МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ КОЛЕДЖ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЗЕМЛЕВПОРЯДКУВАННЯ НАУ

Дипломний проект

Тема: «Програмний модуль шифрування даних»

Виконавець: Гуленко А.Ю.
Керівник: Нечипорук О.П.

Актуальність

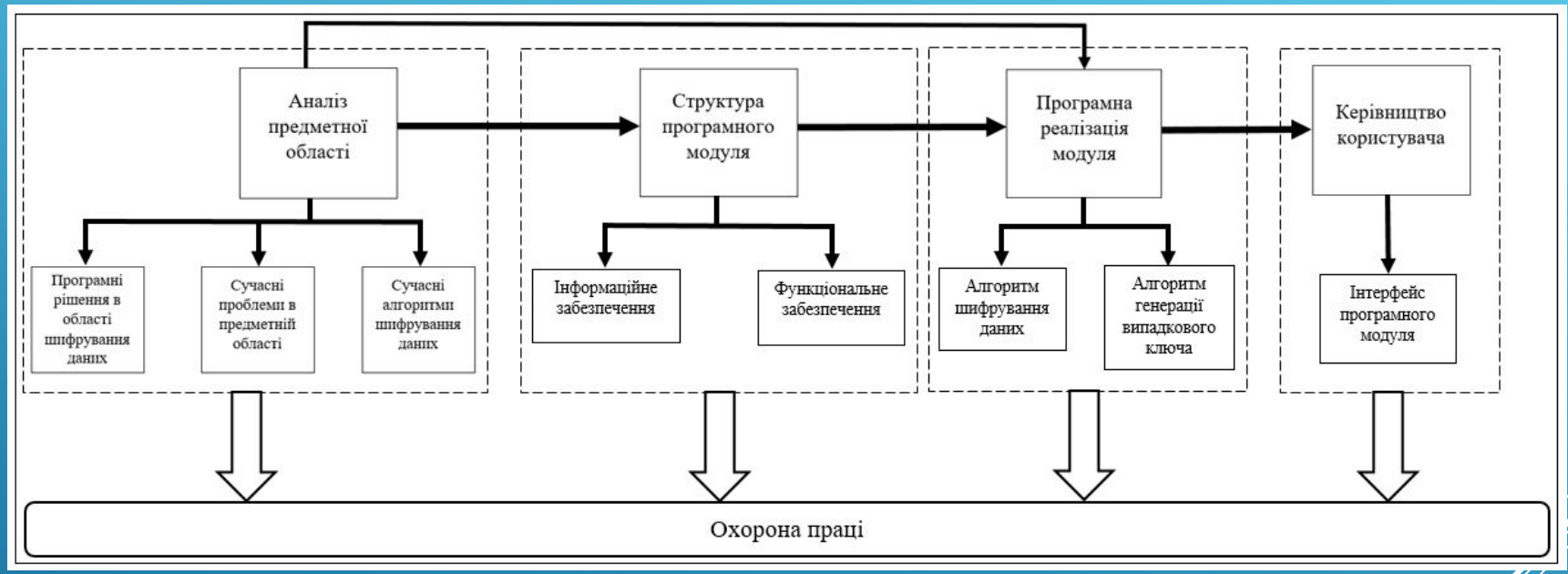
Методи шифрування	Переваги	Недоліки
Симетричне шифрування	1) Висока швидкість шифрування. 2) Менша довжина ключа, ніж в асиметричному шифруванні. 3) Проста реалізація.	1) Публічна передача ключів, враховуючи велику ймовірність порушення секретності ключа. 2) Квадратична залежність числа ключів при великій кількості користувачів.
Асиметричне шифрування, або шифрування з відкритим ключем	1) Вирішена проблема розподілу ключів між користувачами, так як кожен користувач може згенерувати свою пару ключів сам, а відкриті ключі користувачів можуть вільно публікуватися. 1) Зникає квадратична залежність числа ключів від числа користувачів (2N та N(N-1)/2).	1) Повільніше ніж симетричне шифрування, оскільки при шифруванні і розшифрування використовуються досить ресурсомісткі операції. 2) Необхідність захисту відкритих ключів від підміни. 3) Немає математичних доказів незворотності використовуваних функцій.

1. **Об’єкт проектування** – захист текстової інформації.
2. **Предмет проектування** – програмний модуль шифрування даних.
3. **Мета дипломного проекту** – розробити програмний модуль шифрування даних, реалізований методом XOR з додатковою функцією генерації ключа.

Завдання проектування

1. Проаналізувати методи шифрування текстових даних.
2. Розробити структуру модуля шифрування даних.
3. Розробити простий та зручний інтерфейс модуля.
4. Реалізувати алгоритм шифрування XOR.
5. Протестувати роботу модуля шифрування даних.

Структурна схема проекту



Функціональна схема модуля шифрування даних

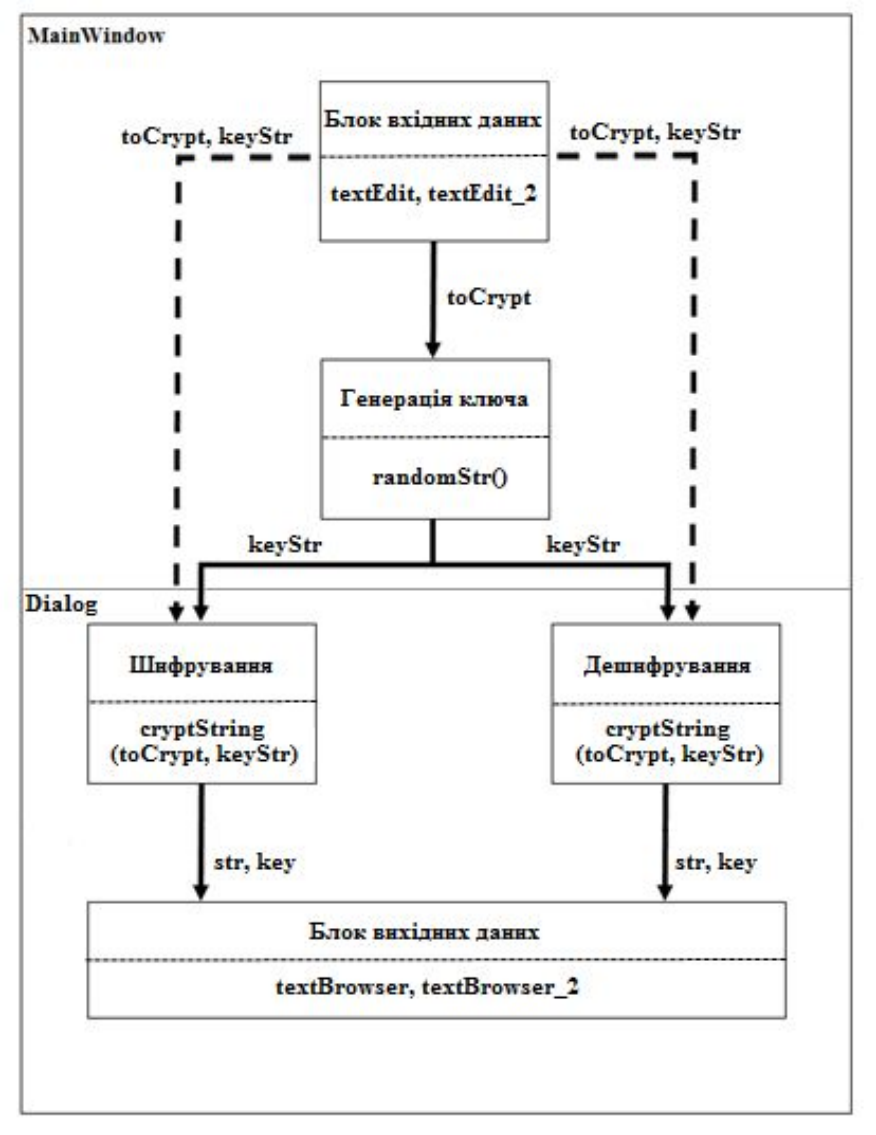


Схема алгоритму роботи циклу, який застосовує XOR

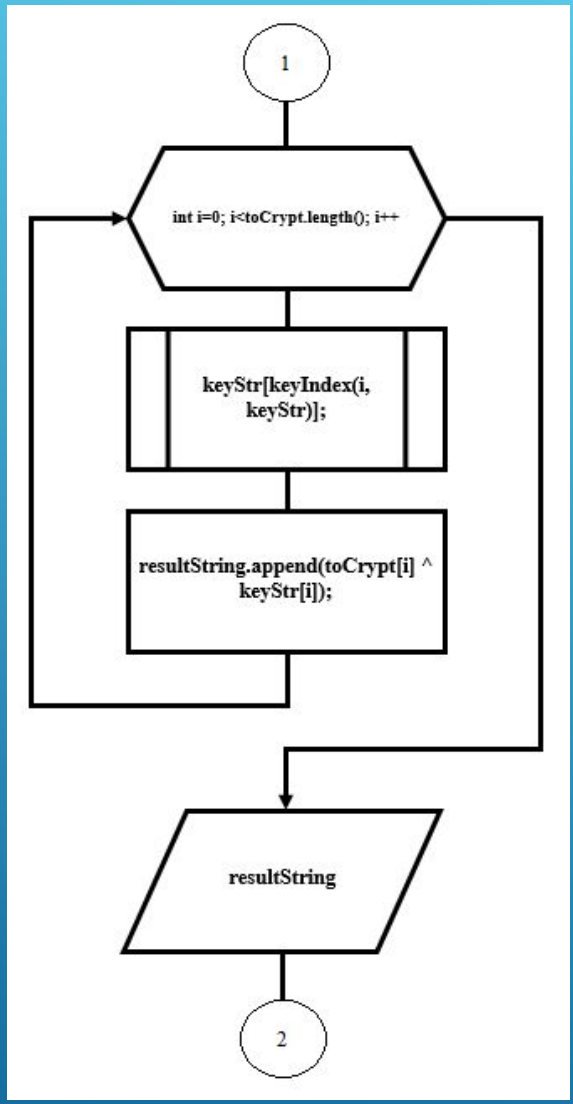


Схема алгоритму роботи функції keyIndex()

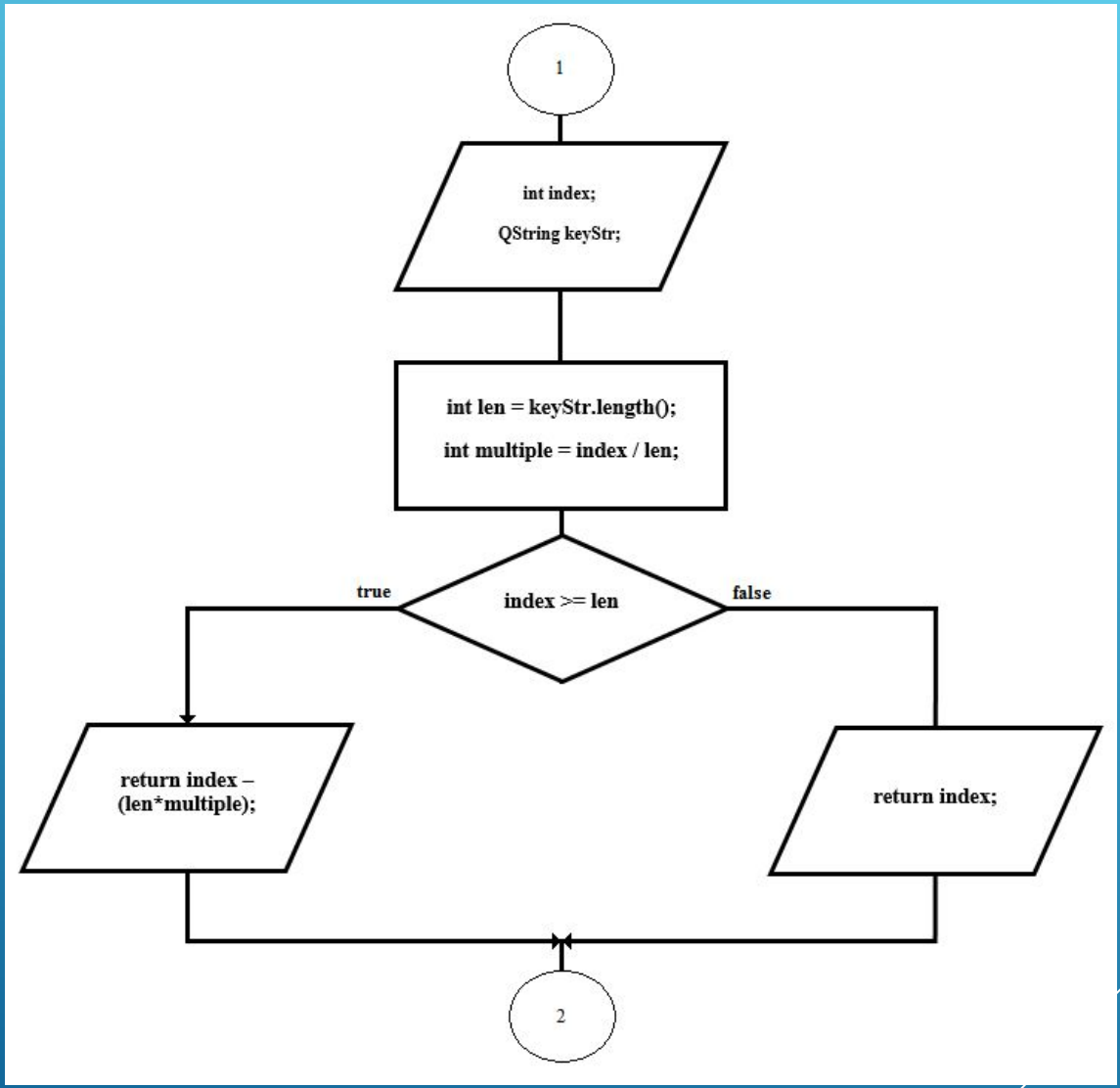
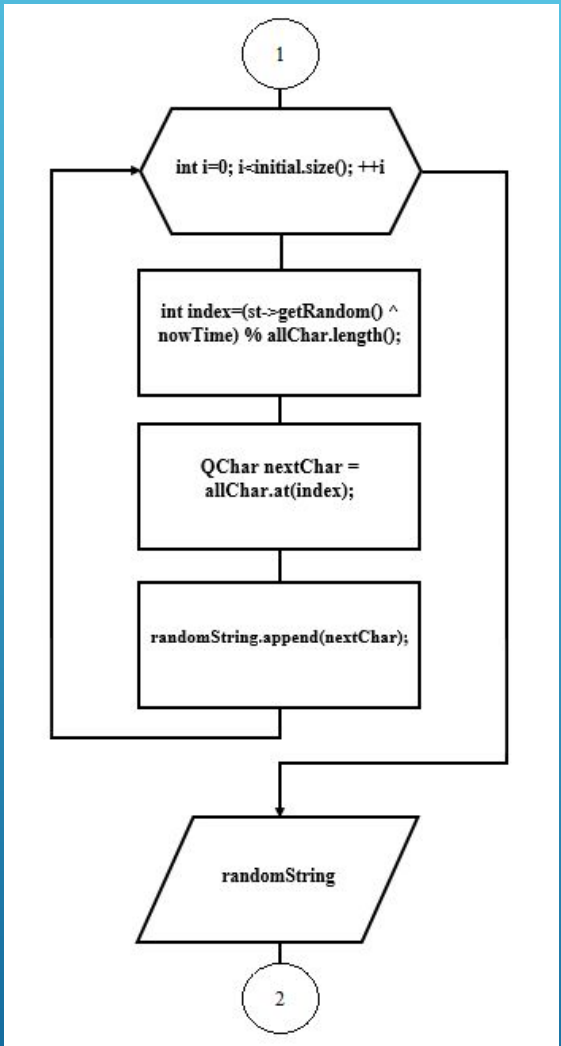
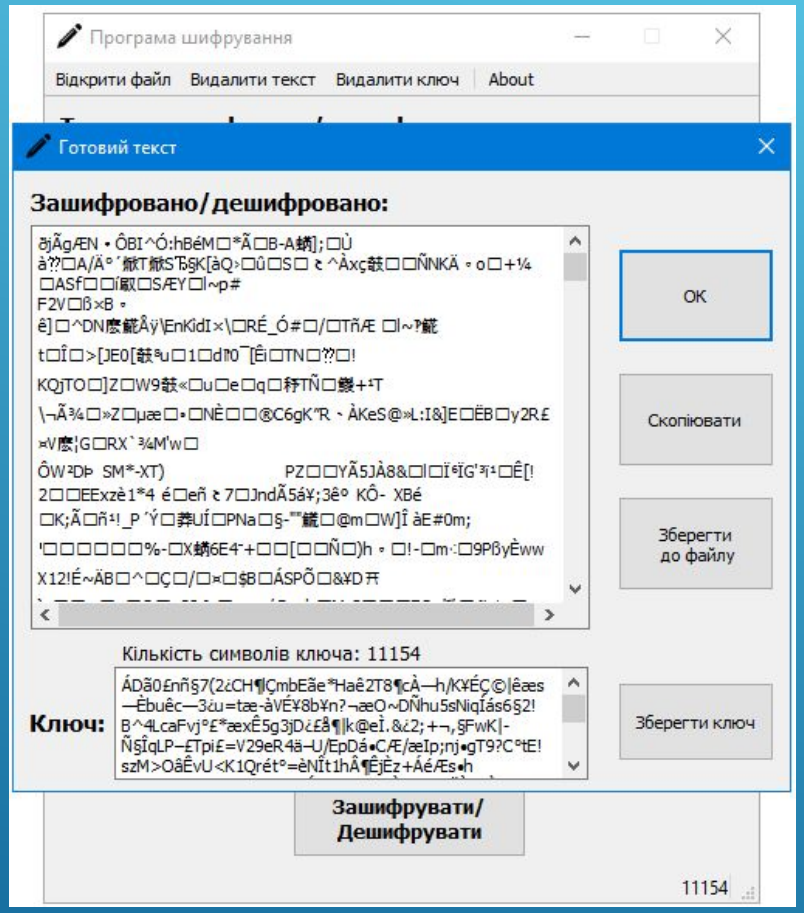


Схема алгоритму роботи циклу, який генерує випадовий рядок



Приклад роботи програмного модуля



Висновки

1. Проаналізовано існуючі програмні рішення з шифрування даних. Вирішено створити окремий програмний модуль, який має зручний інтерфейс та за допомогою якого можна швидко шифрувати будь-які текстові дані. Для даного програмного модуля обрано ОС Windows.
2. Проаналізовано сучасні алгоритми шифрування даних та проблеми, які пов'язані з цими алгоритмами. В даній предметній області є достатньо мало стійких алгоритмів, які досить тяжко взломати та які б швидко працювали. Симетричні алгоритми дуже надійні, але не такі зручні, як алгоритми з відкритим ключем. В свою чергу асиметричні алгоритми складні і це спричиняє падінню швидкодії програми. Враховуючі всі фактори різних алгоритмів шифрування, в даному проекті обрано потоковий метод шифрування XOR. Даний алгоритм має досить високий рівень швидкості та саме головне – просту реалізацію. Швидкість криптоаналізу даного алгоритму залежить від надійності заданого ключа. Тому вирішено, що програмний модуль буде мати в собі функцію генерації випадкового, складного для взлому, ключа.

Висновки

3. Розроблено структуру програмного модуля шифрування даних. Структура модуля складається з двох форм даних та кнопок обробки цих форм. На кожній з форм знаходяться по 2 браузери тексту, які відображують вхідні та вихідні дані відповідно. Форма вхідних даних може працювати з користувацькими текстовими файлами будь-якого розширення, що економить час користувача. Вихідна форма даних показує результат роботи алгоритму шифрування даних та забезпечує роботу з цими даними для подальшого їх використання.
4. Реалізовано алгоритм шифрування даних XOR, який надає можливість зашифрувати або дешифрувати будь-які текстові дані. Алгоритм шифрування представляє собою алгебрагічну функцію «виключне або». Вхідні дані переобразовуються в двійковий вид і до них застосовується бітова операція XOR. Це дозволяє швидко зашифровувати/дешифровувати будь-які текстові дані. Даний алгоритм є симетричним: дані зашифровуються та розшифровуються за допомогою одного і того ж відповідного їм ключа. Головний плюс даного алгоритму полягає у шифруванні та розшифруванні без втрати початкового тексту.

Висновки

5. Реалізовано функцію генерації випадкового ключа, яка створює ключ із символів таблиці ASCII. Для ключа вибираються випадкові символи з 128 можливих варіантів таблиці ASCII. Згенерований ключ майже неможливо зламати, а якщо і можливо, то це відніме велику кількість часу. Вже для ключа довжиною в 2 елементи кількість його варіантів дорівнює 16384. Для впровадження цієї функції було обрано генератор випадкових чисел XORShift, який базується на операції XOR. Сутність генератору в зміщенні бітів вхідного числа на n-позицій. Алгоритм створення випадкового ключа працює наступним чином. Число, яке отримується за допомогою генератору XORShift, та час роботи програмного модуля з його початку (в мілісекундах) застосовують між собою XOR. Отримане число «відкидує» свої біти доки не війде в діапазон 0-128. Число, отримане в цьому діапазоні являтиме собою індекс елемента, який буде взятий із таблиці ASCII. Випадково відібрані елементи сворюють собою ключ. Даний алгоритм генерації ключа працює достатньо швидко для великого об’єму вхідної інформації.

Висновки

6. Протестовано модуль шифрування даних. На вхід до модуля можна подавати інформацію великого об’єму і при цьому стабільність роботи від даного чинника не буде порушуватися. Модуль тестувався в два проходи на відповідність симетричності алгоритму: було взято певний початковий текст, далі відповідно до нього генерувався ключ, вихідний зашифрований текст зберігався і проходив через алгоритм з тим самим ключем. В ході тестування дефектів не виявлено.
7. Створено простий та зручний інтерфейс для програмного модуля шифрування даних. Початкова форма містить всі необхідні кнопки для роботи з початковими даними, а саме: «відкрити файл», «видалити текст», «видалити ключ». Впроваджено статус бар знизу форми, який описує повну дію певного елемента форми. В нижньому правому кутку відображується кількість символів початкового тексту. Вихідна форма даних вбирає в себе всі необхідні функції роботи з вихідними даними. На ній розташовано два поля для відображення зашифрованого, або дешифрованого тексту, та кнопки, які дозволяють з цим текстом працювати. Створено 4 кнопки: «скопіювати» – копіює текст в буфер обміну, «зберегти до файлу» – зберігає вихідний текст, «зберегти ключ» – зберігає до файлу ключ, «ОК» – повертає користувача до форми вхідних даних.

Висновки

8. Практичне значення результатів проектування полягає у наданні можливості шифрування текстових даних, що дозволяє підвищити загальний рівень захисту тексту від несанкціованого доступу на персональних компьютерів.

Дякую за увагу