. До них відносяться пристрої захисту територій і будинків, де розміщена апаратура, грати на вікнах, електронно-механічне устаткування охоронної сигналізації. До апаратних технічних засобів відносяться пристрої, що вбудовуються в обчислювальну техніку та телекомунікаційну апаратуру.

Програмні засоби являють собою програмне забезпечення, призначене для виконання функцій захисту інформації. За допомогою програмних засобів дані перетворюються в незрозумілу форму для її передачі по каналах зв’язку.

Організаційні засоби захисту передбачають організаційно-технічні й організаційно-правові заходи, здійснювані в процесі створення й експлуатації апаратури телекомунікацій для забезпечення захисту інформації. Вони охоплюють усі структурні елементи на всіх етапах їх життєвого циклу (будівництво помешкань, проектування системи, монтаж і наладка устаткування).

Морально-етичні засоби захисту реалізуються у вигляді норм, що склалися традиційно в даній країні. Ці норми зазвичай не є обов'язковими, як законодавчі міри, але їх недотримання веде до втрати авторитету і престижу співробітника.

Законодавчі засоби захисту визначаються законодавчими актами країни, які регламентують правила використання, опрацювання і передачі інформації

обмеженого доступу і встановлюють міри відповідальності за порушення цих правил. На рис. 1.1 наведена класифікація засобів захисту інформації.

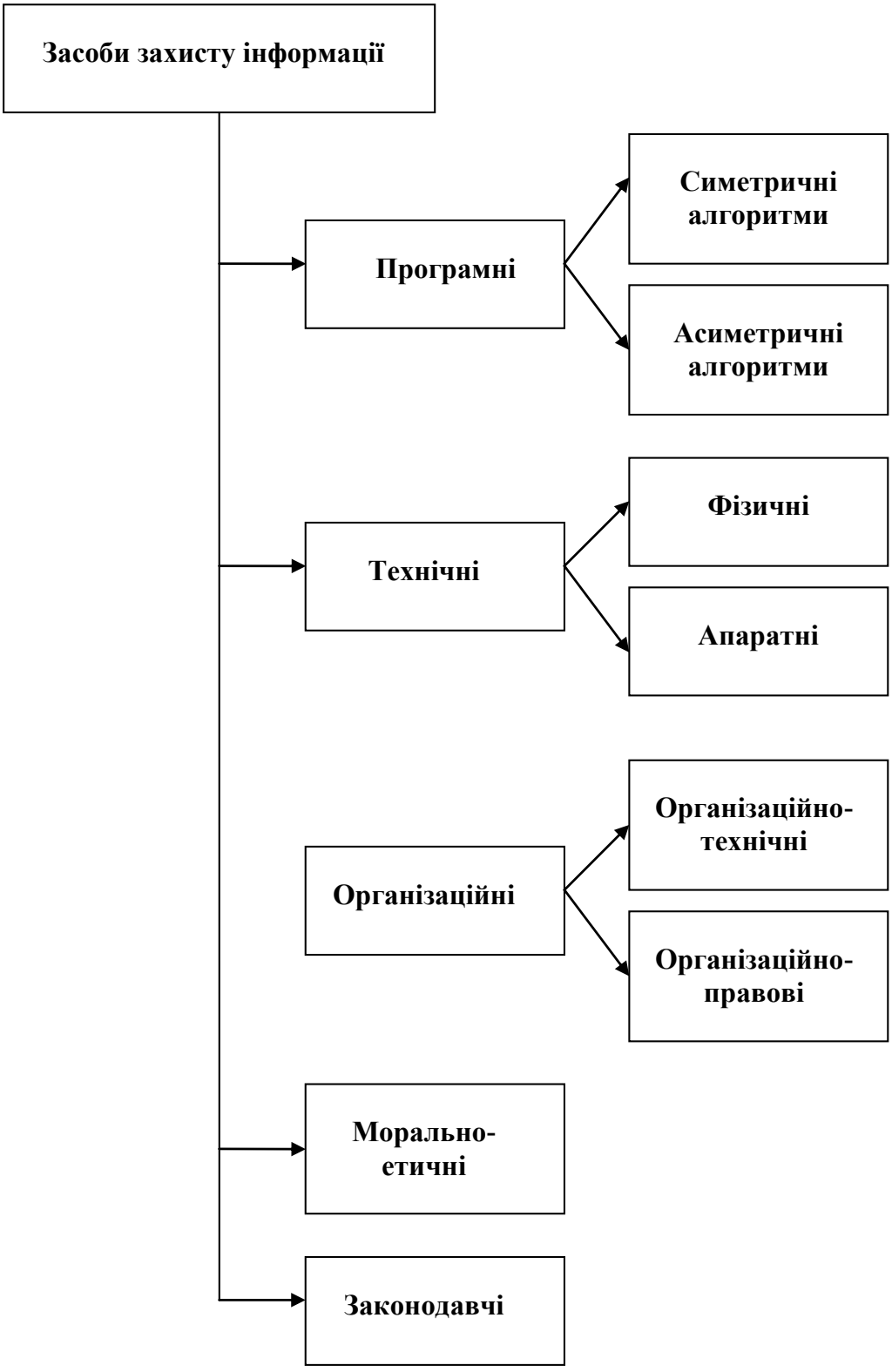


Рисунок 1.1 – Класифікація засобів захисту інформації

Необхідно також відзначити, що всі розглянуті засоби захисту діляться на формальні, що виконують захисні функції строго по заздалегідь передбаченій процедурі без особистої участі людини, і неформальні, обумовлені цілеспрямованою діяльністю людини або регламентуючої цієї діяльності.

Технічний захист інформації − це сукупність організаційних структур, поєднаних цілями захисту інформації, нормативно-правової та матеріально-технічної бази і спрямована на забезпечення інженерно-технічними засобами конфіденційності, цілісності та доступності інформації. [4].

Технічний захист інформації спрямований на забезпечення інженерно-технічними засобами порядку доступу до інформації, яка становить державну та іншу таємницю. Витік інформації, яка становить державну або конфіденційну таємницю є загрозою національній безпеці України в інформаційній сфері.

Загрози інформаційній безпеці зумовлені наступними факторами:

– неефективністю державної політики в галузі інформаційних технологій;

– діяльністю іноземних держав;

– діяльністю політичних партій та окремих осіб у політичній боротьбі та конкуренції;

– злочинною діяльністю, спрямованою на протизаконне отримання інформації;

Напрямки державної політики у сфері технічного захисту інформації:

– нормативно-правове забезпечення;

– розробка нормативних актів захисту важливої відкритої інформації;

– організація забезпечення технічного захисту інформації;

– контроль за імпортом технологій технічного захисту інформації;

– підготовка кадрів у галузі технічного захисту інформації;

– розвиток міжнародної співпраці у сфері технічного захисту інформації.

1.4 Криптографічний захист інформації

Основу забезпечення інформаційної безпеки в інформаційно-телекомунікаційних системах складають криптографічні методи і засоби захисту інформації. Історично криптографія використовується з метою збереження секретних даних. Основними задачами, які вирішує криптографія, є забезпечення конфіденційності, цілісності, достовірності, юридичної значущості та оперативності доступу до інформації.

Криптографія вивчає методи, прийоми і системи шифрування та дешифрування текстових повідомлень. Криптографічні проблеми вимагають глибоких знань з історії, лінгвістики, філології, інформатики, психології та математики. Криптографічні методи мають давню історію, їх застосовували в Давньому Єгипті, Давній Греції та Римі, Київській Русі. Криптографія знаходила застосування в зовнішній політиці, військовій справі, релігійних підпільних рухах, кримінальному та антиурядовому середовищі. Наприклад, кілька систем ручного шифрування винайшли декабристи: метод транспозиції (переміщення літер і слів у тексті, що мало наслідком зміну їх порядку при написанні), метод заміни літер тексту літерами іншого алфавіту [24].

Навіть сама писемність була свого роду шифруванням . У стародавньому Китаї тільки вищі шари суспільства могли навчатися читанню і листуванню, а перший досвід застосування криптографії в Єгипті відноситься до 1900 року до н. е. Автор напису користувався незвичайними ієрогліфами. Є і інші приклади: дощечки з Месопотамії, на яких зашифрована формула виготовлення керамічної глазурі (1500 рік до н. э.), єврейський шифр ATBASH (500-600 роки до н. е.), грецький «небесний лист» (486 рік до н. е.) і шифр простої підстановки Юлія Цезаря (50-60 рік до н. е.). Кама Сутра Ватс’яяни навіть ставить мистецтво тайнопису на 44-е, а мистецтво секретної розмови на 45-е місце в списку 64 мистецтв, якими повинні володіти чоловіки і жінки [13].

Одним із ключових критеріїв при розробці програмного забезпечення, що використовує криптографію, є застосування алгоритму шифрування. В даний час

існує достатня кількість алгоритмів для реалізації методів шифрування.

Конфіденційність – це властивість інформації бути доступною тільки обмеженій групі осіб. Під цілісністю розуміється властивість інформації зберігати свою структуру і зміст в процесі зберігання і передачі. Достовірність інформації полягає в строгій приналежності об'єкту, який є її джерелом. Здатність інформації бути доступною для кінцевого користувача відповідно до його часових вимог забезпечується оперативністю. Юридична значущість означає, що документ володіє юридичною силою.

В основі криптографічних методів лежить поняття криптографічного перетворення інформації на основі певних математичних законів з метою виключити доступ до неї сторонніх користувачів. Криптографічне перетворення називається алгоритмом шифрування. Процес шифрування однозначно відображає множину відкритих повідомлень в множину криптограм.

Шифри повинні володіли наступними властивостями:

– Законний одержувач зможе виконати зворотне перетворення і однозначно розшифрувати текст, знаючи криптографічний алгоритм;

– Криптоаналітик (зловмисник), що перехопив повідомлення, не зможе відновити по ньому початкове повідомлення без часових витрат і засобів, які зроблять інформацію непридатною.

Для коректної передачі секретної інформації по каналах зв'язку з використанням криптографічного алгоритму сторони інформаційного обміну повинні дотримуватись певної послідовності дій, що називається криптографічним протоколом. В основі криптографічного протоколу лежить шифр. Криптографічні протоколи є важливою складовою частиною криптографічної системи. Через наявність слабих місць в протоколі можливі ситуації, коли завдання забезпечення безпеки інформації не розв'язуються.

Кожна дія криптопротоколу є або обчисленнями, що виконуються діючими суб'єктами протоколу, або розсилкою повідомлень між ними. Атаки на протоколи з боку противника можуть бути направлені як проти криптографічних алгоритмів, використовуваних в протоколах, так і проти самих протоколів.

При пасивній атаці противник обмежується спостереженням за діями сторін протоколу і намагається витягнути із спостережень корисну для себе інформацію, не втручаючись в реалізацію протоколу. При активній атаці на криптографічний протокол противник видозмінює протокол в своїх інтересах. Це може привести до введення в протокол нових повідомлень, підміни одних повідомлень іншими, видалення з протоколу реальних даних, виводу з ладу каналу зв'язку або пам'яті, в якій зберігається інформація.

Основними завданнями забезпечення інформаційної безпеки за допомогою криптографічних протоколів є:

– Обмін алгоритмами з подальшим захистом обміну даними;

– Аутентифікація сторін, що встановлюють зв'язок;

– Авторизація користувачів при доступі до телекомунікаційних і інформаційних служб.

1.5 Класифікація алгоритмів захисту інформації

Шифрування даних є одним з важливих рішень проблеми криптографічного захисту. Зашифровані дані стають доступними тільки для того, хто знає, як їх розшифрувати, і тому викрадення зашифрованих даних пов’язане з великими труднощами для несанкціонованих користувачів. Шифри використовувались задовго до появи комп’ютерної техніки. При шифруванні використовуються алгоритми і ключі. Алгоритм дозволяє використати порівняно короткий ключ для шифрування наскільки завгодно великого тексту [13, 24].

Криптографічний захист – це захист даних з допомогою криптографічного перетворення, під яким розуміється перетворення даних шифруванням. Шифруванням даних називається процес перетворення відкритих даних на зашифровані з допомогою шифру, а розшифруванням даних – процес перетворення закритих даних на відкриті з допомогою шифру. Ключ – це секретний параметр алгоритму криптографічного перетворення даних, що забезпечує вибір одного варіанту із множини для даного алгоритму.

Крипостійкість шифру визначається його стійкістю до дешифрування. Звичайно ця характеристика визначається періодом часу, необхідним для дешифрування. Важливим критерієм при розробці програмного забезпечення для криптографічного захисту інформації є вибір алгоритму шифрування. В даний час існує велика кількість алгоритмів для реалізації методів шифрування.

У криптографія з ключем алгоритми шифрування переданих даних можуть бути відомі усім стороннім особам, але вони ще залежить від деякого параметра – "ключа", яким володіють лише відправник і одержувач повідомлення. В залежності від кількості ключів, які застосовуються в алгоритмі, криптографічні алгоритми поділяються на 3 категорії:

– Безключові алгоритми, що не використовують ключів при шифруванні;

– З одним ключем, що використовують для шифрування та дешифрування один ключ;

– З двома ключами, один для шифрування інформації, – інший для дешифрування. Зазвичай один ключ є секретним, а інший відкритим;

На рисунку 1.2 наведена класифікація криптографічних алгоритмів за кількістю ключів.

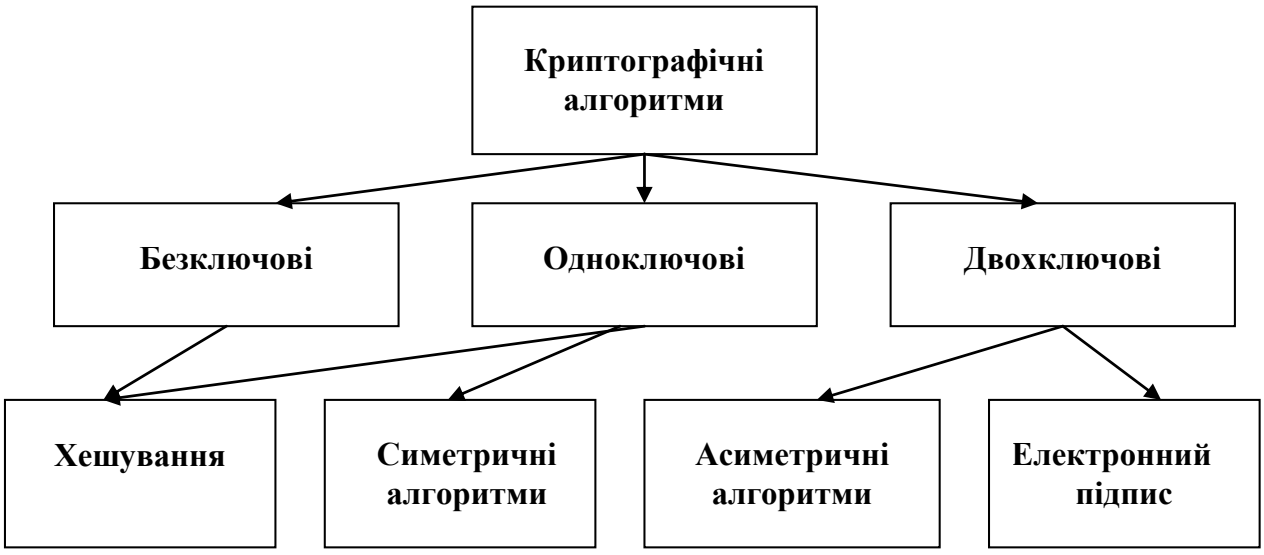


Рисунок 1.2 – Класифікація криптографічних алгоритмів за кількістю ключів

В залежності від використання кількості ключів та їх типів,

криптогалгоритми поділяються на симетричні та асиметричні. В симетричних алгоритмах для шифрування та дешифрування повідомлень використовується один і той ключ. В асиметричних алгоритмах для шифрування повідомлення використовується один відкритий ключ, що відомий усім бажаючим, а для дешифрування – другий закритий, який існує тільки в одержувача зашифрованого шифру. На рисунку 13. наведена класифікація криптосистем за кількістю та типом використання ключів.

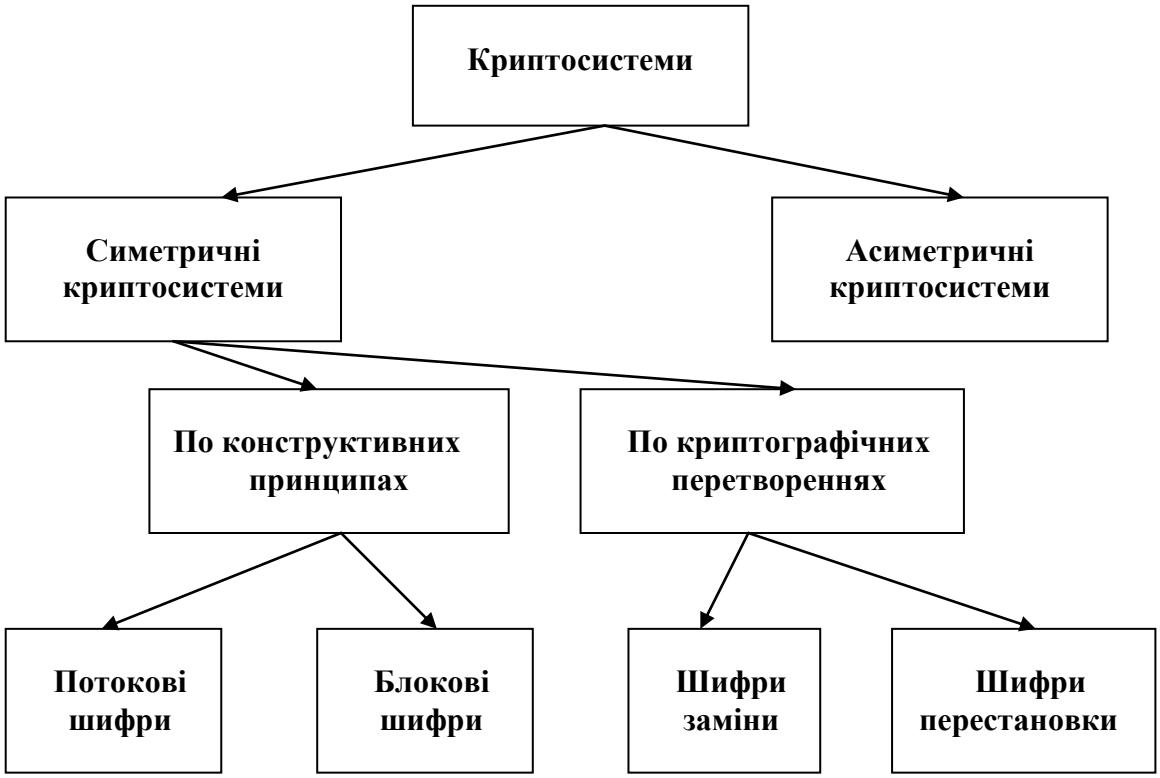


Рисунок 1.3 – Класифікація криптосистем за кількістю та типом ключів

Системи, в котрих для шифрування і дешифрування повідомлень використовується однаковий ключ, називаються симетричними. Симетричні системи використовують переважно для перетворення тексту, який є комбінацією перестановок і замін. Симетричні алгоритми по конструктивному принципу діляться на потокові та блочні шифри.

В потокових шифрах **о**диницею кодування є один біт. Результат кодування не залежить від попереднього вхідного потоку. Потокова схема застосовується в системах передачі потоків інформації, тобто в тих випадках, коли передача

інформації починається і закінчується в довільні моменти часу. Поширеними представниками потокових шифрів є скремблери.

В блочних шифрах одиницею кодування є блок з декількох байтів. Результат кодування залежить від усіх байтів цього блоку. Схема застосовується при пакетній передачі інформації та кодуванні файлів. Блочні шифри оперують з блоками відкритого тексту, на які накладаються наступні вимоги:

– Достатня криптостійкість;
– Простота процедур шифрування і дешифрування;
– Висока надійність.

В залежності від криптографічних перетворень, що здійснюються над даними, симетричні алгоритми діляться на шифри заміни та перестановки.

При використанні шифру заміни кожний елемент початкового тексту взаємно-однозначно замінюється одним, або декількома знаками деякого алфавіту. Шифр простої заміни замінює кожний знак вхідного алфавіту на деякий знак з того ж алфавіту, Результат заміни не залежить від розташування знаку у відкритому тексті. Ключами для шифрів заміни є таблиці. В алгоритмах заміни змінюється порядок блоків інформації за законами криптосистеми. Переважна більшість сучасних алгоритмів належить цій групі.

В перестановочних алгоритмах блоки інформації (байти, біти, або інші одиниці) не змінюються, але змінюється їх порядок проходження, що робить шифровану інформацію недоступною сторонній особі.

Шифри перестановки відрізняються від шифрів заміни тим, що при шифруванні буква відкритого тексту переходить не у фіксований знак алфавіту, а в іншу букву того ж відкритого тексту, внаслідок чого букви розташовуються на нових місцях, тобто переставляються. Ключем для даного шифру також служить таблиця заміни, тільки не букв алфавіту, а їх індексів в тексті, який підлягає шифруванню. Розмір таблиці заміни дорівнює довжині відкритого тексту.

Симетричні алгоритми характеризуються можливістю швидкого шифрування великих потоків інформації в каналах зв’язку. Вони забезпечують

високу ступінь секретності. Симетричні алгоритми дають змогу використовувати одні і ті ж апаратні засоби для шифрування і дешифрування інформації [13, 22].

Схематично процес шифрування і дешифрування інформації показано на рис. 1.4.

Рисунок 1.4 – Схематичне представлення процесу шифрування і дешифрування інформації

На передавальній стороні виконується шифрування відкритого повідомлення за допомогою алгоритму шифрування з використанням ключів. В результаті отримуємо криптограму, яка передається відкритим каналом зв’язку. На приймальній стороні до отриманого зашифрованого повідомлення розшифровується. Розшифрування буде вірним, якщо криптограма не була змінена під час передачі по каналу зв’язку.

1.6 Симетричні та асиметричні алгоритми шифрування

Класичні методи шифрування використовують симетричний функціонал. До них відносяться шифри перестановки, простої й складної заміни, а також їхні модифікації та комбінації. Широко на практиці застосовується комбінації шифрів перестановки й заміни. Класична криптографія використовувала для шифрування й дешифрування інформації один ключ. Його передача здійснювалась надійним каналом обміну [13, 22, 24].

Симетричний метод перестановки ґрунтується на криптографічному перетворюванні, яке містить правило переставляння символів у відкритому повідомленні. Шифри перестановки мають иалу криптостійкість, тому їх не використовують без додаткових перетворень.

Шифри підстановоу діляються на одноалфавітні та багатоалфавітні. В одноалфавітнтх підстановках символи початкового тексту замінюються на інші символи того ж алфавіту. При одноалфавітній підстановці кожний символ початкового тексту замінюється на символ шифрованого тексту за однаковим алгоритмом. Багатоалфавітна підстановка змінює алгоритм перетворювання від символу до символу, тобто один і той символ вхідного тексту може шифруватися різними символами в шифротексті.

Шифрування з закритим ключем ґрунтується на тому, що доступ до ключа має тільки авторизований персонал. Цей ключ повинен триматися в секреті. Якщо ключ потрапить до сторонніх рук, то можна отримати несанкціонований доступ до зашифрованої інформації. Недолік алгоритмів з закритим ключем полягає в тому, що для відправки одержувачу захищеного повідомлення необхідно володіти безпечним способом передачі закритого ключа .

Прикладом алгоритму з закритим ключем є стандарт Data Encryption Standard (DES). Цей алгоритм, розроблений компанією IBM в 70-их роках минулого століття. Він прийнятий в якості американського стандарту для комерційних і несекретних урядових комунікацій.

Інші відомі системи шифрування із закритим ключем — це RC2, RC4, RC5, потрійний DES і IDEA. Потрійний DES-алгоритм забезпечує достатній ступінь

захисту. Він використовує метод шифрування DES, але застосовує його тричі, використовуючи три різні ключі.

На відміну від симетричних алгоритмів з закритими ключами широко використовуються асиметричні алгоритми або алгоритми з відкритими ключами, В асиметричних алгоритми використовується два ключі: один відкритий – для шифрування початкових текстів, другий секретний ключ – для розшифрування шифротекстів. Відкритий ключ можна вільно поширювати по незахищених каналах зв'язку. За допомогою відкритого ключа можна зашифрувати будь-яку інформацію і вільно відправити її адресату. Адресат в свою чергу зможе розшифрувати її виключно секретним ключем [7, 22].

Криптографічні алгоритми з відкритими ключами дуже популярні. Перед кожним сеансом генеруються секретні ключі, після чого відбувається передача інформації. Основною перевагою алгоритмів з відкритими ключами є відсутність необхідності передавати іншій особі секретний ключ.

Шифрування з відкритим ключем ґрунтується на тому, що для шифрування даних використовується один ключ, а для розшифрування інший. Дані, які зашифровані за допомогою відкритого ключа, можна розшифрувати виключно за допомогою відповідного закритого ключа, а цифровий підпис даних, підписаних за допомогою закритого ключа, можна перевірити тільки за допомогою відповідного відкритого ключа.

1.7 Основні вимоги до алгоритмів шифрування

Процес криптографічного захисту даних може здійснюватися як програмно, так і апаратно. Апаратна реалізація відрізняється істотно більшою вартістю. Але їй притаманні і висока продуктивність, простота, захищеність і т.д. Програмна реалізація є практичнішою, допускає гнучкість у використанні [24]. Для сучасних криптографічних систем захисту інформації ставляться наступні вимоги [5]:

– зашифроване повідомлення повинно піддаватися читанню тільки при наявності ключа;

– число операцій для визначення ключа шифрування за фрагментом шифрованого повідомлення і відповідного йому відкритого тексту, має бути не менше загального числа можливих ключів;

– число операцій, необхідних для дешифрування інформації шляхом перебору різноманітних ключів, повинно виходити за межі можливостей сучасних комп'ютерів;

– знання алгоритму шифрування не впливають на надійність захисту;

– незначна зміна ключа повинна приводити до істотної зміни виду зашифрованого повідомлення;

– незначна зміна вихідного тексту повинно приводити до істотної зміни виду зашифрованого повідомлення навіть при використанні одного і того ж ключа;

– структурні елементи алгоритму шифрування повинні бути незмінними;

– додаткові дані, що вводяться в повідомлення в процесі шифрування, повинен бути повністю та надійно сховані в зашифрованому тексті;

– не повинно бути простих і легко встановлюваних залежностей між ключами, що послідовно використовуються в процесі шифрування;

– будь-який ключ із множини можливих повинен забезпечувати надійний захист інформації.

Процес обміну інформацією здійснюється таким способом:

– одержувач обчислює відкритий і секретний ключі, секретний ключ зберігає в таємниці;

– відправник, відкритим ключем одержувача, зашифровує сеансовий ключ, який пересилається одержувачу по незахищеному каналу;

– одержувач отримує сеансовий ключ і розшифровує його, використовуючи свій секретний ключ.

2 КРИПТОГРАФІЧНА СИСТЕМА ЗАХИСТУ ДАНИХ НА ОСНОВІ МАТРИЧНОГО ШИФРУ ПЕРЕСТАНОВОК

2.1 Класифікація шифрів перестановок

Основна ідея шифру перестановки полягає в перестановці символів в початковому тексті так, щоб без знання правил цієї перестановки, неможливо прочитати шифротекст. Є різні способи організації шифру перестановок:

– Числова перестановка;
– Матрична перестановка;
– Блочна перестановка шифрування.

Будь-який спосіб задання ключа можна представити в вигляді перестановки. Тому кількість можливих ключів для тексту довжини n рівно $n!$.

При шифрі перестановок початковий текст розбивається на блоки довжиною k, де k – довжина ключа. Після цього букви в кожному блоці переставляються згідно перестановці символів ключа. Якщо в останньому блоці не вистачає букв, то додаються випадкові, щоб довжина останнього блоку була рівна довжині ключа перестановки.

В шифрах перестановок змінюється порядок блоків інформації, що робить шифровану інформацію недоступною сторонній особі. Шифри діляться на:

– Шифр частоколу;
– Матричний шифр перестановок;
– Шифр Першої світової війни (ADFGVX-шифр).

Одним із найпростіших шифрів перестановки є шифр частоколу. Він дуже схожий на матричний шифр. Він характеризується висотою частоколу. Наприклад, шифрування слова “студент” шифром із висотою частоколу 2. Для цього запишемо його так:

т	д	н	
с	у	е	т

Далі зчитуємо спочатку верхній рядок «тдн», а потім нижній «сует». В результаті одержимо зашифроване повідомлення «тднсует». Для частоколу висотою 3 отримаємо такий зашифрований текст:

у	н	
т	е	
с	д	т

Тепер зчитуємо спочатку верхній рядок «ун», потім – другий «те», а потім нижній “сдт”. В результаті одержимо зашифроване повідомлення **«**унтесдт**».**

2.2. Опис матричного шифру перестановок

Існує багато криптографічних алгоритмів та протоколів, що використовуються для захисту інформації [23], але більшість із них орієнтовані на послідовну обробку скалярних даних. Але в локальних та глобальних мережах, системах зв'язку та телекомунікацій поширені при передачі двовимірні масиви та зображення. Так як при послідовній обробці та шифруванні скалярних даних використовуються одні і ті ключі, то послідовні алгоритми є не дуже стійкими до криптоаналізу. Тому актуальною є проблема модифікації відомих алгоритмів та протоколів для криптошифрувань на матричні, коли дані, що шифруються і дешифруються, представляються у вигляді багатовимірних, наприклад, матричних масивів.

Суть матричного шифру полягає в наступному. Відкритий текст записують послідовно рядок за рядком у квадратну матрицю. Літери шифротексту виписуються із цієї матриці по стовпчиках. Зашифруємо текст «матричний шифр перестановок**»** на прикладі матриці розміром 5×5.

м	**а**	**т**	**р**	**и**
ч	**н**	**и**	**й**	**ш**
и	**ф**	**р**	**п**	**е**
р	**е**	**с**	**т**	**а**
н	**о**	**в**	**о**	**к**

Зчитуючи літери по стовпчиках, отримаємо шифротекст: «м ч и р н а н ф е о т и р с в р й п т о и ш е а к». Криптостійкість такого шифру незначна, але її можна підсилити з допомогою використання ключових слів.

Зашифруємо цей текст на ключових словах «слово» та «група». Ключове слово «слово» записуємо згори від матриці тексту, а ключове слово «група»– зліва від матриці тексту. Далі переставляємо рядки згідно з позицією кожної літери слова «група» у алфавіті (а, г, п, р, у). Матриця з відкритим текстом «матричний шифр перестановок» перетвориться в матрицю, показану на рис. 2.1.

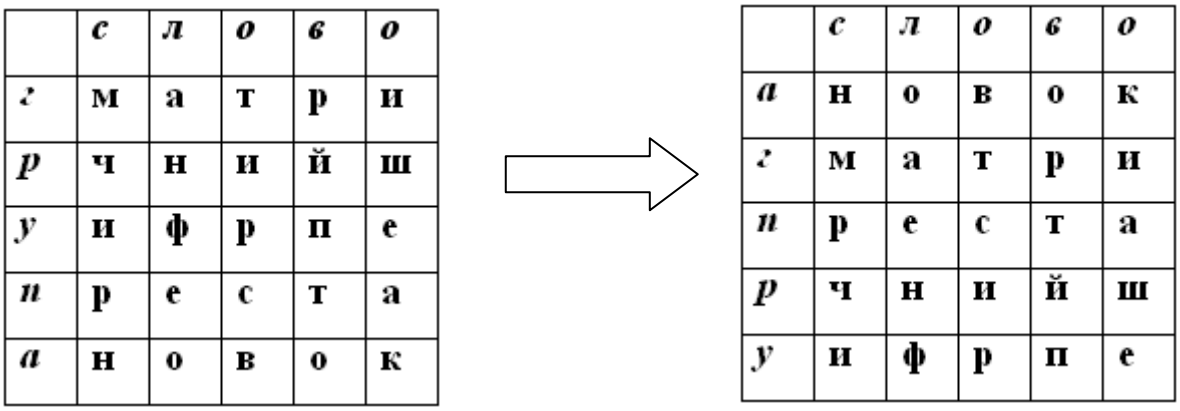
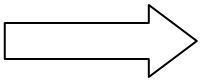


Рисунок 2. 1 – Переставляння рядків матриці згідно з позицією літер ключового слова у алфавіті

Далі переставляємо стовпчики згідно з позицією кожної літери слово «слово» у алвавіті (в, л, о, о, с). Матриця на рис. 2.1 (справа) перетвориться в матрицю, показану на рис. 2.2.

	с	л	о	в	о
а	н	о	в	о	к
г	м	а	т	р	и
п	р	е	с	т	а
р	ч	н	и	й	ш
у	и	ф	р	п	е



	в	л	о	о	с
а	о	о	в	к	н
г	р	а	т	и	м
п	т	е	с	а	р
р	й	н	и	ш	ч
у	п	ф	р	е	и

Рисунок 2.2 – Переставляння стовпчиків матриці згідно з позицією літер ключового слова у алфавіті

В результаті проведених перестановок рядків і стовпчиків одержано матрицю, показану на рис. 2.2. Прочитавши інформацію з матриці по стовпцях, отримаємо шифротекст:

«оовкнратимтесарйнишчпфреи».

Розшифрувати шифтекст можна, записавши слова в матрицю по стовпчиках згідно з порядком розташування літер ключів по абетці, а потім, переставивши стовпчики та рядки так, щоби ключі утворили зв'язані слова, зчитуємо розшифрований текст по рядках [22].

Модифікацією матричного шифру є шифр Першої світової війни ADFGVX. Цей шифр широко використовувався під час Першої світової війни. Шифр є комбінацією підстановки та перестановки за ключовим словом.

Даний шифр використовує матрицю 7×7. В першому стовпчику та першому рядку матриці записуються послідовно літери A, D, F, G, V, X. В матрицю вписуються всі латинські літери і арабські цифри від 0 до 9.

Для зашифрування тексту знаходимо його літери в таблиці. Далі вибираємо літери, які стоять у першому лівому стовпчику та верхньому рядку. Відповідно, вибрана літера замінюється парою літер, що стоять у першому лівому стовпчику та верхньому рядку.

Для дешифрування шифротекстів пари літер замінюємо літерою, яка стоїть на перетині відповідного рядка та стовпця побудованої матриці. Аналогічно можна створити схожі шифри для української мови та інших мов.

2.3 Опис алгоритму блочної перестановки шифрування та дешифрування

При використанні алгоритму блочної перестановки відкритий текст поділяється на блоки. Довжина блоків рівна довжині ключового слова або фрази. Якщо останній блок відкритого тексту коротший від ключового слова, то при необхідності він доповнюється довільними символами до розміру ключа.

Далі записуються номери букв в ключовому слові по зростанню їх появи в алфавіті. Послідовність номерів записуються під кожним блоком. Потім кожна літера блоку записується в порядку номерів літер ключа. Шифрування полягає в записуванні символів в блок на нові позиції. Відповідно, переставляються символи в кожному блоці, на які розбивається початкове повідомлення.

Для шифрування використовується алфавіт з українських букв і символу пропуску. В ролі ключа вибрано слово «диплом». Записуємо номери букв в ключовому слові по зростанню їх появи в алфавіті.

д	**и**	**п**	**л**	**о**	**м**
1	2	6	3	5	4

Текст для шифрування відкритого повідомлення «**матричний шифр перестановок»** розіб’ємо на блоки по 6 символів, так як довжина ключового слова рівна шести. Останній блок доповнюємо символами «а» (три символи). Пронумеруємо символи в блоках. Під кожним блоком підпишемо нумерацію символів ключа. Процес шифрування блочною перестановкою показано на рис. 2.3.

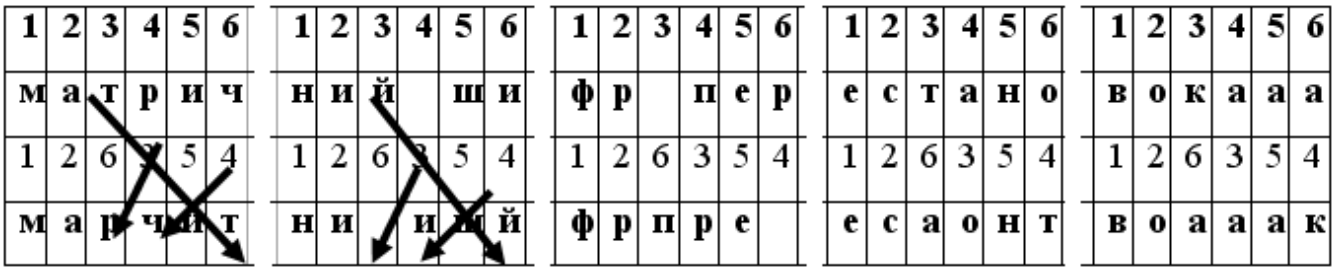


Рисунок 2.3 – Процес шифрування блочною перестановкою

Для початкового тексту «матричний шифр перестановок» шифротекст матиме вигляд:

«марчитни ишйфрпре есаонтвоааак»

Порядок шифрування складається з наступних кроків:

– Відкритий текст доповнюється будь-якими символами, так щоб його довжина стала кратною довжині ключового слова або фрази;

– Символи використовуваного ключа нумеруються згідно порядку їх появи в алфавіті;

– Номери ключа використовуються для шифрування тексту.

Для дешифрування символи з блоку шифрограми виписуються згідно ключу “**диплом**”. На рис. 2.4 показано обернений процес дешифрування.

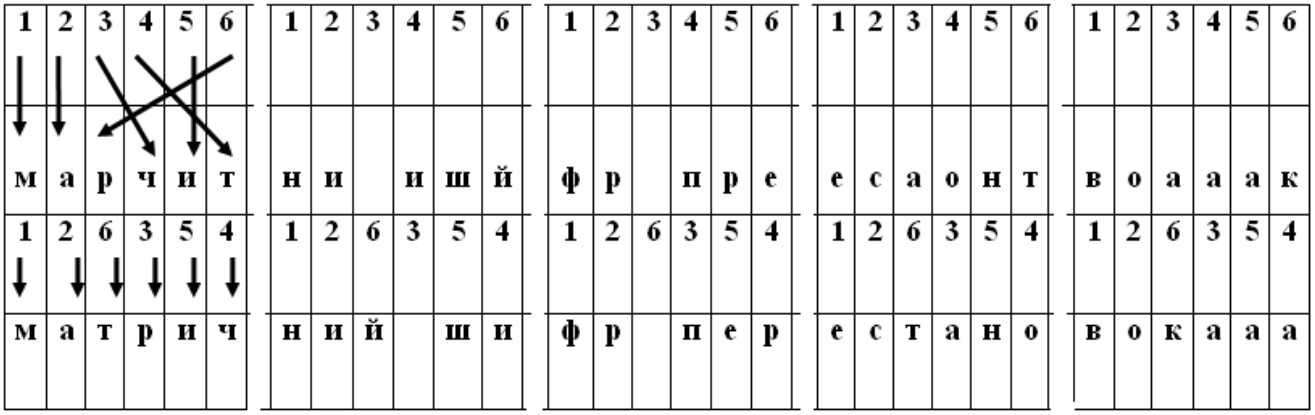


Рисунок 2.4 – Процес дешифрування блочною перестановкою

2.4 Багатокроковий матричний афінний шифр

Існує багато криптографічних алгоритмів, що використовуються для послідовної обробки скалярних даних. Однак часто виникає потреба в системах зв'язку та телекомунікаціях передавати та опрацьовувати, двовимірні масиви та зображення. Тому доцільною розробляти криптографічні алгоритми на матричній основі, коли дані, що шифруються і дешифруються, представляються у вигляді багатовимірних, наприклад, лвомірних масивів. Багатокроковий матричний афінний шифр використовується для створення криптосистем з метою опрацювання двовимірних масивів та зображень.

Відомі результати моделювання модифікованого алгоритму створення двомірного ключа для криптосистем. В основу цього алгоритму покладено протокол та математичні моделі протоколу Діффі-Хеллмана. Ці результати розширили функціональні можливості та стійкість матричного шифрування. При цьому всі обчислювальні процедури виконуються над матрицями.

При використанні систем матричного шифрування двом користувачам, які встановлюють безпечний обмін інформацією, необхідно на основі модифікованого алгоритму встановити спільний секретний ключ. Цей ключ, представлений у вигляді матриці, зображення або сукупності зображень. Його можна використати для подальшого шифрування і дешифрування, оскільки обидві сторони отримують однаковий ключ. Модифікований матричний афінний багатокроковий шифр має порівняно з традиційним афінним асиметричним шифром переваги.

Матричний афінний багатокроковий шифр належать до асиметричних систем та алгоритмів. Для реалізації алгоритму вибирають два матричні ключі A та S у вигляді матриць заданої розмірності. Елементи ключових матриць вибираються з діапазону $[1,\ \ n]$, де $n-$ досить велике просте число. Ці ключі використовують для шифрування. Крім того, вводиться ще параметр B, який задає кількість кроків шифрування.

Для знаходження двох матричних ключів *AD* та *SD* для дешифрування, потрібно для кожного елемента a_{ij} матриці *A* знайти обернене за модулем *n* число ad_{ij} матриці *AD* та для елемента s_{ij} обчислити значення елемента sd_{ij} матриці *SD*.

Відповідно, процеси шифрування та дешифрування для матричного повідомлення *M* та криптограми *C* виражаються матричними формулами:

$$C = (M \otimes A + S) \bmod n, \tag{2.1}$$

$$M = (C \otimes AD + SD \bmod n), \tag{2.2}$$

де символ $\otimes$ – означає поелементне множення матриць,

символ $+$ – означає поелементне додавання матриць,

$K \bmod n$ – означає поелементне взяття за модулем n всіх елементів матриці K.

Для підвищення криптостійкості алгоритму операції множення та додавання матриць здійснюється p разів. Тоді процеси шифрування та дешифрування для матричного повідомлення *M* та криптограми *C* виражаються такими матричними формулами:

$$C = ((((M \otimes A + S) \otimes A + S) \otimes A + S) \quad \cdots \quad + S) \bmod n, \tag{2.3}$$

$$M = ((((C \otimes AD + SD) \otimes AD + SD) \otimes AD + SD) \quad \cdots \quad SD) \bmod n. \tag{2.4}$$

Ці формули легко перетворювати. Це дасть змогу спростити відповідні обчислювальні процедури та виконувати їх покроково.

3 РОЗРОБКА АЛГОРИТМІВ І ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ШИФРУВАННЯ ТА ДЕШИФРУВАННЯ ТЕКСТІВ НА ОСНОВІ МАТРИЧНОГО ШИФРУ ПЕРЕСТАНОВОК

3.1 Опис системи шифрування текстів на основі матричного шифру перестановок

При використанні матричного шифру перестановок для захисту даних відкритий текст представляється у вигляді прямокутної матриці. Метод шифрування полягає у послідовному записуванні символів відкритого тексту рядок за рядком у матрицю. Для одержання шифротексту символи виписуються з цієї матриці по стовпчиках.

Для підсилення крипостійкості матричного алгоритму при шифруванні відкритого повідомлення використовують ключі. В ролі ключа може бути ключове слово або фраза. Довжина ключа рівна кількості символів у рядку матриці. Якщо довжина повідомлення не кратна довжині ключа, то повідомлення доповнюється будь-якими символами. Процес шифрування здійснюється за допомогою перестановки символів в ключовій фразі.

В першому рядку матриці записуються літери ключового слова. Далі послідовно записуються символи відкритого тексту рядок за рядком у матрицю. Після чого стовпці матриці переставляємо згідно номера позиції кожної літери ключа в алфавіті.

Результат шифрування зчитуємо з матриці по стовпцях. Така перестановка суттєво ускладнює процес дешифрування. Але це не означає, що цей шифр не можна дешифрувати за допомогою аналізу частот появи різних літер та їх блоків у зашифрованому тексті [22].

Процес шифрування повідомлень на основі матричного шифру перестановок складається з наступних кроків:

– Читання з файлу ключового слова;
– Запис номерів літер ключазгідно номерів позиції їх в алфавіті;

– Читання з файлу відкритого тексту і запис його в рядки матриці блоками, довжина яких рівна довжині ключа;

– Стовпці матриці переставляються згідно номерів літер ключа.

Нехай треба зашифрувати повідомлення «КОМПЮТЕРНА МОДЕЛЬ» за допомогою ключового слова «КОМА». Довжина даного повідомлення має 17 символів і не кратна довжині ключа. Тому повідомлення доповнюємо до 20-ти символів, наприклад, трьома символами ‘А’. Далі записуємо номери літер в ключовому слові по зростанню їх появи в алфавіті. На рис. 3.1 показано систему шифрування тексту «КОМПЮТЕРНА МОДЕЛЬ» на основі матричного шифр

К	О	М	А
2	4	3	1
К	О	М	П
Ю	Т	Е	Р
Н	А		М
О	Д	Е	Л
Ь	А	А	А

А	К	М	О
1	2	3	4
П	К	М	О
Р	Ю	Е	Т
М	Н		А
Л	О	Е	Д
А	Ь	А	А

Рисунок 3.1 – Шифрування тексту «КОМПЮТЕРНА МОДЕЛЬ» на основі матричного шифру

Шифротекст зчитуємо з матриці по стовпцях. Тому після зашифрування повідомлення “КОМПЮТЕРНА МОДЕЛЬ”, отримаємо таку криптограму:

«ПРМЛАКЮНОЬМЕ ЕАОТАДА»

Нехай треба зашифрувати повідомлення

«ІНФОРМАЦІЙНА БЕЗПЕКА МАЄ ВАЖЛИВЕ ЗНАЧЕННЯ ДЛЯ ФУНКЦІОНУВАННЯ СУСПІЛЬСТВА»

за допомогою ключового слова “ДИПЛОМ”. На рис. 3.2 показано криптографічну систему шифрування повідомлення на основі матричного шифру.

Д	*И*	*П*	*Л*	*О*	*М*
1	*2*	*6*	*3*	*5*	*4*
І	Н	Ф	О	Р	М
А	Ц	І	Й	Н	А
	Б	Е	З	П	Е
К	А		М	А	Є
	В	А	Ж	Л	И
В	Е		З	Н	А
Ч	Е	Н	Н	Я	
Д	Л	Я		Ф	У
Н	К	Ц	І	О	Н
У	В	А	Н	Н	Я
	С	У	С	П	І
Л	Ь	С	Т	В	А

Д	*И*	*Л*	*М*	*О*	*П*
1	*2*	*3*	*4*	*5*	*6*
І	Н	О	М	Р	Ф
А	Ц	Й	А	Н	І
	Б	З	Е	П	Е
К	А	М	Є	А	
	В	Ж	И	Л	А
В	Е	З	А	Н	
Ч	Е	Н		Я	Н
Д	Л		У	Ф	Я
Н	К	І	Н	О	Ц
У	В	Н	Я	Н	А
	С	С	І	П	У
Л	Ь	Т	А	В	С

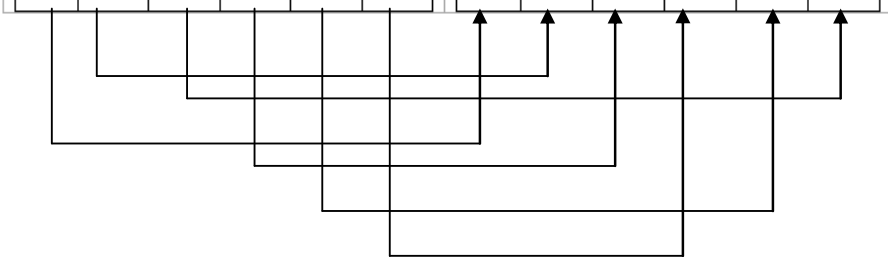


Рисунок 3.2 – Криптографічна система шифрування повідомлення на основі матричного шифру перестановок

Після шифрування повідомлення отримаємо наступну криптограму:

«ІА К ВЧДНУ ЛНЦБАВЕЕЛКВСЬОЙЗМЖЗН ІНСТМАЕЄИА УНЯІАРНПАЛНЯФОНПВФІЕ А НЯЦАУС»

3.2 Опис алгоритму для шифрування текстів на основі матричного шифру перестановок

Розробити алгоритм для шифрування заданого тексту з використанням матричного методу і реалізувати його у вигляді програмного забезпечення. Такі алгоритми є стійкими до впливу завад та різних спотворень, що виникають при передачі документів по каналах зв’язку. На рис. 3.3 зображено структурну схему алгоритму шифрування текстів на основі матричного шифру перестановок.

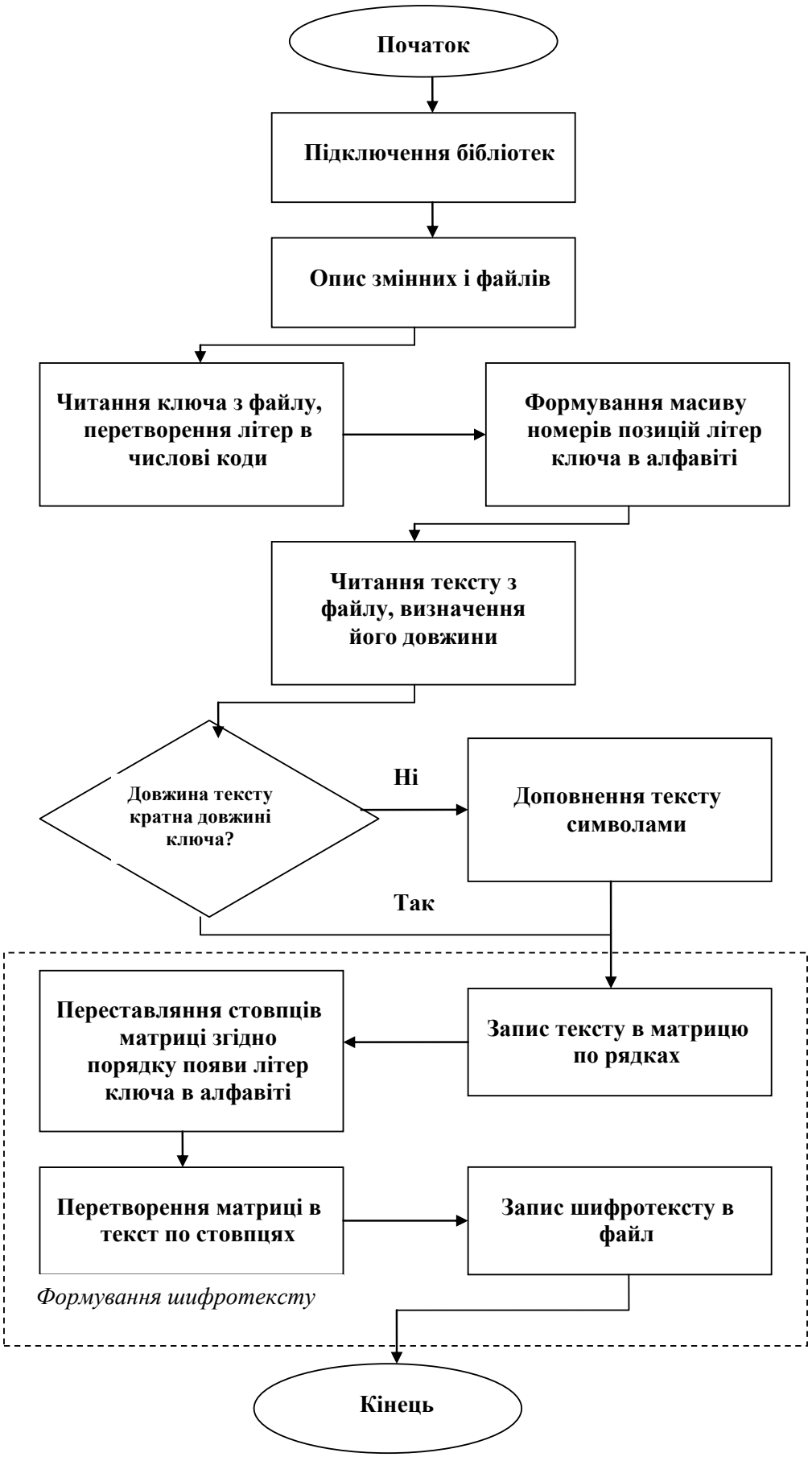


Рисунок 3.3 – Структурна схема алгоритму шифрування текстів на основі матричного шифру перестановок

При використанні матричного шифру літерам алфавіту присвоюються числові коди. В табл. 3.1 наведено відповідність між числовими кодами і літерами алфавіту, вибраного для шифрування тексту.

Таблиця 3.1 – Відповідність між числовими кодами і літерами

Код	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Літера	А	Б	В	Г	Д	Е	Є	Ж	З	И	І	Ї	Й	К	Л	М
Код	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Літера	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ю	Я

Алгоритм шифрування матричним шифром виконує наступні функції:

– Читання ключа шифрування з файлу file_key.txt;

– Кодування літер ключового слова на основі табл. 3.1;

– Читання тексту для шифрування з файлу file_input_text.txt;

– Шифрування тексту за допомогою матричного шифру з ключем, який задається відправником повідомлення і тримається в секреті;

– Запис зашифрованої інформації в формі матриці в файл file_matr.txt та вивід її на екран;

– Перетворення зашифрованої інформації з матриці в текст і запис його в файл file_shyf_text.txt.

3.3 Опис програми для шифрування текстів матричним шифром

На основі алгоритму розроблено програмне забезпечення. При розробці програмного забезпечення було використано принцип розділення його на окремі функціональні компоненти. В процесі роботи програми ці компоненти взаємодіють між собою. Таким підхід в більшості випадків використовується для розробки багатофункціонального програмного забезпечення.

Проблема економії часу при розробці великих проектів є одною з актуальних задач, з якою зустрічаються фахівці при шифруванні та

розшифруванні повідомлень. Мову програмування С часто використовують при програмній реалізації складних та великих за обсягом проектів.

Описана в дипломному проєкті програма демонструє основні прийоми роботи з шифруванням повідомлень на основі матричного шифру. Шифрування проводиться з ключем, який задається словом. Відкритий текст для шифрування розміщений в текстовому файлі file_key.txt в форматі символьного масиву.

Програма повинна виконувати наступні функції:

– Читати задані тексти з файлів.
– Читати ключ з файлу.
– Зашифровувати текст матричним шифром із заданим ключем.
– Записувати шифротексти в файли.
– Виводити на екран одержані шифротексти.
– Читати з файлу зашифровану інформацію.

Програма складається з таких елементів:

1 Підключення бібліотечних файлів, які містять прототипи стандартних функцій файлового вводу-виводу, функцій обробки символьної інформації:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

2 Опис вказівників на змінні структурного типу FILE, які асоціюють фізичні файли з потоками вводу-виводу:

```
FILE *fp1, *fp2, *fp3, *fp4;
```

Потік fp1 зв’язаний з файлом, що містить відкритий текст, fp2 – з файлом для зберігання шифротексту. Потік fp3 асоціюється з файлом для зберігання ключа, а потік fp4 зв’язаний з файлом для запису проміжних матриць.

3 Задання імен використовуваних файлів:

```
char filename130]=" file_input_text.txt";  /* fp1 */
char filename230]=" \file_shyf_text.txt";   /* fp2 */
char filename330]=" file_key.txt";         /* fp3 */
char filename430]=" file_matr.txt";        /* fp4 */
```

4 Відкриття файлу file_input_text.txt для читання відкритого тексту:

```
fp1=fopen(filename1,mode_r);
if (fp1!=NULL ) {printf("file %s open mode %s\n",filename1, mode_r); }
else  { printf("file %s not open mode  %s\n", filename1, mode_r);  exit(1);}
```

5 Відкриття файлу file_shyf_text.txt" для запису шифротексту:

```
fp2=fopen(filename2,mode_w);
if (fp2!=NULL) {printf("file %s open mode %s\n",filename2, mode_w); }
else {printf("file %s not open mode 3 %s\n",filename2,mode_w); exit(2);}
```

6 Відкриття файлу file_key.txt, для читання ключа:

```
fp3=fopen(filename3,mode_r);
if (fp3!=NULL ){printf("file %s open mode %s\n",filename3, mode_r);}
else { printf("file %s not open mode %s\n",filename3, mode_r);  exit(3);}
```

7 Відкриття файлу file_matr.txt для запису матриці тексту.

```
fp4=fopen(filename4,mode_w);
if (fp4!=NULL ){printf("file %s open mode %s\n",filename4, mode_w);}
else {printf("file %s not open mode  %s\n",filename4, mode_w);  exit(4);}
```

8 Задання літер алфавіту для написання тексту наведено в таблиці 3.2.

Таблиця 3.2 – Задання літер алфавіту для написання відкритого тексту

KOD0]='А';	KOD1]='Б';	KOD2]='В';	KOD[3]='Г';
KOD[4]='Д';	KOD[5]='Е';	KOD[6]='Є';	KOD[7]='Ж';
KOD[8]='З';	KOD[9]='И';	KOD[10]='І';	KOD[11]='Ї';
KOD[12]='Й';	KOD[13]='К';	KOD[14]='Л';	KOD[15]='М';
KOD[16]='Н';	KOD[17]='О';	KOD[18]='П';	KOD[19]='Р';
KOD[20]='С';	KOD[21]='Т';	KOD[22]='У';	KOD[23]='Ф';
KOD[24]='Х';	KOD[25]='Ц';	KOD[26]='Ч';	KOD[27]='Ш';
KOD[28]='Щ';	KOD[29]='Ь';	KOD[30]='Ю';	KOD[31]='Я';

9 Читання літер ключа та визначення його довжини:

```
i=0; char_keyi]=fgetc(fp3);
```

```
while( char_keyi]!=EOF)  { i++;  char_key[i]=fgetc(fp3);}
l_key=strlen(char_key )-1;  /* Довжина ключа */
```

10 Кодування літер ключового слова та формування динамічних масивів *key_MAS* та *key_pos* для зберігання числових кодів літер ключа та їх номерів в алфавіті. Оскільки розмірність цих масивів залежать від довжини ключа і наперед невідома, то оперативна пам'ять для них виділяється динамічно за допомогою стандартної функції calloc():

```
key_MAS=(int *)calloc(l_key, sizeof(int));
key_pos=(int *)calloc(l_key, sizeof(int));
for (i=0; i<l_key; i++)
{key_MAS[i]=kodyvannja (char_key[i]); }
```

11 Читання тексту повідомлення з файлу c:\k\file_input_text.txt в оперативну пам'ять в масив *text_mas* та визначення його довжини:

```
i=0;  text_mas[i]=fgetc(fp1);
while( text_mas[i]!=EOF)
 {  i++;  text_mas[i]=fgetc(fp1);}
l_text=strlen(text_mas)-5;  /* Довжина тексту */
```

12 Визначення кількості рядків *kr_m* матриці *text_matr* запис у неї прочитаного з файлу тексту:

```
if (l_text%l_key==0) {kr_m= l_text/l_key;}
else                 {kr_m= l_text/l_key+1;}
k=0;  for (i=0; i<kr_m; i++)
for (j=0; j<l_key; j++)
{text_matr[i][j]= text_mas[k];k++ ;}
```

13 Запис матриці в файл c:\k\file_matr.txt:

```
for (i=0; i<kr_m; i++)
{for (j=0; j<l_key; j++)
fprintf(fp4,"%c \t",text_matr[i][j]);
 fprintf(fp4,"\n");}
```

14 Шифрування тексту і запис шифроматриці в файл c:\k\file_matr.txt :

```
for (i=0; i<kr_m; i++)
for (j=0; j<l_key; j++)
text_matr_shyf[i][key_pos[j]-1]=text_matr[i][j];
for (i=0; i<kr_m; i++)
{for (j=0; j<l_key; j++)
fprintf(fp4,"%c\t",text_matr_shyf[i][j]);  fprintf(fp4,"\n");}
```

15 Перетворення шифроматриці в текст і запис його у файл c:\k\file_shyf_text.txt:

```
k=0; for (j=0; j<l_key; j++) for (i=0; i<kr_m; i++)
{text_mas_shyf[k]=text_matr_shyf[i][j];
fprintf(fp2,"%c",text_mas_shyf[k]);k++ ;}
```

16 Закриття файлів:

```
fclose(fp1); fclose(fp2);
fclose(fp3); fclose(fp4);
```

Присвоєння числового значення прочитаній літері тексту здійснює функція kodyvannja (), прототип якої має вигляд:

```
int kodyvannja(char cum);
```

Аргументом функції є прочитаний символ, код якого повертає функція.

Результатом роботи програми є створення файлу c:\k\file_shyf_text.txt зашифрованого повідомлення. Повні тексти програми і функції kodyvannja() наведено в додатку А.

3.4 Результати роботи програми шифрування текстів

Вхідними даними для програми є текст для шифрування, який знаходиться у файлі file_input_text.txt та ключ, який знаходиться у файлі file_key.txt. На рис. 3.4 показано текст для шифрування (файл file_input_text.txt).

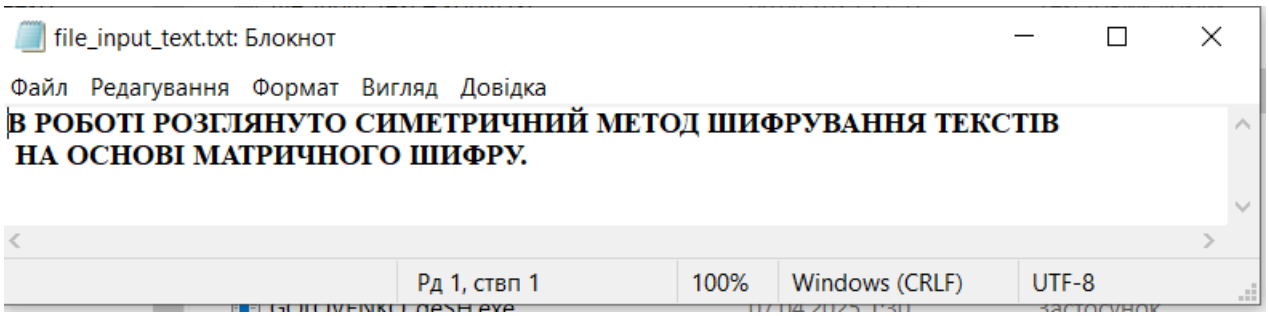


Рисунок 3.4 – Текст для шифрування (файл file_input_text.txt)

На рис. 3.5 показано файл file_key.txt ключа для шифрування.

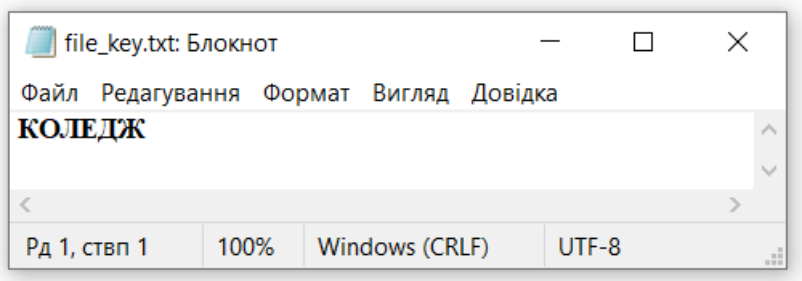


Рисунок 3.5 – Файл file_key.txt ключа для шифрування

Відкритий текст записується у вигляді матриці по рядках у файл file_matr.txt. На рис. 3.6 показано вигляд матриці відкритого тексту.

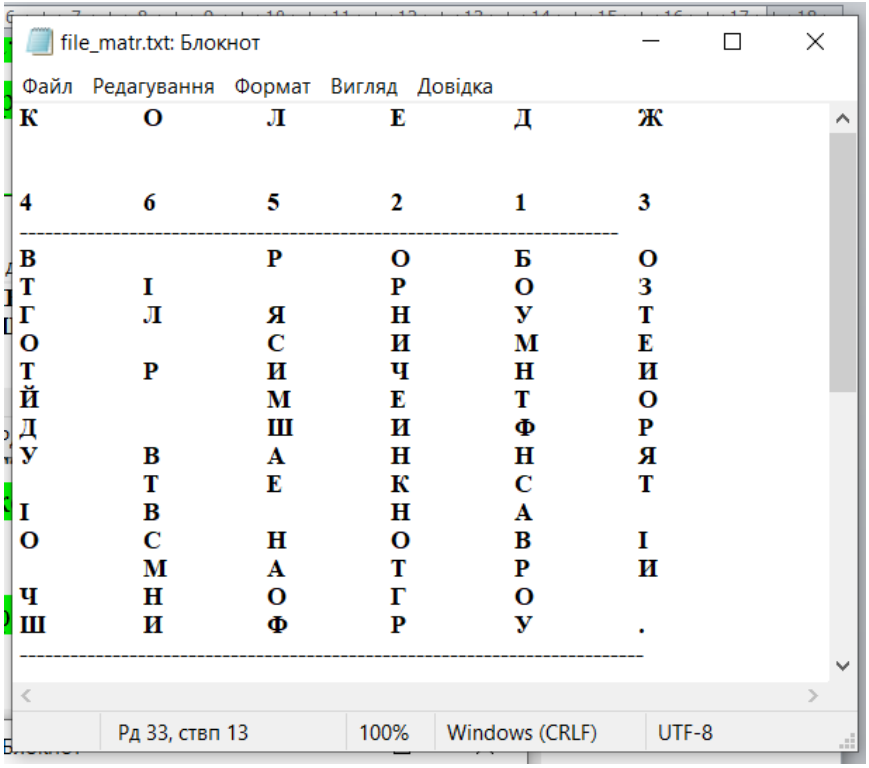


Рисунок 3.6 – Вигляд матриці відкритого тексту у файлі file_matr.txt

Відкритий текст шифрується за допомогою матричного шифру з ключем «КОЛЕДЖ» і записується у вигляді шифрованої матриці у файл file_deshyf_matr.txt. На рис. 3.7 показано вигляд шифрованої матриці.

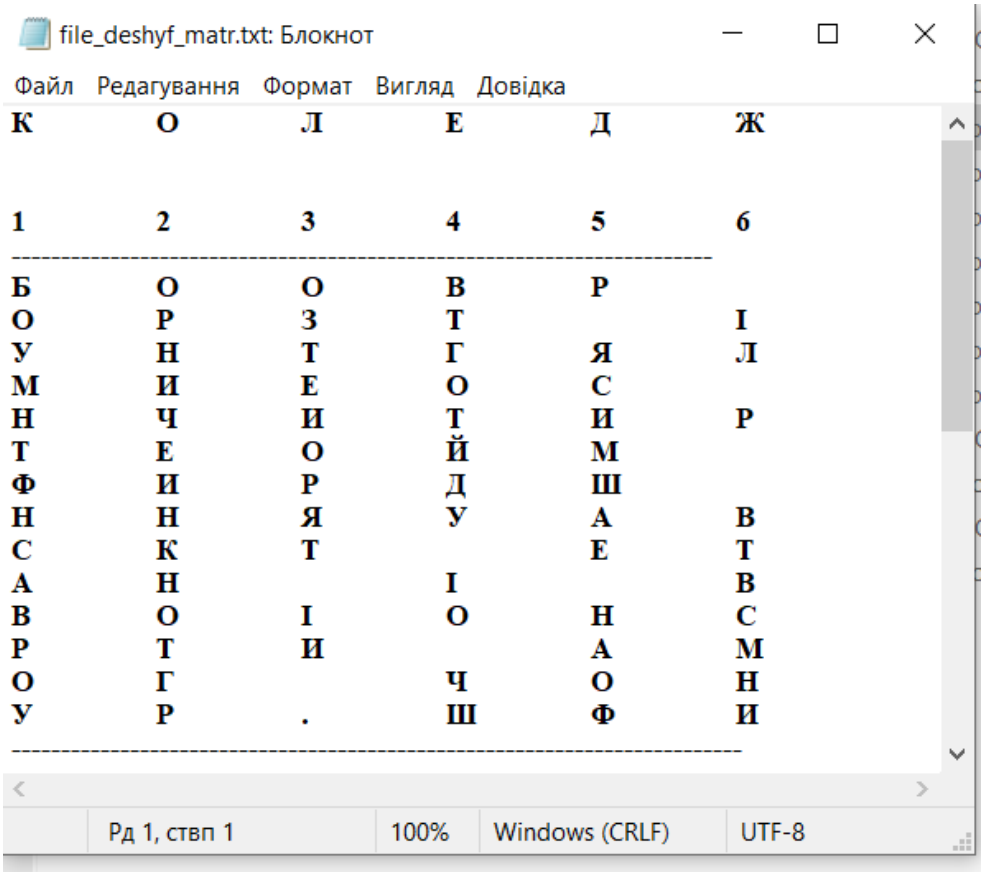


Рисунок 3.7 – Вигляд матриці шифротексту

Результатом роботи програми є зашифрований текст, записаний у файл file_shyf_text.txt по стовпцях. На рис. 3.8 показано файл зашифрованого тексту.

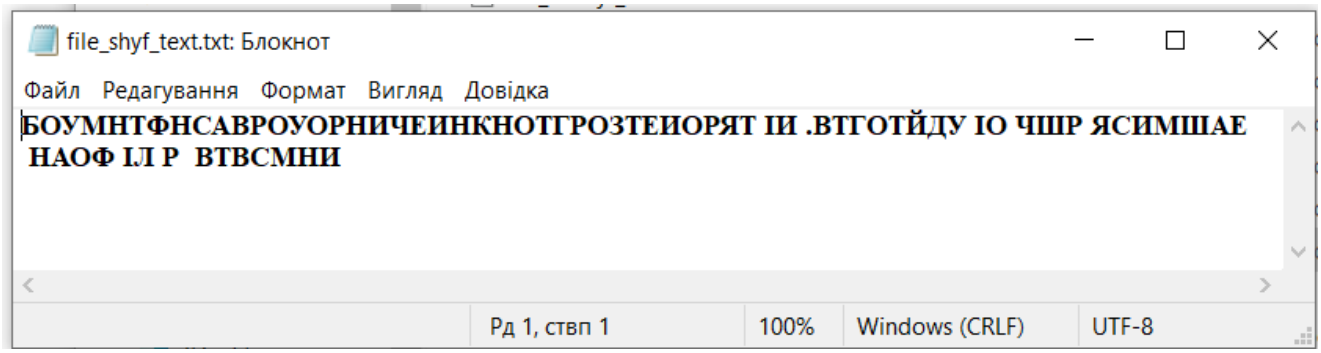


Рисунок 3.8 – Файл зашифрованого тексту

3.5 Опис алгоритму для дешифрування шифротекстів

На рис. 3.9 зображено структурну схему алгоритму дешифрування повідомлень на основі матричного шифру перестановок.

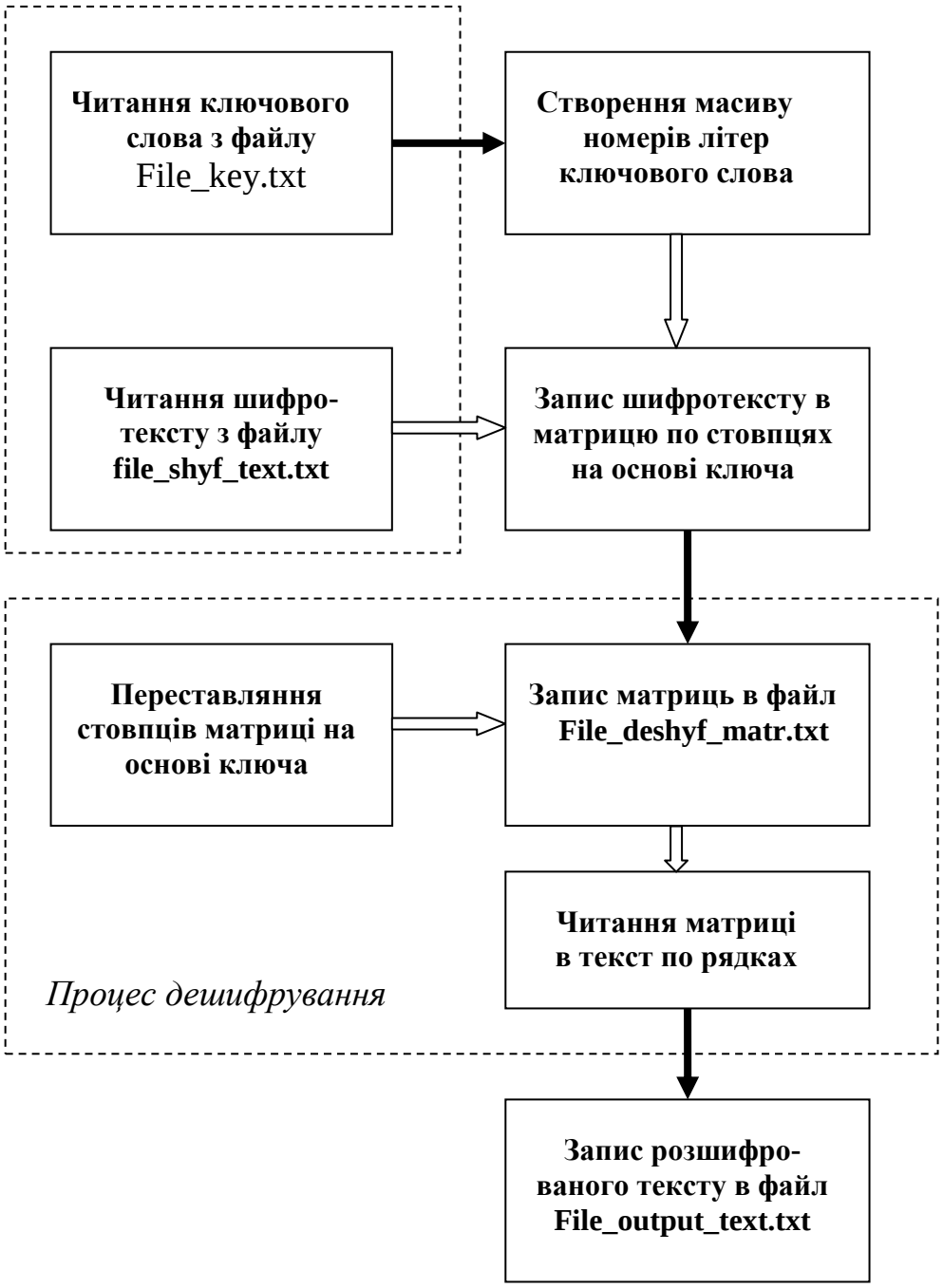


Рисунок 3.9 – Структурна схема алгоритму дешифрування текстів на основі матричного шифру перестановок

Алгоритм розшифрування текстів, зашифрованих за допомогою матричного шифру з використанням ключа, складається з таких кроків:

– Читання з файлу ключового слова;

– Запис номерів літер ключа згідно номерів позиції їх в алфавіті;

– Читання з файлу шифротексту і запис його в стовпці матриці блоками, довжина яких рівна результату від ділення довжини шифротексту на довжину ключа;

– Стовпці матриці переставляються згідно номерів літер ключа.

Одержувач повідомлення для розшифрування тексту повинен знати ключ. Записати розшифровану інформацію в новий файл. Розшифрування здійснюється за певними правилами підстановок і ключового слова. При цьому кожний символ шифротексту замінюється символом початкового тексту.

3.6 Опис програми для дешифрування шифротекстів

В дипломному проєкті розроблено програмне забезпечення для дешифрування зашифрованих повідомлень матричним шифром. Програма демонструє процес розшифрування повідомлень, зашифрованих матричним шифром з ключем. Програма перетворює зашифрований текст в початковий. Зашифрований текст розміщений в текстовому файлі file_shyf_text.txt в символьному форматі. Програма складається з наступних елементів.

1 Підключення бібліотечних файлів, які містять прототипи функцій файлового вводу-виводу, обробки символьної інформації та роботи системи:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

2 Опис вказівників на змінні структурного типу FILE, які асоціюють фізичні файли на диску з потоками вводу-виводу:

```
FILE *fp1,*fp2, *fp3, *fp4;
```

Потік fp1 зв’язаний з файлом, що містить шифротекст, fp2 – з файлом для зберігання відкритого тексту. Потік fp3 асоціюється з файлом для зберігання ключа, а потік fp4 зв’язаний з файлом для запису проміжних матриць.

3 Задання імен використовуваних файлів:

```
char filename1[30]=" file_shyf_text.txt";  /* fp1 */
char filename2[30]=" File_output_text.txt";/* fp2 */
char filename3[30]=" File_key.txt";        /* fp3 */
char filename4[30]=" File_deshyf_matr.txt";/* fp4 */
```

4 Відкриття файлу file_shyf_text.txt, який містить шифротекст:

```
fp1=fopen(filename1,mode_r);
if (fp1!=NULL) {printf("file %s open mode %s\n",filename1, mode_r);}
else { printf("file %s not open mode  %s\n", filename1, mode_r); exit(1);}
```

5 Відкриття файлу File_output_text.txt, для запису розшифрованого тексту:

```
fp2=fopen(filename2,mode_w);
if (fp2!=NULL ) {printf("file %s open mode %s\n",filename2, mode_w);}
else { printf("file %s not open mode %s\n", filename2, mode_w); exit(2);}
```

6 Відкриття файлу file_key.txt, для читання ключа:

```
fp3=fopen(filename3,mode_r);
if (fp3!=NULL ){printf("file %s open mode %s\n",filename3, mode_r);}
else  { printf("file %s not open mode %s\n",filename3, mode_r); exit(3);}
```

7 Відкриття файлу File_deshyf_matr.txt для запису шифроматриці:

```
fp4=fopen(filename4,mode_w);
if (fp4!=NULL ){printf("file %s open mode %s\n",filename4, mode_w);}
else { printf("file %s not open mode  %s\n",filename4, mode_w); exit(4);}
```

При відкритті всі файли перевіряються на правильність їх відкриття. При неможливості відкрити файли виводяться відповідні повідомлення і програма закінчує роботу, так як неможливо прочитати повідомлення для шифрування або записати зашифроване повідомлення.

8 Читання та кодування літер ключа, формування динамічних масивів key_MAS та key_pos для зберігання числових кодів літер ключа та їх номерів в алфавіті, визначення довжини ключа:

```
i=0; char_key[i]=fgetc(fp3);
while( char_key[i]!=EOF)  { i++;  char_key[i]=fgetc(fp3);}
```

```
l_key=strlen(char_key )-1;  /* Довжина ключа */
key_MAS=(int *)calloc(l_key, sizeof(int));
key_pos=(int *)calloc(l_key, sizeof(int));
for (i=0; i<l_key; i++)
{key_MAS[i]=kodyvannja (char_key[i]); }
```

9 Читання шифротексту повідомлення з файлу file_shyf_text.txt і запис його в матрицю shyf_matr[kr_m][l_key], де kr_m кількість рядків матриці shyf_matr та визначення його довжини:

```
i=0; shyf_mas[i]=fgetc(fp1);
while( shyf_mas[i]!=EOF)
 {  i++;  shyf_mas[i]=fgetc(fp1);}
l_text=strlen(shyf_mas)-1;  /* Довжина шифротексту */
```

10 Визначення кількості рядків *kr_m* матриці kr_m запис у неї прочитаного штфротексту, запис матриці в файл:

```
kr_m= l_text/l_key;
k=0; for (j=0; j<l_key; j++)
for (i=0; i<kr_m; i++)
{shyf_matr[i][j]= shyf_mas[k];k++ ;}
for (i=0; i<kr_m; i++)
{for (j=0; j<l_key; j++)
fprintf(fp4,"%c\t",shyf_matr[i][j]);  fprintf(fp4,"\n");}
```

11 Дешифрування шифротексту і запис матриці в файл File_deshyf_matr.txt:

```
for (i=0; i<kr_m; i++)
for (j=0; j<l_key; j++)
shyf_matr_text[i][j]=shyf_matr[i][key_pos[j]-1];
for (i=0; i<kr_m; i++)
{for (j=0; j<l_key; j++)
fprintf(fp4,"%c\t",shyf_matr_text[i][j]);  fprintf(fp4,"\n");}
```

12 Перетворення матриці в розшифрований текст і запис його у файл File_output_text.txt:

```
k=0; for (i=0; i<kr_m; i++)
for (j=0; j<l_key; j++)
{shyf_mas_text[k]=shyf_matr_text[i][j];
fprintf(fp2,"%c",shyf_mas_text[k]);k++ ;}
```

13 Закриття файлів:

```
fclose(fp1);
fclose(fp2);
fclose(fp3);
fclose(fp4);
```

Результатом роботи програми є створення файлу c:\k\File_output_text.txt, в якому знаходиться початковий текст. Повний текст програми наведено в додатку Б.

3.7 Результати роботи програми для дешифрування шифротекстів

Вхідними даними для програми є шифротекст, який знаходиться у файлі file_shyf_text.txt(рис. 3.8). Зашифрований текст записується у вигляді матриці по стовпцях у файл File_deshyf_matr.txt., показаний на рис. 3.7.

Шифротекст розшифровується за допомогою матричного шифру з ключем «КОЛЕДЖ» і записується у вигляді розшифрованої матриці у файл File_shyf_matr.txt. На рис. 3.10 показано вигляд розшифрованої матриці.

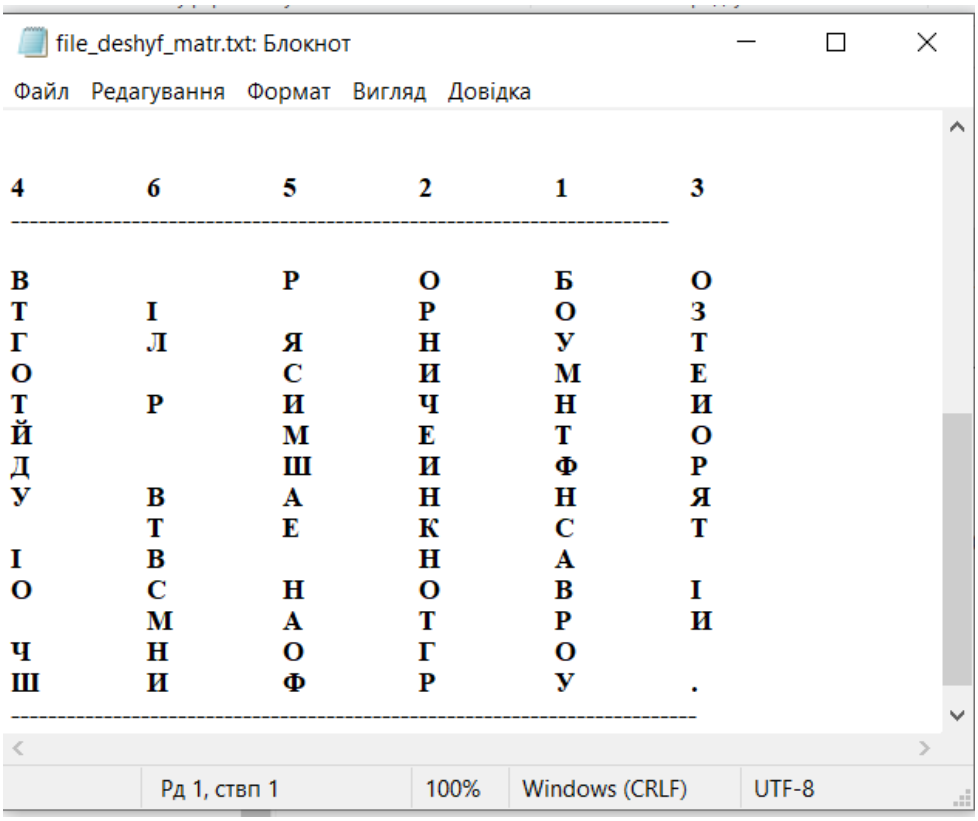


Рисунок 3.10 – Вигляд розшифрованої матриці

Зчитуючи текст матриці по рядках, одержимо початковий текст, який записується у файл File_output_text.txt. На рис. 3.11 показано файл розшифрованого тексту.

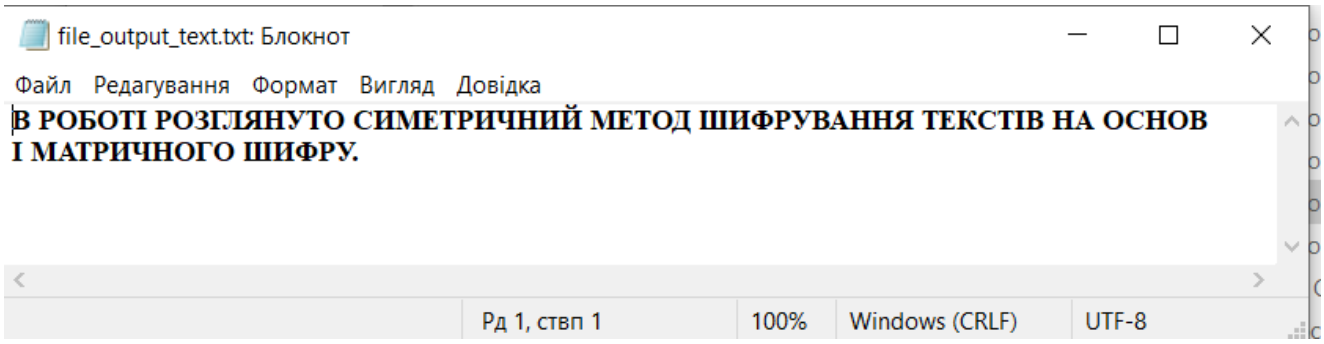


Рисунок 3.11 – Файл розшифрованого тексту

Аналіз одержаних результатів показує, що після проведеної операції шифрування відкритого тексту, наведеного на рис. 3.4, та операції розшифрування зашифрованого тексту, одержано початковий текст, представлений на рис. 3.11.

4 ТЕХНІКО-ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ

Завданням дипломного проєкту є розробка програмного забезпечення для пошуку найкоротших шляхів у мережах зв’язку. Ефект від використання проєктного рішення полягає у пришвидшені часу отримання результатів. Економічний ефект від використання проєктного рішення полягає в зменшенні часових витрат на роботу з кінцевим споживачем.

4.1 Розрахунок витрат на розробку та впровадження проєктного рішення.

Витрати на розробку та впровадження програмних засобів (К) включають:

$$К=К1+К2, \quad (4.1)$$

де К1 – витрати на розробку програмного продукту, грн.; К2 – витрати на налагодження та дослідну експлуатацію програмного засобу на ПК, грн.

Витрати на розробку програмного засобу включають в себе:

– Витрати на оплату праці розробників (З);
– Нарахування на зарплату (НЗ);
– Витрати на куповані витрати (ВК);
– Накладні витрати (НВ);
– Інші витрати (Інші).

Для розробки програмного продукту потрібні чотири спеціалісти-розробники, а саме:

– Керівник проєкту (К);
– Студент дипломник (СД);
– Консультант з охорони праці (КОП);
– Консультант з економічної частини (КЕ).

Згідно з штатним розписом Відокремленого структурного підрозділу «Фахового коледжу інформаційних технологій Національного університету «Львівська політехніка» місячною стипендією студента-дипломника є 1510 грн, а 1 година робочого навантаження становить для:

– Керівника проекту - 118,13 грн;
– Консультанта з економічної частини - 118,13 грн;
– Консультанта з охорони праці - 103,48 грн

Денна оплата студента дипломника визначається:

$$1510/173 = 8{,}73 \text{ (грн)},$$

де 173- місячний фонд робочого часу, годин.

Розрахунок витрат на оплату праці всіх спеціалістів проєкту визначається за формулою:

$$В_{оп} \sum_{i=0}^{N} n_j * t_j * ЗП\ г_i ; \quad (4.2)$$

де n_j – чисельність розробників проєкту 1-ої спеціальності чол.;t_j – час, витрачений на розробку проєкту працівником 1-ої спеціальності, дні; ЗП $г_i$ – погодинна заробітна плата розробника 1-ої спеціальності, грн;

Таким чином, витрати на оплату праці розробників складають:

- Зк = 1*14*118,13 = 1653,82 (грн);
- Зке = 1*1*118,13 = 118,13 (грн);
- Зкоп = 1*1*103,48 = 103,48 (грн);
- Зст = 1*180*8,73 = 1571,40 (грн.).

Сумарні витрати на оплату праці:

$$Воп= З_к+ З_{ке}+ З_{ст}+ З_{коп.}$$

Відповідно,

Воп = 1653,82 +118,13+103,48+1571,40=3446,83 (грн.)

Розрахунок витрат на оплату праці розробників наведено у таблиці 4.1.

Таблиця 4.1 – Розрахунок витрат на оплату праці

Спеціальність розробника	Кількість розробників роб.	Час роботи, год.	Погодинна заробітна плата розробника, грн.	Витрати на оплату праці, грн.
Керівник проєкту	1	14	118,13	1653,82
Консультант з економічної частини	1	1	118,13	118,13
Консультант з охорони праці	1	1	103,48	103,48
Студент-дипломник	1	180	8,73	1571,40
Всього	-	-	-	3446,83

Нарахування на зарплату становить згідно з нормативом 22% від фонду оплати праці:

$$Нз= Воп•22,0/100, \quad (4.3)$$

де Воп – витрати на оплату праці, тис.грн.; 22 – норматив нарахувань на зарплату,%.

НЗ = (1653,82+118,13+103,48)*0,22 = 412,60 (грн).

4.2 Розрахунок витрат на куповані вироби

Витрати на куповані вироби (папір, друк) визначаються за їх фактичними цінами з врахуванням найменування, номенклатури та необхідної кількості в проєкті. Транспортно-заготівельні витрати становлять 10% від суми витрат на куповані вироби. Розрахунок витрат на куповані вироби наведено в табл. 4.2.

Таблиця 4.2 – Розрахунок витрат на куповані вироби

Найменування купованих виробів	Марка, тип	Кількість на розробку, шт.	Ціна за одиницю, грн.	Сума витрат, грн.
Папка для проєкту	Формат А4	1	180	180
Папір, пачок	Формат А4	1	210	210
Роздрук пояснювальної записки	Формат А4	100	3	300
Разом	–	–	–	690
Транспортно-заготівельні витрати (10%)	–	–	–	69
Всього	–	–	–	759

Витрати на куповані вироби становлять:

$$Вк = 180+ 210 + 300 + 69 = 759,00 \text{ (грн.)}.$$

4.3 Розрахунок накладних та інших витрат

Накладні витрати (Нв) розраховуються за встановленими відсотками (30%) до витрат на оплату праці:

$$Нв = 3446,83\ 30/100 = 1034,05 \text{ (гри)}.$$

Інші витрати розраховуються по їх питомій вазі у структурі собівартості (10%):

$$Він = (Воп+Нз+Нв+Вк)•10/90; \quad (4.4)$$

$$Він = (3446,83+412,60 + 1034,05 + 759,00)*10/90 = 628,05 \text{ (грн)};$$

Витрати на розробку проєктного рішення визначаються за формулою:

$$К1=Воп+Нз+Нв+Вк+Він; \quad (4.5)$$

$$К1 = 3446,83 +412,60 + 1034,05 +759,00 + 628,05 = 6280,53 \text{ (грн)};$$

4.4 Розрахунок витрат на налагодження проєктного рішення

Програма була розроблена протягом 45 днів із розрахунком 4 год на день (t=180 год.).

Потужність обчислювальної техніки (Р) включає ноутбук, який споживає 0,47 кВт*год. Отже, вартість однієї машино-години роботи визначається за формулою:

$$Sm.r.= Р•Тф, \quad (4.6)$$

де Р – потужність комп'ютерної техніки, кВт; Тф - вартість 1 кВт-год електроенергії, грн. (Тф-7,50).

$$Sm.r = 0,47* 7,50= 7,97 \text{ (грн/год)}.$$

Витрати на налагодження та дослідну експлуатацію програмного продукту на ПК визначаються за формулою:

$$К2= Sm.r \bullet t, \qquad (4.7)$$

де Sm.r. – вартість однієї машино-години роботи, грн\год; t – машинний час, витрачений на налагодження та дослідну експлуатацію програмного продукту, год.

$$К2 = 7,97*180 = 1434,60 \text{ (грн)}.$$

Таким чином, витрати на розробку та впровадження програмного продукту становлять:

$$К = 6280,53 +1434,60 =7715,13 \text{ (грн)}.$$

Кошторис витрат на розробку та впровадження проєктного рішення наведений в таблиці 4.3.

Таблиця 4.3 Кошторис витрат розробки та реалізації програмного продукту.

Найменування елементів витрат	Сума витрат, грн.
Витрати на оплату праці	3446,83
Нарахування на зарплату	412,60
Витрати на куповані вироби	759,00
Накладні витрати	1034,05
Інші витрати	628,05
Витрати на налагодження та дослідну експлуатацію	1434,60
Всього (К=К1+К2)	7715,13

Таким чином, загальні витрати на розробку та впровадження проєктного рішення становлять 7715,13 (грн.).

5 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ

5.1 Розміщення робочих місць в комп’ютерному класі

Робота над дипломним проєктом виконувалася в комп’ютерному класі. Програмне забезпечення розроблялося на персональному комп’ютері (ПК). Для друкування матеріалів пояснювальної записки використовувався принтер HP LaserJet-P1005. При роботі на комп’ютері впливають такі шкідливі фактори:

– підвищена температура зовнішнього середовища;
– підвищений рівень шуму;
– недостатня освітленість робочого місця.

Охорона праці регулюється законом України «Про охорону праці». Закон України визначає основні положення щодо реалізації конституційного права громадян на охорону їх життя і здоров'я у процесі трудової діяльності, регулює за участю відповідних державних органів відносини між власником організації або уповноваженим ним органом (далі – власник) і працівником з питань безпеки, гігієни праці та виробничого середовища і встановлює єдиний порядок організації охорони праці в Україні. Прийнятий 14 жовтня 1992 р.; закон діє у редакції від 21 листопада 2002 р.

Приміщення комп'ютерного класу і його розміри повинні відповідати кількості працюючих і розміщених технічних засобів. Не дозволяється розташовувати комп’ютерні приміщення з робочими місцями у підвалах і цокольних поверхах. Не можна розташовувати поряд приміщення категорій А і Б, а також виробництва з мокрими технологічними процесами.

Згідно ДСанПІН 3.3.2.007-98 площа приміщення на одного працюючого з ПК повинна складати не менше 6 м^2, а об’єм – не менше 20 м^3.

Робочі місця користувачів ПК розташовуються відносно світлових прорізів так, щоб природне світло падало збоку переважно зліва, а від стіни з віконними прорізами — на відстані 1,5 м, від інших стін на відстані – 1 м, відстань між столами становить 1,5 м.

При розміщенні робочого місця поряд з вікном кут між екраном дисплея і площиною вікна рівний 90° (для виключення відблисків). При розміщенні робочих столів для комп'ютерів необхідно дотримуватися такої відстані: 3 м між бічними поверхнями ПК, відстань від тильної поверхні одного ПК до тильної поверхні екрана іншого ПК – 1,5 м, а також відстань від тильної поверхні одного ПК до екрана іншого ПК – 2,5м. План розміщення робочих місць для роботи на персональному комп'ютері зображено на рисунку 5.1.

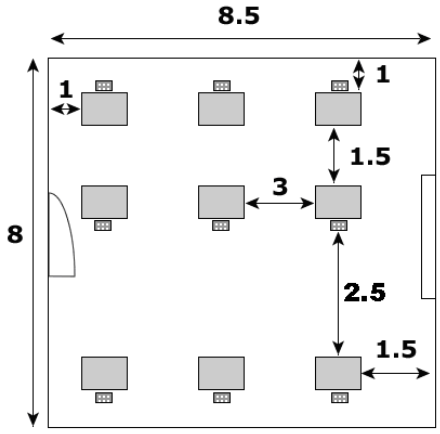


Рисунок 5.1– Розміщення робочих місць для роботи на ПК

Для внутрішнього оздоблення приміщень з персональними комп'ютерами слід використовувати дифузно-відбивні матеріали з коефіцієнтами відбиття для стелі 0,7-0,8, для стін 0,5-0,6. Покриття підлоги повинне бути матовим. У приміщеннях для розміщення комп'ютерів слід щоденно робити вологе прибирання. Крім того, ці приміщення мають бути оснащені аптечками першої медичної допомоги [10, 15, 16].

5.2 Електробезпека в комп'ютерному приміщенні

Персональні комп'ютери та інші периферійні пристрої повинні мати апаратуру захисту від струму короткого замикання та інших аварійних режимів. Під час монтажу та експлуатації ліній електромережі необхідно повністю унеможливити виникнення електричного джерела загоряння внаслідок короткого замикання та перевантаження проводів.

Безпечна напруга — це напруга не більше 42 В змінного струму та не більше 120 В постійного струму, що застосовується в електричних колах для зменшення небезпеки ураження електричним струмом. Найбільший ступінь безпеки досягається за напруги до 12 Вольт. У виробництві частіше використовують мережі напругою 12 Вольт та 36 Вольт. Для створення таких напруг, використовують знижувальні трансформатори.

Заземлення обладнання виконано відповідно ДНАОП 0.00-1.21-98 „Правила безпечної експлуатації електроустановок споживачів”. Схему розташування вертикального заземлювача захисного заземлення зображено на рис. 5.2.

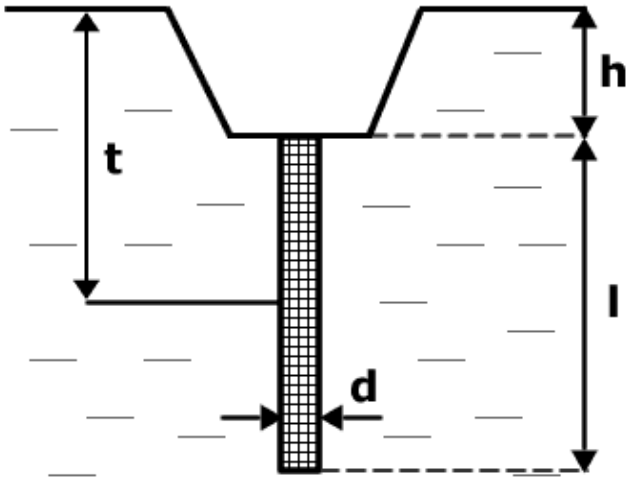


Рисунок 5.2 – Схема розташування вертикального заземлювача (t – відстань від поверхні ґрунту до середини заземлювача; d – діаметр заземлювача; h – глибина заземлювачі; l – довжина заземлювача)

Опір одиничного заземлення обчислюється за формулою:

$$R_1 = \frac{p}{2 * П * l}\left(\ln\frac{2l}{d} + \frac{1}{2}\ln\frac{4h+l}{4h-l}\right), \qquad (5.1)$$

де p – питомий опір ґрунту (p = 40 Ом/м); l – довжина планки кутникової сталі, м; d – діаметр планки кутникової сталі, м; h – глибина розташування заземлювача в ґрунті, м;

Використовуючи формулу (5.1) для розмірів заземлювача: d=0,06, l=3, h=0,7 м, одержимо

$$R_1=40/(2*3,14*3)*\ln((2*3/0,06)+0,5\ln\frac{4*0,7+3}{4*0,7-3})=19,53.$$

Визначаємо опір горизонтального заземлювача R_2, прокладеного на глибині h від поверхні землі, за формулою:

$$R_2=\frac{p}{2*П*L}*\ln*\frac{2*L^2}{b*h}, \quad (5.2)$$

Використовуючи формулу (5.2), одержимо:

$$R_2 = 40/(2*3,14*12)*ln(2*12^2)/(0,03*0,7) = 4,92\ Ом.$$

Загальний опір системи захисного заземлення обчислюється за формулою

$$R_3=\frac{R_1*R_2}{n_2*R_2*n_3+R_1+R_2}, \quad (5.3)$$

де n_2 – коефіцієнт, що враховує взаємне екранування електродів; n_3 – коефіцієнт при замкненому контурі.

Підставивши ці значення в формулу (5.3), одержимо: R_3=0,975 Ом.

5.3 Пожежна профілактика

У приміщеннях з персональними комп’ютерами заборонено курити і розкидати запалені сірники, застосовувати в приміщеннях відкритий вогонь. Не можна користуватися у приміщеннях з ПК електрокип’ятильниками, електрочайниками, не залишати без нагляду ввімкнені в електромережу кондиціонери, комп’ютери, радіоприймачі тощо

Можливими джерелами виникнення пожежі є коротке замикання в силових лініях та перевантаження при нештатних ситуаціях проводів струмами, вищими за допустимі значення. Профілактичними заходами в комп’ютерному класі є:

– Правильний вибір конструкції електрообладнання, способу встановлення й класу ізоляції (опір ізоляції згідно з ПУЕ 500 кОм);

– Правильний вибір, монтаж і експлуатація електричних мереж;

– Електричний захист електричних мереж, електрообладнання (швидкодіючі реле, автоматичні вимикачі, запобіжники);

– Профілактика пожеж від перевантажень:

– При проектуванні необхідно правильно вибирати переріз провідників мереж і схем за допустимою густиною струму, щоб $I_{доп.} >= I_{р}$;

– У процесі експлуатації електричних мереж не можна включати додатково електроприймачі, якщо мережа на це не розрахована;

– Для захисту електрообладнання від струмів перевантаження найбільш ефективні автоматичні й електронні схеми захисту, вимикачі, теплові реле і плавкі запобіжники.

Для гасіння пожежі приміщення лабораторії повинно бути оснащене ручними вуглекислотним вогнегасником ВВК-1,4, розміщеним на видному та доступному місці згідно правил експлуатації та типових норм належності вогнегасників.

5.4 Вимоги до освітлення комп’ютерного класу

Згідно ДСанПІН 3.3.2.007-98 приміщення, що розглядається, повинне мати природне і штучне освітлення. Денне (природне) освітлення приміщення відбувається за системою бічного освітлення. Природне світло проникає у приміщення через два світлові прорізи (віконні отвори). Також наявні штори (жалюзі) з можливістю захисту працюючих від прямого попадання сонячних променів і регулювання рівня освітленості в приміщенні. Всередині приміщення стіни обклеєні світлими шпалерами, стеля побілена (переважає білий колір), у якості підлогового покриття використаний ламінат світлого кольору. В досліджуваному приміщенні використовується система загального рівномірного штучного освітлення. Підвісні лампи рівномірно розміщені на стелі.

5.5 Висновки до розділу з охорони праці

Охорона праці при роботі на персональних комп’ютерах є надзвичайно важливою, оскільки забезпечує безпеку життя і здоров’я людини та запобігає пошкодженню матеріальних цінностей. При виконанні дипломного проекту необхідно було дотримуватися електробезпеки, через роботу з електронними пристроями, як комп’ютер, пожежної безпеки через небезпеку пожежі при короткому замиканні або несправній проводці.

Добре налаштоване робоче місце допомагає зберегти здоров’я через правильне положення спини, шиї і т. д. Це також допоможе запобігти таким хворобам, як остеохондроз, сколіоз тощо. Правильно освітлення в кімнаті, де працюють з персональним комп’ютером, допомагає більш активно виконувати роботу.

Проаналізувавши безпечні умови виконання дипломного проекту на персональному комп’ютері, можна зробити висновок, що умови праці в комп’ютерному класі є задовільними. Для покращення умов праці відповідно до ДСанПІН 3.3.2.007-98 можна вказати наступні рекомендацій:

– у приміщенні повинні бути медичні аптечки першої допомоги;

у приміщенні слід щоденно проводити вологе прибирання;

ВИСНОВКИ

В дипломному проєкті проаналізовано алгоритми шифрування і дешифрування повідомлень при їх передачі по каналах зв’язку. Перетворення даних в незрозумілу форму здійснюється для забезпечення захисту інформації від доступу сторонніх осіб. Вибір алгоритму залежить від вимоги до швидкості роботи криптографічної системи та ступеня важливості захисту інформації.

В проєкті розглянуто симетричний метод шифрування текстів на основі матричного шифру. Особливістю цього шифру є можливість його використання для захисту даних, представлених у вигляді матричних масивів. Існує багато криптографічних алгоритмів, що використовуються для захисту інформації, але більшість із них орієнтовані на послідовну обробку скалярних даних. Але в системах зв'язку є потреба передавати двовимірні масиви та зображення.

Метод шифрування полягає в записі відкритого тексту в матрицю, стовпці та рядки якої переставляються в відповідності з порядком літер ключового слова.

Для шифрування відкритих текстів розроблено програмне забезпечення. Розроблені алгоритми та програми демонструють застосування матричного шифру для шифрування відкритих текстів і їх розшифрування. Програмний захисту інформації дозволяє оперативно і швидко шифрувати і розшифровувати секретні повідомлення великих об’ємів з малими затратами часу.

В компанії “Донбас Арена” реалізовані сучасні технології створення, підтримки та моніторингу систем комунікацій. Для передачі секретних даних по каналах зв’язку використовується система захисту інформації.

Для забезпечення сприятливих умов роботи на персональному комп’ютері при виконанні практичної частини дипломної роботи проведено розрахунок оптимальних характеристик, що відповідають нормам та правилам охорони праці, які регламентовані чинними нормативно-правовими актами.

ПЕРЕЛІК ПОСИЛАНЬ

1 Бедрій Я.І., Піча В.М. Безпека життєдіяльності.-Львів, 2009. - 216с.

2 Бредли Л. Джонс, Питер Єйткен, Освой самостоятельно C за 21 день, 6-е изд.: Пер. З англ. — М.: Издательский дом "Вильямс", 2003.- 800с.

3 Директива 97/66/ЄС Європейського Парламенту і Ради "Стосовно обробки персональних даних і захисту права на невтручання в особисте життя в телекомунікаційному секторі" від 15 грудня 1997 року // www.iu.org.ua

4 ДСТУ 3396.0-96 Захист інформації. Технічний захист інформації. Основні положення.

5 Ємець В., Мельника., Попович Р. Сучасна криптографія. Основні поняття. – Львів:Бак, 2003. – 144 с.

6 Закон України «Про електронний цифровий підпис» //ВВР, 2003, № 36, ст. 276.

7 Закон України "Про захист інформації в автоматизованих системах" // Відомості Верховної Ради України, 1994. - № 31. – С. 286.

8 Златопольський Д.М. Найпростіші методи шифрування тексту. / Д.М. Златопольський - М.: Чисті ставки, 2007

9 Конвенція про захист осіб стосовно автоматизованої обробки даних особистого характеру. Страсбург, 28 січня 1981 року // www.iu.org.ua

10 Кормич Б.А. Інформаційна безпека: організаційно-правові основи.– К., 2004.

11 Олійник О. В., Соснін О. В. Політико-правові аспекти формування інформаційного суспільства суверенної і незалежної держави // www.iu.org.ua

12 Організаційно-правові основи захисту інформації з обмеженим доступом. Навчальний посібник. / За ред. В. С. Сідака. – К., 2006.

13 Остапов С.Г., Валь Л.О. Основи криптографії. Чернівці: Книги – ХХІ, 2008. – 188 с.

14 Положення про порядок здійснення криптографічного захисту інформації в Україні: Указ... 22.05.98 р. № 505/98 // Уряд. кур'єр, 1998. – с. 2.

15 Проект Закону України “Про інформацію персонального характеру“ // www.khpg.org.ua

16 Про затвердження Концепції технічного захисту інформації в Україні: Постанова... 8 жовтня 1997 р. № 1126 // ДВУ, 1997. – № 12. – С. 1714.

17 Про захист осіб у зв’язку з автоматизованою обробкою персональних даних Електронний ресурс : Конвенція №108 Ради Європи. – Режим доступу : http://tzi.com.ua/konv_108.html

18 . Про захист персональних даних : Закон України від 1 червня 2010 року № 2297-VI // Відомості Верховної ради України. – 2010. – № 34. – Ст. 481.

19 Про правовий захист баз даних Електронний ресурс : Директива 96/9/ЄС Європейського Парламенту та Ради Європи. – Режим доступу : http://zakon4.rada.gov.ua/laws/show/994_24

20 Савченко С.В. Інформаційна безпека та організаційно-правовий захист інформації: Конспект лекцій. Дніпропетровськ, НМетАУ, 2008. − 52 с.

21 Сулятицький П.Р. Класичні методи шифрування інформації простою заміною / П.Р. Сулятицький, Ю.І. Грицюк // Науковий вісник НЛТУ України : зб. наук.-техн. праць. – Львів : РВВ НЛТУ України. – 2011. – Вип. 21.9. – С. 306-316.

22 Фаль О. М.Криптографія: основні ідеї та застосування. – К: Вид-во. НТУУ КПІ, 2004.

23 Цимбалюк В. С. Інформаційне право (основи теорії і практики). – К.: «Освіта України», 2010. – 235 с.

24 Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные коды на языке С. – М: Триумф, 2002. – 816с.

25 Яковлєв А.В., Безбогов А.А., Родін В.В., Шамкін В.М. Криптографічний захист інформації. / Навчальний посібник - Тамбов: Вид-во Тамбо. держ. техн. ун-ту, 2006.

ДОДАТОК А

Текст програми на мові С для шифрування текстів на базі матричного шифру перестановок

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>

FILE *fp1,*fp2, *fp3, *fp4;
char mode_w[4]="w";
char mode_r[4]="r";

char filename1[30]="d:\\file_input_text.txt";  /* fp1 */
char filename2[30]="d:\\file_shyf_text.txt";   /* fp2 */
char filename3[30]="d:\\file_key.txt";         /* fp3 */
char filename4[30]="d:\\file_matr.txt";        /* fp4 */

char unsigned  cumvol, cumvol1, cumvol11, cumvol21, cumvol22, cumvol23;
int cumvol_z, k=0,  i4;
 char unsigned KOD[100], KOD1;// c2[2];

 char unsigned text_mas[800];/*Масив для читання тексту*/
char unsigned  text_mas_shyf[800];/*Масив для шифрованого тексту*/
 char unsigned text_matr[255][255];/*Матриця для початкового тексту*/
char unsigned  text_matr_shyf[255][255];/*Матриця  для шифрованого  тексту*/
 char unsigned char_key[255]; /* Масив для читання символів ключа */
 int *key_MAS,*key_pos;/*Масиви для кодів ключа і номерів їх позицій*/
 int l_key=0 ; /* Довжина ключа  */
 int l_text, kr_m ; /* Довжина тексту, кількість рядуів матриці */ ;
 int im,i,j,k, r,pos,i_pos;

 /*---- Функція для присвоєння числових кодів буквам алфавіту -----*/
void kkk (char unsigned x)
{
 /* Задання літер алфавіту, яким задаються і шифруються тексти */
KOD[0]='А';
KOD[1]='Б';
KOD[2]='В';
KOD[3]='Г';
KOD[4]='Д';
KOD[5]='Е';
KOD[6]='Є';
KOD[7]=(char unsigned) 'Ж';
KOD[8]='З';
KOD[9]='И';
KOD[10]=(char unsigned)'І';
KOD[11]='Ї';
```

```
KOD[12]='Й';
KOD[13]='К';
KOD[14]='Л';
KOD[15]='М';
KOD[16]='Н';
KOD[17]='О';
KOD[18]='П';
KOD[19]='Р';
KOD[20]='С';
KOD[21]='Т';
KOD[22]='У';
KOD[23]='Ф';
KOD[24]='Х';
KOD[25]='Ц';
KOD[26]='Ч';
KOD[27]='Ш';
KOD[28]='Щ';
KOD[29]='Ь';
KOD[30]='Ю';
KOD[31]='Я';

switch (x)
   {
case (unsigned char)'А': KOD1=0;  break;
case (unsigned char)'Б': KOD1=1;  break;
case (unsigned char)'В': KOD1=2;   break;
case (unsigned char) 'Г': KOD1=3;   break;
case (unsigned char)'Д':  KOD1=4;   break;
case (unsigned char)'Е':  KOD1=5;   break;
case (unsigned char)'Є': KOD1=6;   break;
case (unsigned char)'Ж': KOD1=7;   break;
case (unsigned char)'З': KOD1=8;    break;
case (unsigned char) 'И': KOD1=9;     break;
case (unsigned char)'І': KOD1=10;   break;
case (unsigned char)'Ї': KOD1=11;   break;
case (unsigned char)'Й': KOD1=12;    break;
case (unsigned char)'К':  KOD1=13;  break;
case (unsigned char)'Л': KOD1=14;   break;
case (unsigned char)'М':  KOD1=15; break;
case (unsigned char)'Н': KOD1=16;    break;
case (unsigned char)'О': KOD1=17;  break;
case (unsigned char)'П': KOD1=18;   break;
case (unsigned char)'Р':KOD1=19;   break;
case (unsigned char)'С':   KOD1=20;   break;
case (unsigned char)'Т': KOD1=21;   break;
case (unsigned char)'У':  KOD1=22;   break;
case (unsigned char)'Ф': KOD1=23;    break;
case (unsigned char)'Х':  KOD1=24; break;
case (unsigned char)'Ц':  KOD1=25;  break;
case (unsigned char)'Ч': KOD1=26;   break;
case (unsigned char)'Ш': KOD1=27;   break;
case (unsigned char)'Щ': KOD1=28;   break;
```

```
case (unsigned char)'Ь': KOD1=29;    break;
case (unsigned char)'Ю': KOD1=30;   break;
case (unsigned char)'Я': KOD1=31;   break;

default: cumvol1=cumvol; KOD1=176;
        }
     //   printf("\nkod1=%d\t", KOD1);
  /* Кодування літер   */
 switch (KOD1)
   {
case 0:  fprintf(fp2, "%c" ,KOD[0]);  break;
case 1:  fprintf(fp2, "%c" ,KOD[1]); break;
case 2:  fprintf(fp2, "%c" ,KOD[2]);   break;
case 3:  fprintf(fp2, "%c" ,KOD[3]);   break;
case 4:    fprintf(fp2, "%c" ,KOD[4]);  break;
case 5: fprintf(fp2, "%c" ,KOD[5]); break;
case 6:  fprintf(fp2, "%c" ,KOD[6]);   break;
case 7:   fprintf(fp2, "%c" ,KOD[7]);  break;
case 8:   fprintf(fp2, "%c" ,KOD[8]);   break;
case 9:  fprintf(fp2, "%c" ,KOD[9]);   break;
case 10:   fprintf(fp2, "%c" ,KOD[10]);   break;
case 11:  fprintf(fp2, "%c" ,KOD[11]);   break;
case 12:  fprintf(fp2, "%c" ,KOD[12]);   break;
case 13:  fprintf(fp2, "%c" , KOD[13]); break;
case 14:   fprintf(fp2, "%c" ,KOD[14]);   break;
case 15:  fprintf(fp2, "%c" ,KOD[15]);   break;
case 16:  fprintf(fp2, "%c" ,KOD[16]);   break;
case 17:  fprintf(fp2, "%c" ,KOD[17]);   break;
case 18:   fprintf(fp2, "%c" ,KOD[18]);   break;
case 19:   fprintf(fp2, "%c" ,KOD[19]);   break;
case 20:   fprintf(fp2, "%c" ,KOD[20]);   break;
case 21:   fprintf(fp2, "%c" ,KOD[21]);   break;
case 22:   fprintf(fp2, "%c" ,KOD[22]);   break;
case 23:   fprintf(fp2, "%c" ,KOD[23]);   break;
case 24:   fprintf(fp2, "%c" ,KOD[24]);   break;
case 25:   fprintf(fp2, "%c" ,KOD[25]);   break;
case 26:   fprintf(fp2, "%c" ,KOD[26]);   break;
case 27:   fprintf(fp2, "%c" ,KOD[27]);   break;
case 28:   fprintf(fp2, "%c" ,KOD[28]);   break;
case 29:   fprintf(fp2, "%c" ,KOD[29]);   break;
case 30:   fprintf(fp2, "%c" ,KOD[30]);   break;
case 31:   fprintf(fp2, "%c" ,KOD[31]);   break;
default: fprintf(fp2, "%c" ,cumvol1); //printf("\nzz=%d" , cumvol);
        }
          }
void  main()
{
   /* Відкриття файлу file_input.txt, який містить відкритий текст */
fp1=fopen(filename1,mode_r);
if (fp1!=NULL ) {printf("file %s open mode %s\n",filename1, mode_r); }
else    { printf("file %s not open mode  %s\n", filename1, mode_r);  exit(1);}
    /* Відкриття файлу file_text.txt, для запису шифрованого тексту */
```

```
fp2=fopen(filename2,mode_w);
if (fp2!=NULL ) {printf("file %s open mode %s\n",filename2, mode_w); }
else    { printf("file %s not open mode 3 %s\n", filename2, mode_w); exit(2);}
fp3=fopen(filename3,mode_r);
if (fp3!=NULL ){printf("file %s open mode %s\n",filename3, mode_r);}
else    { printf("file %s not open mode %s\n",filename3, mode_r);  exit(3);}
/* Відкриття файлу d:file_matr.txt для запису матриці тексту*/
fp4=fopen(filename4,mode_w);
if (fp4!=NULL ){printf("file %s open mode %s\n",filename4, mode_w);}
else    { printf("file %s not open mode  %s\n",filename4, mode_w);  exit(4);}
// char  unsigned TB_PL[6][6], TL[36] ;
//int m,n, m1,n1, m2[2],n2[2], l=-1;
i=0;
key_pos=(int *)calloc(1, sizeof(int));

/* Читання символів ключа з файлу d:\\file_key.txt */
l_key=l_key+1;
m0:
cumvol=fgetc(fp3);

if( cumvol!=255)

 {  if ((cumvol!=10) )
 if (cumvol!=208)
                        {cumvol21=cumvol;

 char_key[i]=cumvol21;
/* Формуванням масиву key_pos  номерів  позицій символів ключа */
key_pos[i]=cumvol21-143;
i++;
key_pos=(int *)realloc(key_pos, i+1);

//printf( "\n21_21=%d\n" ,cumvol21);

 if (cumvol21<132) fprintf(fp4,"%c" , cumvol21); else
 {
fprintf(fp4, "%c" ,208);
fprintf(fp4, "%c\t" ,cumvol21);

  }
 }
  goto m0;
 }
  fprintf(fp4,"\n\n");
l_key=strlen(char_key );  /* Довжина ключа */
//l_key=strlen(char_key )-1;
 printf( "\n Довжина ключового сллова l_key=%d\n" ,l_key);
//l_key=6;  /* Довжина ключа */
key_MAS=(int *)calloc(l_key, sizeof(int));

 fprintf(fp4,"\n");
```

```
key_pos[0]=4;key_pos[1]=6;key_pos[2]=5;
key_pos[3]=2;key_pos[4]=1;key_pos[5]=3;
//for (i=0; i<l_key; i++)
//printf("key_pos[%d]=%d\n",i,key_pos[i]);
//for (j=0; j<l_key; j++)
// fprintf(fp4,"%c\t",char_key[j]);
// fprintf(fp4,"\n\n");
 for (j=0; j<l_key; j++)
 fprintf(fp4,"%d\t",key_pos[j]);
fprintf(fp4,"\n-------------------------------------------------------------------------\n");

/* Читання символів відкритого  тексту з файлу file_input.txt */

i=0;
m1:
cumvol=fgetc(fp1);

if( cumvol!=255)

 {  if ((cumvol!=10) )
 if (cumvol!=208)
                          {cumvol21=cumvol;
 //printf( "\n21_21=%d\n" ,cumvol21);

// if (cumvol21<132)
// {fprintf(fp2,"%c" , cumvol21); text_mas[i]=cumvol21; i++;}
//  else
// {  fprintf(fp2, "%c" ,208);  text_mas[i]=cumvol21; i++;
//  kkk(cumvol21);}


text_mas[i]=cumvol21; i++;
 }
  goto m1;
 }
l_text=strlen(text_mas)-1;  /* Довжина тексту */
 printf( "\n  Довжина початкового тексту  l_text=%d\n" ,l_text);
if (l_text%l_key==0) {kr_m= l_text/l_key;}
else                 {kr_m= l_text/l_key+1;}
 r=l_text-(l_text/l_key)*l_key;
 printf(" \n Кількість рядків матриці  kr_m=%d\n", kr_m);
/*Перетворення тексту в матрицю і запис у файл d:\\file_matr.txt */
k=0; for (i=0; i<kr_m; i++)
for (j=0; j<l_key; j++)
{text_matr[i][j]= text_mas[k];k++ ;}

/*  Доповнення тексту будь-якими літерами */
if (r!=0) for (j=r; j<l_key; j++)
{text_matr[kr_m-1][j]= '*';k++ ;}

for (i=0; i<kr_m; i++)
{for (j=0; j<l_key; j++)
```

```
if (text_matr[i][j]<132) fprintf(fp4,"%c\t" , text_matr[i][j]); else
 {
fprintf(fp4, "%c" ,208);
fprintf(fp4, "%c\t" ,text_matr[i][j]);
  }
  fprintf(fp4,"\n");
 }
fprintf(fp4,"-------------------------------------------------------------------------\n\n\n");

for (j=0; j<l_key; j++)  fprintf(fp4,"%d \t",j+1);
fprintf(fp4,"\n-------------------------------------------------------------------------\n");

/*Шифрування тексту і запис шифроматриці в файл d:\\file_matr.txt */
 for (i=0; i<kr_m; i++)
 for (j=0; j<l_key; j++)
 text_matr_shyf[i][key_pos[j]-1]=text_matr[i][j];
for (i=0; i<kr_m; i++)
{for (j=0; j<l_key; j++)

if (text_matr_shyf[i][j]<132) fprintf(fp4,"%c\t" , text_matr_shyf[i][j]); else
 {
fprintf(fp4, "%c" ,208);
fprintf(fp4, "%c\t" ,text_matr_shyf[i][j]);
  }
 fprintf(fp4,"\n");}
fprintf(fp4,"-------------------------------------------------------------------------\n");

/*Перетворення  шифроматриці в текст і запис у файл d:\\file_shyf_text.txt */
k=0; im=0;
for (j=0; j<l_key; j++)
for (i=0; i<kr_m; i++)
{text_mas_shyf[k]=text_matr_shyf[i][j];

if (text_mas_shyf[k]<132)
 {fprintf(fp2,"%c" , text_mas_shyf[k]);im++; }
  else
 {  fprintf(fp2, "%c" ,208);
  kkk(text_mas_shyf[k]);im++;}
k++;

if (im==65 )    {
fprintf(fp2, "\n" );  im=0;}
}

/* Закриття файлів d:\\file_input_text.txt, d:\\file_shyf_text.txt,
                d:\\file_key.txt, d:\\file_matr.txt*/
fclose(fp1);
fclose(fp2);
fclose(fp3);
fclose(fp4);
                                                    }
```

ДОДАТОК Б

Текст програми на мові С для розшифрування текстів на базі матричного шифру перестановок

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

FILE *fp1,*fp2, *fp3, *fp4;
char mode_w[4]="w";
char mode_r[4]="r";

char filename1[30]="d:\\file_shyf_text.txt";  /* fp1 */
char filename2[30]="d:\\file_output_text.txt";/* fp2 */
char filename3[30]="d:\\file_key.txt";        /* fp3 */
char filename4[30]="d:\\file_deshyf_matr.txt";/* fp4 */
char unsigned  cumvol, cumvol1, cumvol11, cumvol21, cumvol22, cumvol23;
int cumvol_z, k=0,  i4;
 char unsigned KOD[100], KOD1;
  char unsigned shyf_mas[800], shyf_mas_text[800];
 char unsigned shyf_matr[255][255], shyf_matr_text[255][255];

 char unsigned char_key[255]; /* Масив для читання символів ключа */
 int *key_MAS,*key_pos;/*Масиви для кодів ключа і номерів їх позицій*/
 int l_key=0 ; /* Довжина ключа  */
 int l_text, kr_m ; /* Довжина тексту, кількість рядуів матриці */ ;
 int im,i,j,k, r,pos,i_pos;

void kkk (char unsigned x)
{
 /* Задання літер алфавіту, яким задаються і шифруються тексти */
KOD[0]='А';
KOD[1]='Б';
KOD[2]='В';
KOD[3]='Г';
KOD[4]='Д';
KOD[5]='Е';
KOD[6]='Є';
KOD[7]=(char unsigned) 'Ж';
KOD[8]='З';
KOD[9]='И';
KOD[10]=(char unsigned)'І';
KOD[11]='Ї';
KOD[12]='Й';
KOD[13]='К';
KOD[14]='Л';
KOD[15]='М';
KOD[16]='Н';
```

```
KOD[17]='О';
KOD[18]='П';
KOD[19]='Р';
KOD[20]='С';
KOD[21]='Т';
KOD[22]='У';
KOD[23]='Ф';
KOD[24]='Х';
KOD[25]='Ц';
KOD[26]='Ч';
KOD[27]='Ш';
KOD[28]='Щ';
KOD[29]='Ь';
KOD[30]='Ю';
KOD[31]='Я';

switch (x)
   {
case (unsigned char)'А': KOD1=0;  break;
case (unsigned char)'Б': KOD1=1;  break;
case (unsigned char)'В': KOD1=2;   break;
case (unsigned char) 'Г': KOD1=3;   break;
case (unsigned char)'Д':  KOD1=4;   break;
case (unsigned char)'Е':  KOD1=5;   break;
case (unsigned char)'Є': KOD1=6;   break;
case (unsigned char)'Ж': KOD1=7;   break;
case (unsigned char)'З': KOD1=8;    break;
case (unsigned char) 'И': KOD1=9;     break;
case (unsigned char)'І': KOD1=10;   break;
case (unsigned char)'Ї': KOD1=11;   break;
case (unsigned char)'Й': KOD1=12;    break;
case (unsigned char)'К':  KOD1=13;  break;
case (unsigned char)'Л': KOD1=14;   break;
case (unsigned char)'М':  KOD1=15; break;
case (unsigned char)'Н': KOD1=16;    break;
case (unsigned char)'О': KOD1=17;  break;
case (unsigned char)'П': KOD1=18;   break;
case (unsigned char)'Р':KOD1=19;   break;
case (unsigned char)'С':   KOD1=20;   break;
case (unsigned char)'Т': KOD1=21;   break;
case (unsigned char)'У':  KOD1=22;   break;
case (unsigned char)'Ф': KOD1=23;    break;
case (unsigned char)'Х':  KOD1=24; break;
case (unsigned char)'Ц':  KOD1=25;  break;
case (unsigned char)'Ч': KOD1=26;   break;
case (unsigned char)'Ш': KOD1=27;   break;
case (unsigned char)'Щ': KOD1=28;   break;
case (unsigned char)'Ь': KOD1=29;    break;
case (unsigned char)'Ю': KOD1=30;   break;
case (unsigned char)'Я': KOD1=31;   break;

default: cumvol1=cumvol; KOD1=176;
```

```
        }
     //   printf("\nkod1=%d\t", KOD1);
 /* Кодування літер   */
 switch (KOD1)
   {
case 0:  fprintf(fp2, "%c" ,KOD[0]);  break;
case 1:  fprintf(fp2, "%c" ,KOD[1]); break;
case 2:  fprintf(fp2, "%c" ,KOD[2]);   break;
case 3:  fprintf(fp2, "%c" ,KOD[3]);   break;
case 4:    fprintf(fp2, "%c" ,KOD[4]);  break;
case 5: fprintf(fp2, "%c" ,KOD[5]); break;
case 6:  fprintf(fp2, "%c" ,KOD[6]);   break;
case 7:   fprintf(fp2, "%c" ,KOD[7]);  break;
case 8:   fprintf(fp2, "%c" ,KOD[8]);   break;
case 9:  fprintf(fp2, "%c" ,KOD[9]);   break;
case 10:   fprintf(fp2, "%c" ,KOD[10]);   break;
case 11:  fprintf(fp2, "%c" ,KOD[11]);   break;
case 12:  fprintf(fp2, "%c" ,KOD[12]);   break;
case 13:  fprintf(fp2, "%c" , KOD[13]); break;
case 14:   fprintf(fp2, "%c" ,KOD[14]);   break;
case 15:  fprintf(fp2, "%c" ,KOD[15]);   break;
case 16:  fprintf(fp2, "%c" ,KOD[16]);   break;
case 17:  fprintf(fp2, "%c" ,KOD[17]);   break;
case 18:   fprintf(fp2, "%c" ,KOD[18]);   break;
case 19:   fprintf(fp2, "%c" ,KOD[19]);   break;
case 20:   fprintf(fp2, "%c" ,KOD[20]);   break;
case 21:   fprintf(fp2, "%c" ,KOD[21]);   break;
case 22:   fprintf(fp2, "%c" ,KOD[22]);   break;
case 23:   fprintf(fp2, "%c" ,KOD[23]);   break;
case 24:   fprintf(fp2, "%c" ,KOD[24]);   break;
case 25:   fprintf(fp2, "%c" ,KOD[25]);   break;
case 26:   fprintf(fp2, "%c" ,KOD[26]);   break;
case 27:   fprintf(fp2, "%c" ,KOD[27]);   break;
case 28:   fprintf(fp2, "%c" ,KOD[28]);   break;
case 29:   fprintf(fp2, "%c" ,KOD[29]);   break;
case 30:   fprintf(fp2, "%c" ,KOD[30]);   break;
case 31:   fprintf(fp2, "%c" ,KOD[31]);   break;
default: fprintf(fp2, "%c" ,cumvol1); //printf("\nzz=%d" , cumvol);
        }
         }
int  main()
{ /* Відкриття файлу d:\\file_shyf_text.txt, для читання шмфротексту*/

fp1=fopen(filename1,mode_r);
if (fp1!=NULL ) {printf("file %s open mode %s\n",filename1, mode_r); }
else    { printf("file %s not open mode  %s\n", filename1, mode_r);   exit(1);}

/*Відкриття файлу d:\\File_output_text.txt, для запису початкового тексту*/

fp2=fopen(filename2,mode_w);
if (fp2!=NULL ) {printf("file %s open mode %s\n",filename2, mode_w); }
else    { printf("file %s not open mode %s\n", filename2, mode_w);  exit(2);}
```

```
/*Відкриття файлу d:\\file_key.txt,для читання ключа */

fp3=fopen(filename3,mode_r);
if (fp3!=NULL ){printf("file %s open mode %s\n",filename3, mode_r);}
else    { printf("file %s not open mode %s\n",filename3, mode_r);  exit(3);}

/* Відкриття файлу d:\\File_deshyf_matr.txt для запису матриці тексту*/

fp4=fopen(filename4,mode_w);
if (fp4!=NULL ){printf("file %s open mode %s\n",filename4, mode_w);}
else    { printf("file %s not open mode  %s\n",filename4, mode_w);  exit(4);}

 /****    Читання ключа ****/
i=0;
key_pos=(int *)calloc(1, sizeof(int));

/* Читання символів ключа з файлу d:\\file_key.txt */
l_key=l_key+1;
m0:
cumvol=fgetc(fp3);

if( cumvol!=255)

 {  if ((cumvol!=10) )
 if (cumvol!=208)
                          {cumvol21=cumvol;

 char_key[i]=cumvol21;
/* Формуванням масиву key_pos  номерів  позицій символів ключа */
key_pos[i]=cumvol21-143;
i++;
key_pos=(int *)realloc(key_pos, i+1);

//printf( "\n21_21=%d\n" ,cumvol21);

 if (cumvol21<132) fprintf(fp4,"%c" , cumvol21); else
 {
fprintf(fp4, "%c" ,208);
fprintf(fp4, "%c\t" ,cumvol21);

  }
 }
  goto m0;
 }


fprintf(fp4,"\n\n");


l_key=strlen(char_key );  /* Довжина ключа */
//l_key=strlen(char_key )-1;
```

```
 printf( "\n Довжина ключового сллова l_key=%d\n" ,l_key);
//l_key=6;  /* Довжина ключа */
key_MAS=(int *)calloc(l_key, sizeof(int));

 fprintf(fp4,"\n");


key_pos[0]=4;key_pos[1]=6;key_pos[2]=5;
key_pos[3]=2;key_pos[4]=1;key_pos[5]=3;

for (j=0; j<l_key; j++)
 fprintf(fp4,"%d \t",j+1);
 fprintf(fp4,"\n------------------------------------------------------------------------\n");

i=0;
m1:
cumvol=fgetc(fp1);

if( cumvol!=255)

 {  if ((cumvol!=10) )
 if (cumvol!=208)
                          {cumvol21=cumvol;

shyf_mas[i]=cumvol21; i++;
 }
  goto m1;
 }

 l_text=strlen(shyf_mas);  /* Довжина тексту */
 printf( "\n  Довжина шиифротексту  l_text=%d\n" ,l_text);

kr_m= l_text/l_key;
 printf("l_text=%d\tkr_m=%d\n",l_text, kr_m);

/*Перетворення шмфротексту в шифроиатрицю і запис у файл  d:\\File_deshyf_matr.txt*/
k=0;
for (j=0; j<l_key; j++)
for (i=0; i<kr_m; i++)
{shyf_matr[i][j]= shyf_mas[k];k++ ;}

for (i=0; i<kr_m; i++)
{for (j=0; j<l_key; j++)
if (shyf_matr[i][j]<132) fprintf(fp4,"%c\t" , shyf_matr[i][j]); else
 {
fprintf(fp4, "%c" ,208);
fprintf(fp4, "%c\t" ,shyf_matr[i][j]);
  }
  fprintf(fp4,"\n");
 }
fprintf(fp4,"------------------------------------------------------------------------\n\n\n");
```

```
for (j=0; j<l_key; j++)
 fprintf(fp4,"%d \t",key_pos[j]);
 fprintf(fp4,"\n--------------------------------------------------------------------------\n");

 fprintf(fp4,"\n");
/*Запис розшифрованого тексту у матрицю ( файл ) d:\\File_deshyf_matr.txt*/
 for (i=0; i<kr_m; i++)
 for (j=0; j<l_key; j++)
 shyf_matr_text[i][j]=shyf_matr[i][key_pos[j]-1];


for (i=0; i<kr_m; i++)
{for (j=0; j<l_key; j++)

if (shyf_matr_text[i][j]<132) fprintf(fp4,"%c\t" , shyf_matr_text[i][j]); else
 {
fprintf(fp4, "%c" ,208);
fprintf(fp4, "%c\t" ,shyf_matr_text[i][j]);
  }
 fprintf(fp4,"\n");}
fprintf(fp4,"--------------------------------------------------------------------------\n");

/*Запис розшифрованого тексту у файл  d:\\File_output_text.txt */

k=0;
for (i=0; i<kr_m; i++)
for (j=0; j<l_key; j++)
{shyf_mas_text[k]=shyf_matr_text[i][j];

if (shyf_mas_text[k]<132)
 {fprintf(fp2,"%c" , shyf_mas_text[k]);im++; }
  else
 {  fprintf(fp2, "%c" ,208);
  kkk(shyf_mas_text[k]);im++;}

if (im==65 )    {
fprintf(fp2, "\n" );  im=0;}
k++ ;}

/* Закриття файлів d:\\file_shyf_text.txt,d:\\File_output_text.txt,
                d:\\file_key.txt, d:\\File_deshyf_matr.txt */
fclose(fp1);
fclose(fp2);
fclose(fp3);
fclose(fp4);
return 0;
   }
```