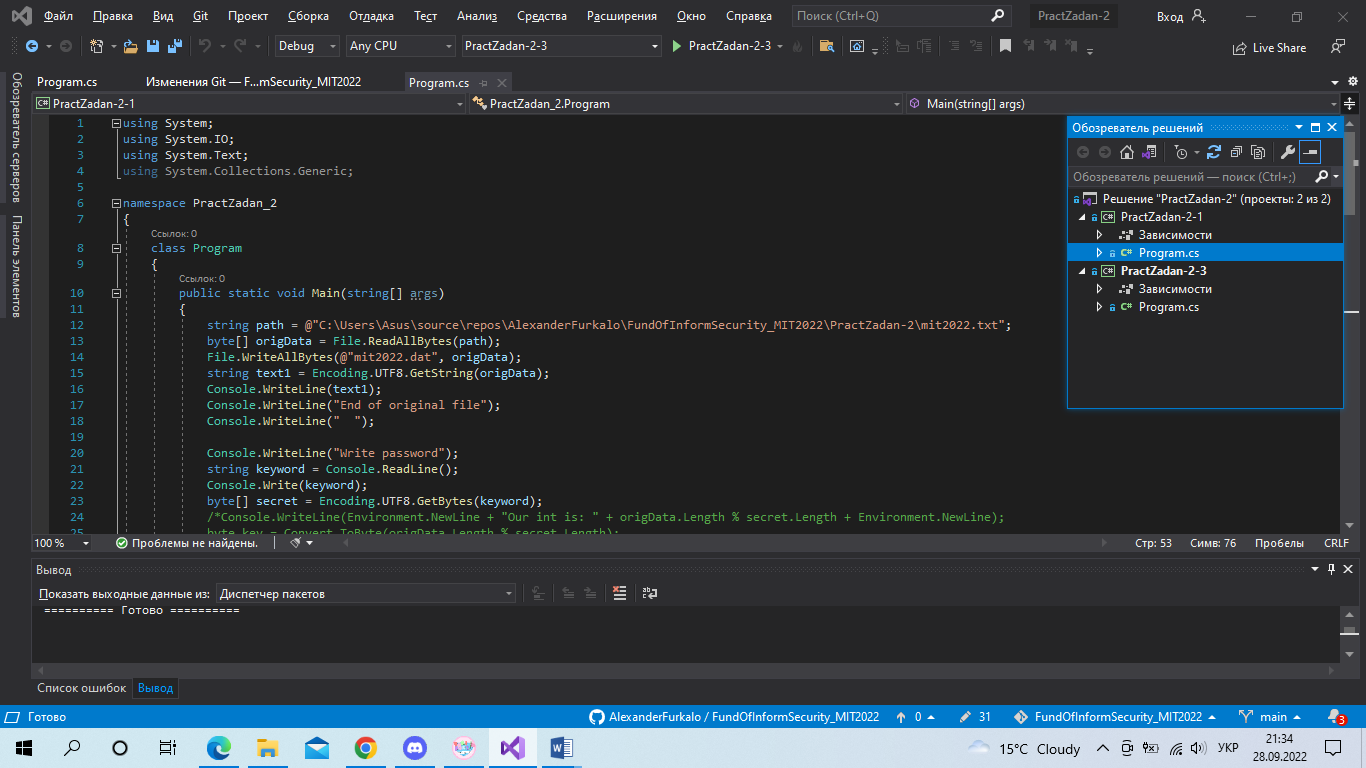
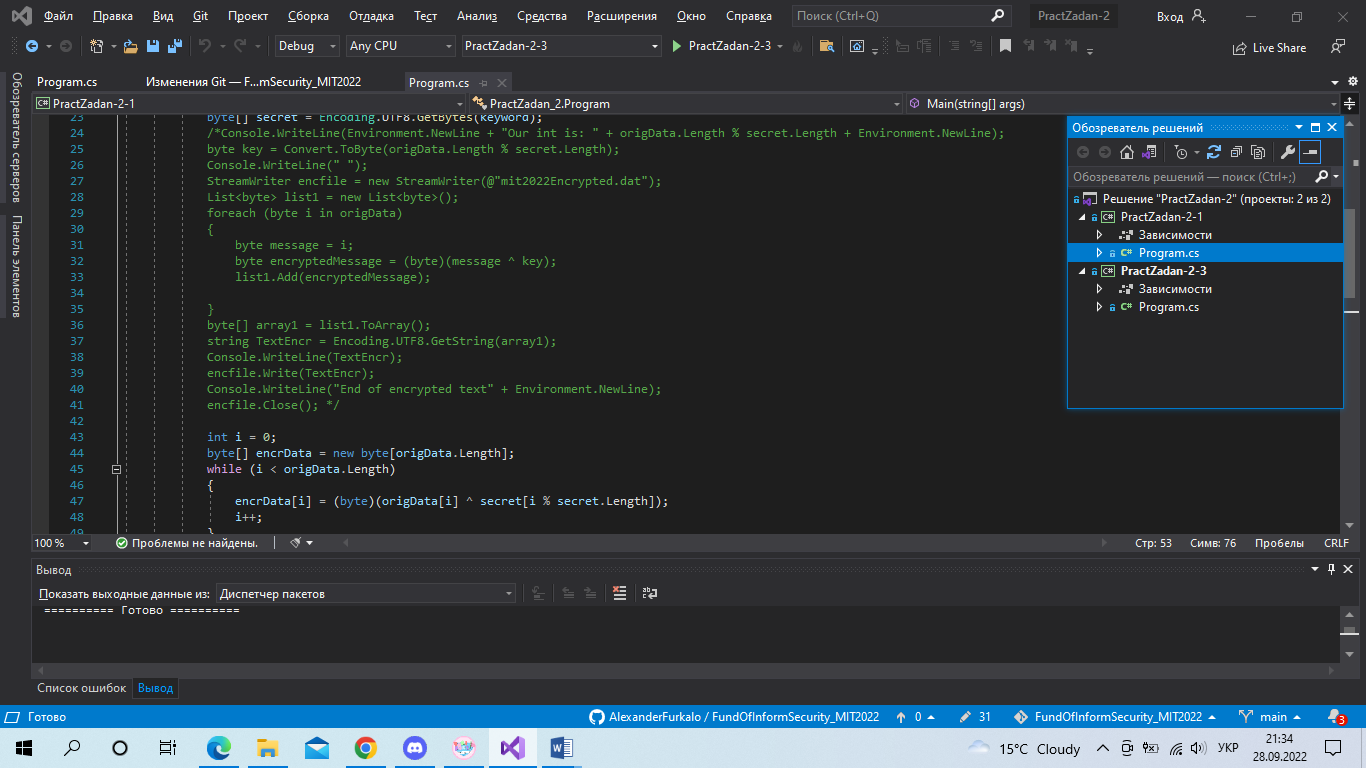
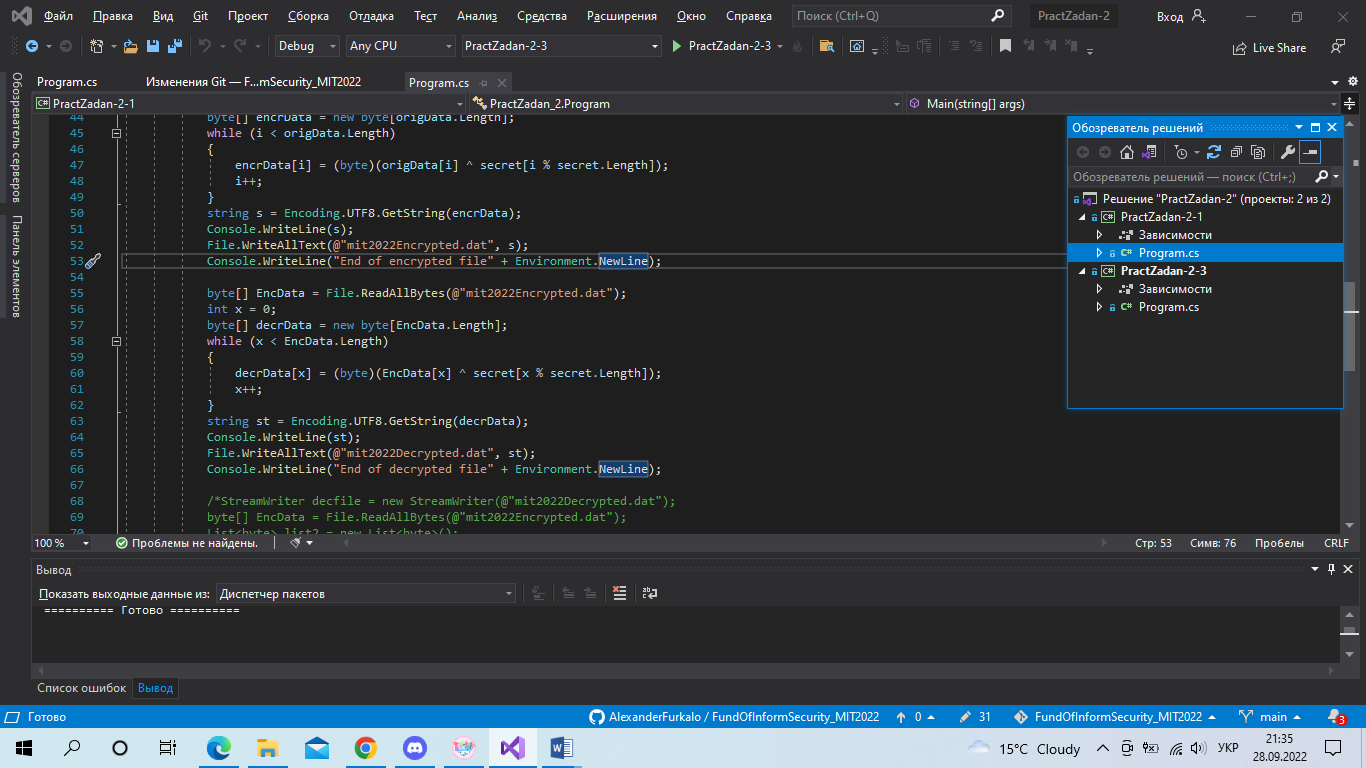
ПРАКТИЧНА РОБОТА №2  
З ОСНОВ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ  
СТУДЕНТА КИЇВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ІМЕНІ ТАРАСА ШЕВЧЕНКО  
ФАКУЛЬТЕТУ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ  
КАФЕДРИ МЕРЕЖЕВИХ ТА ІНТЕРНЕТ ТЕХНОЛОГІЙ  
ДРУГОГО КУРСУ, ДРУГОЇ ПІДГРУПИ  
ОЛЕКСАНДРА ОЛЕКСАНДРОВИЧА ФУРКАЛА  
ЗА ТЕМОЮ “РЕАЛІЗАЦІЯ ШИФРУ ВЕРНАМА”  
ЗВІТ  
28.09.2022

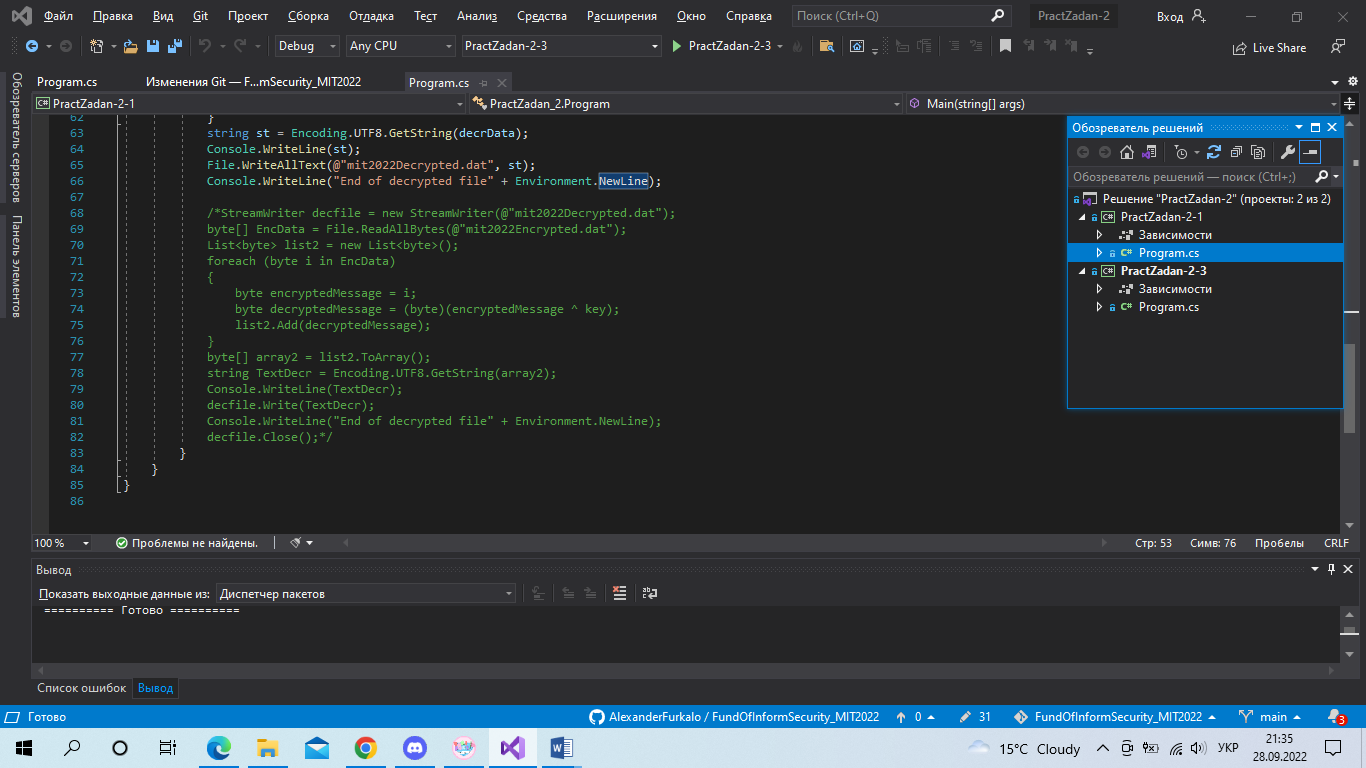
Завдання цієї практичної роботи були такі:  
- 1. Створити програму для зчитування файлу, шифрування його вмісту та запису зашифрованого фалу у ту ж саму папку, але з   
розширенням \*.dat.  
2. Модифікувати програму так, щоб можна було розшифровувати зашифрований файл.  
3. Створити програму для дешифрування (не плутати з розшифруванням) зашифрованого файлу методом «грубої сили» або методом перебору із використанням апріорної інформації про зашифрований текст, а саме: довжина пароля – 8 символів; у тексті присутня фраза "Mit21"  
4. Оформити Звіт

Перейдемо до першого з двох кодів:

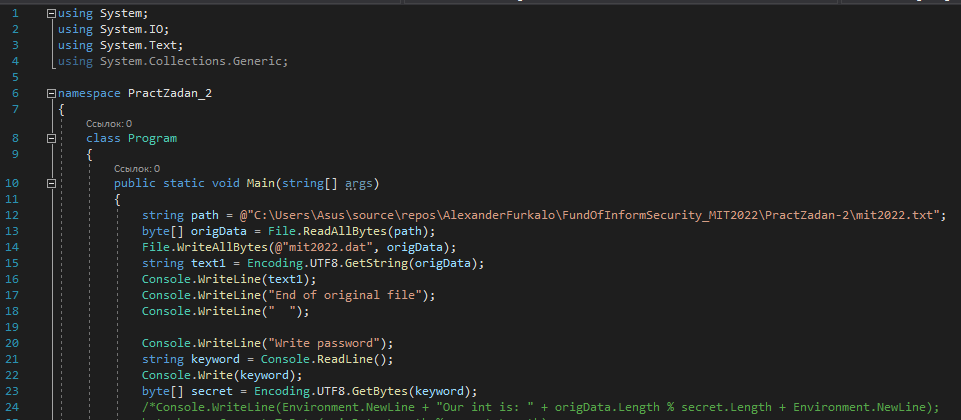


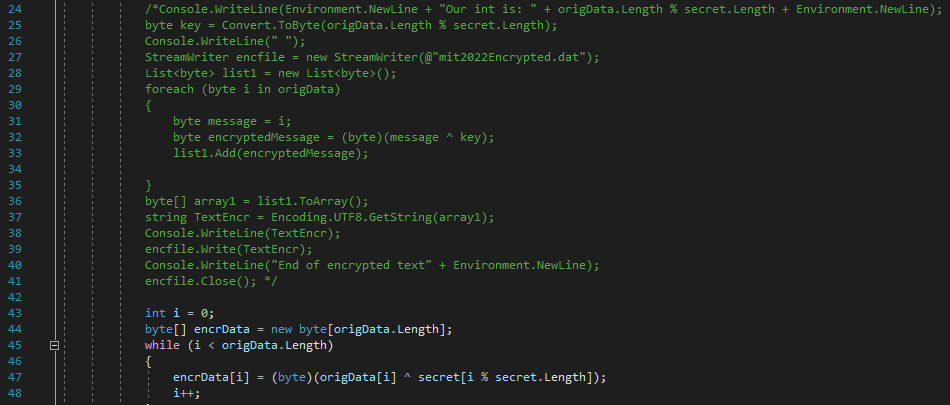


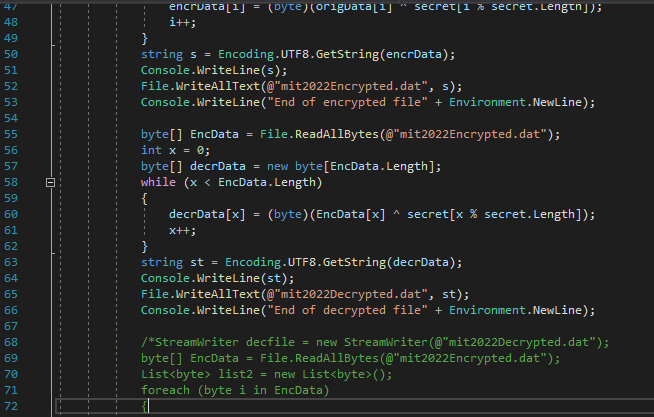


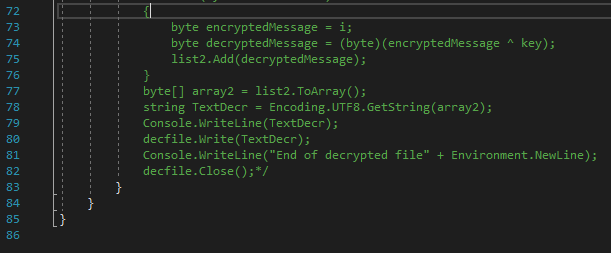


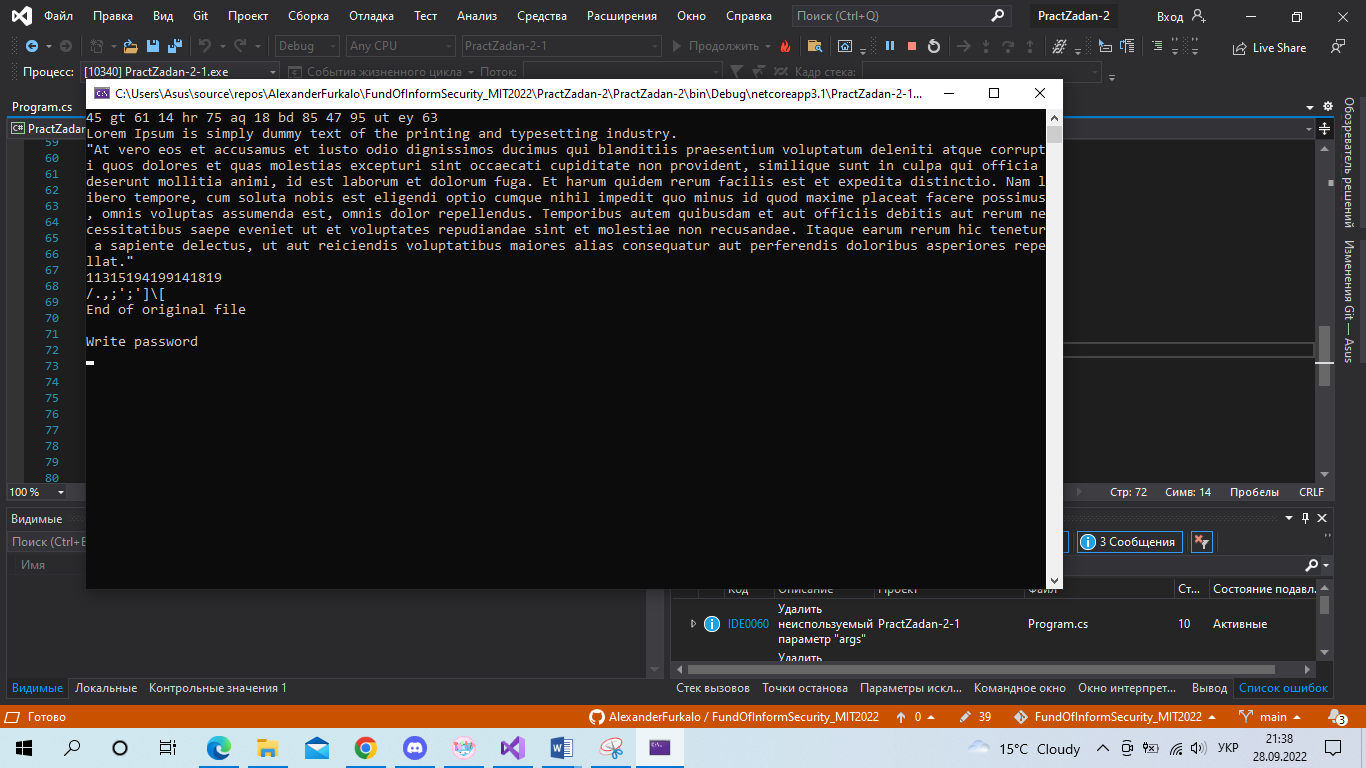
Трохи наблизивши:

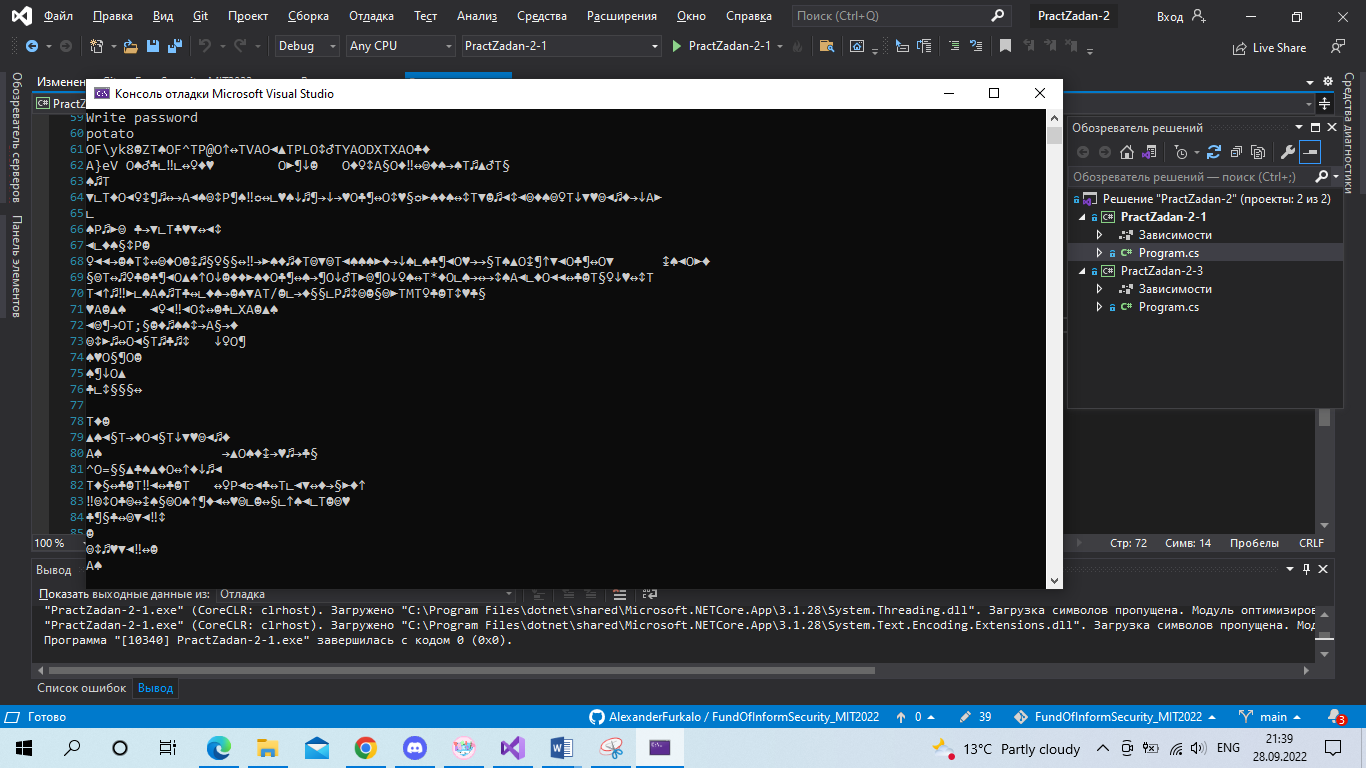


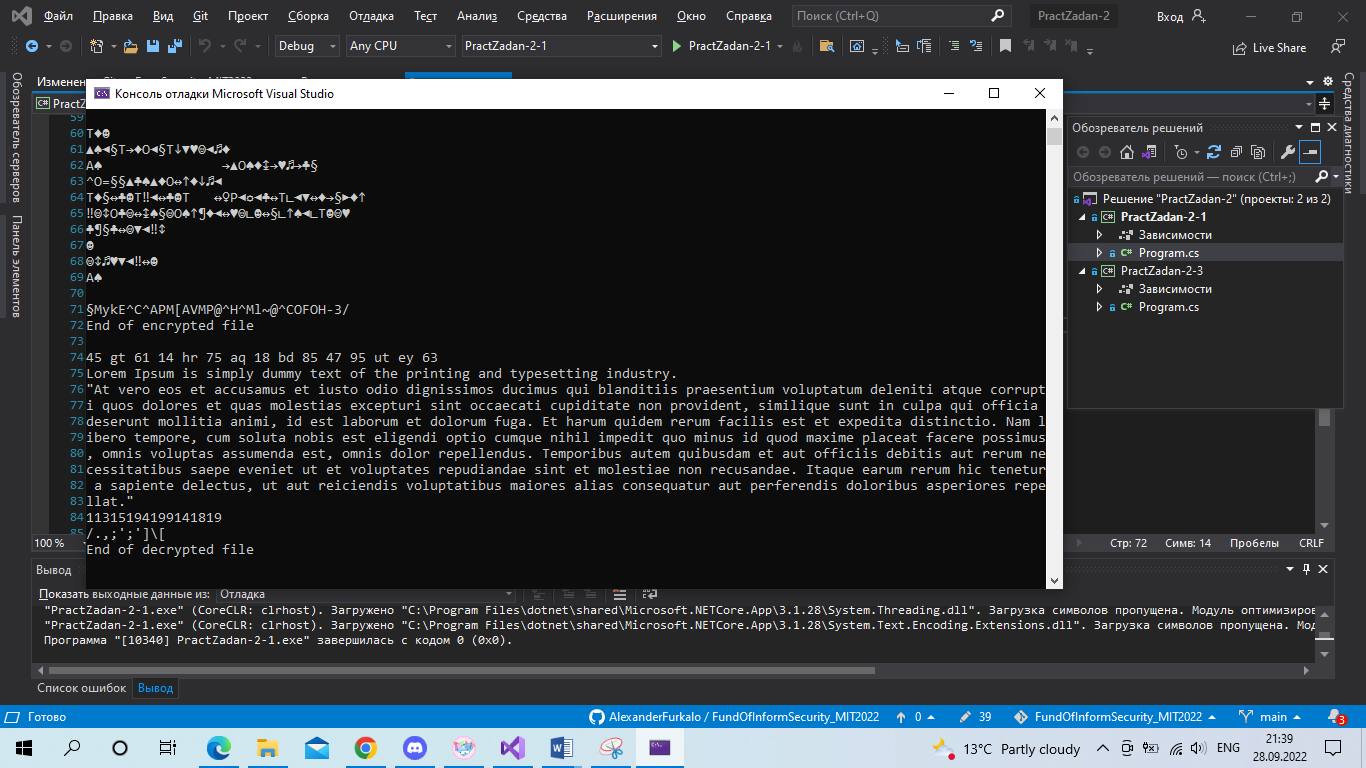


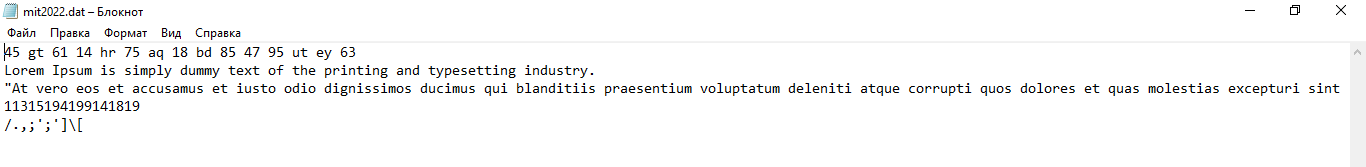
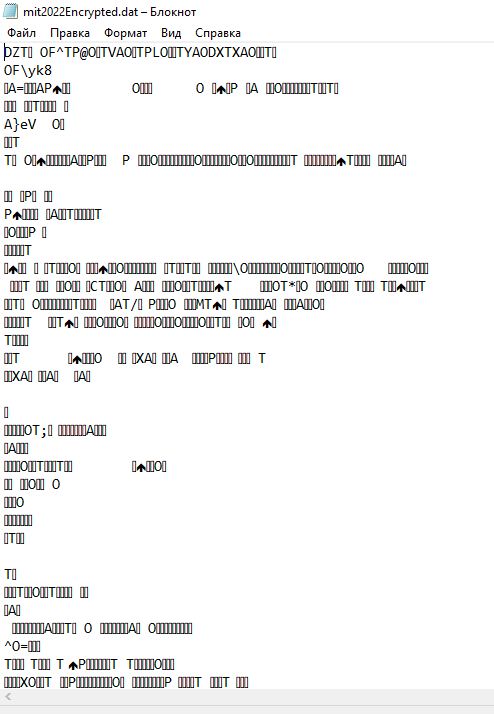
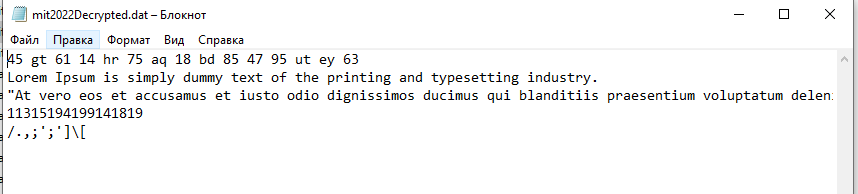




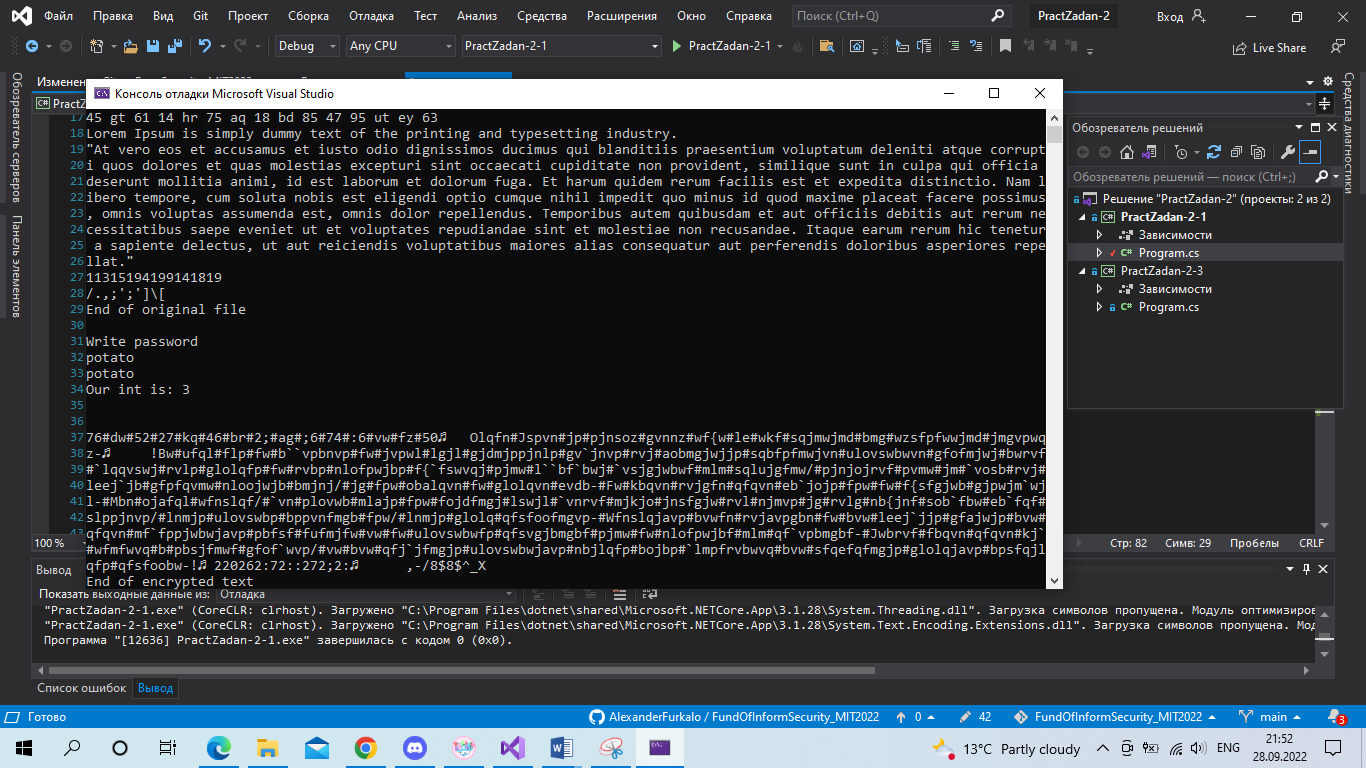
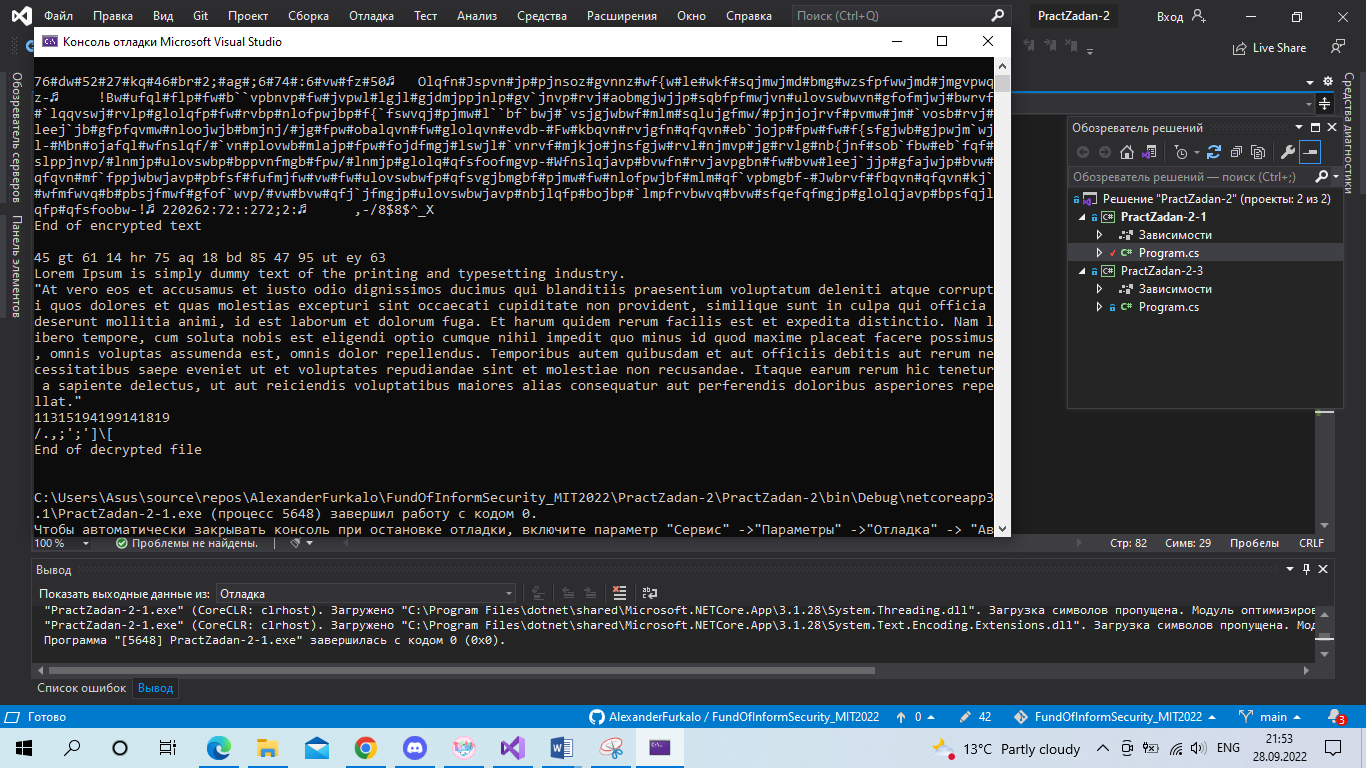
Результати роботи першої реалізації (основний варіант):  






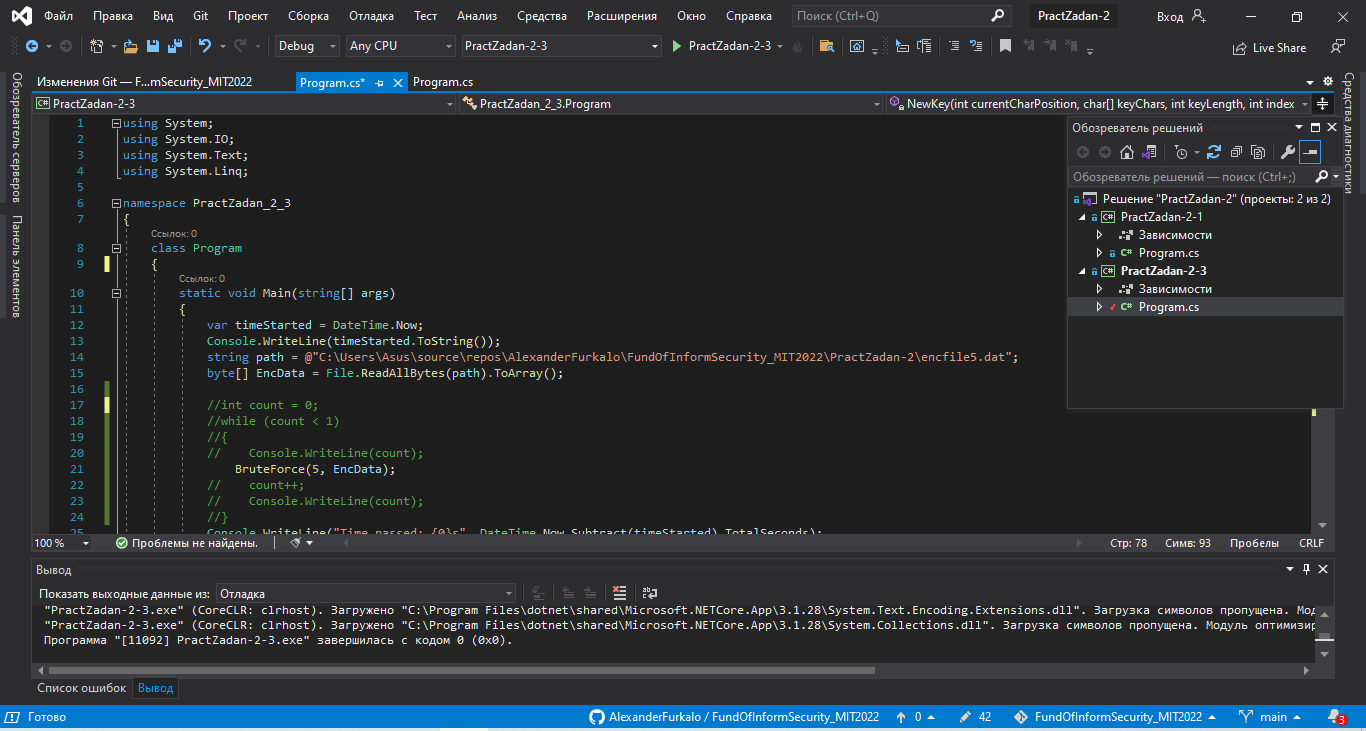
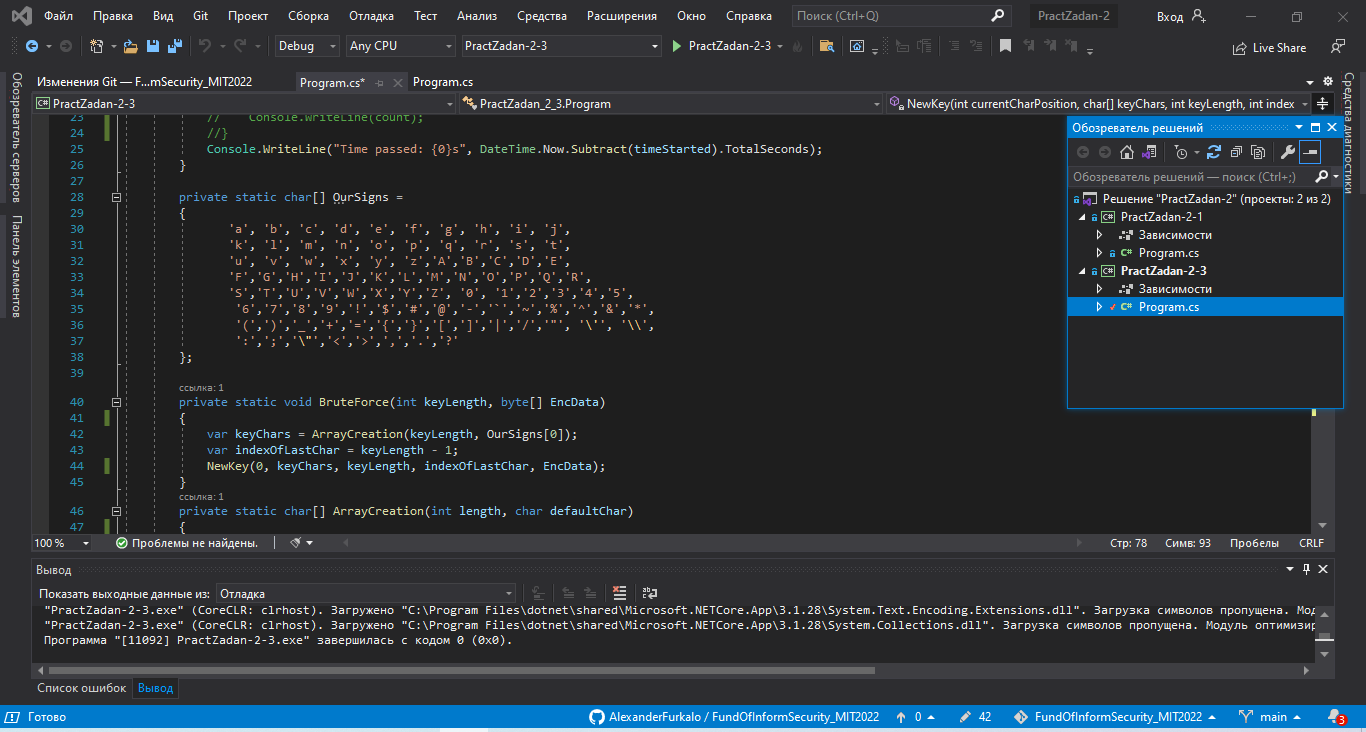
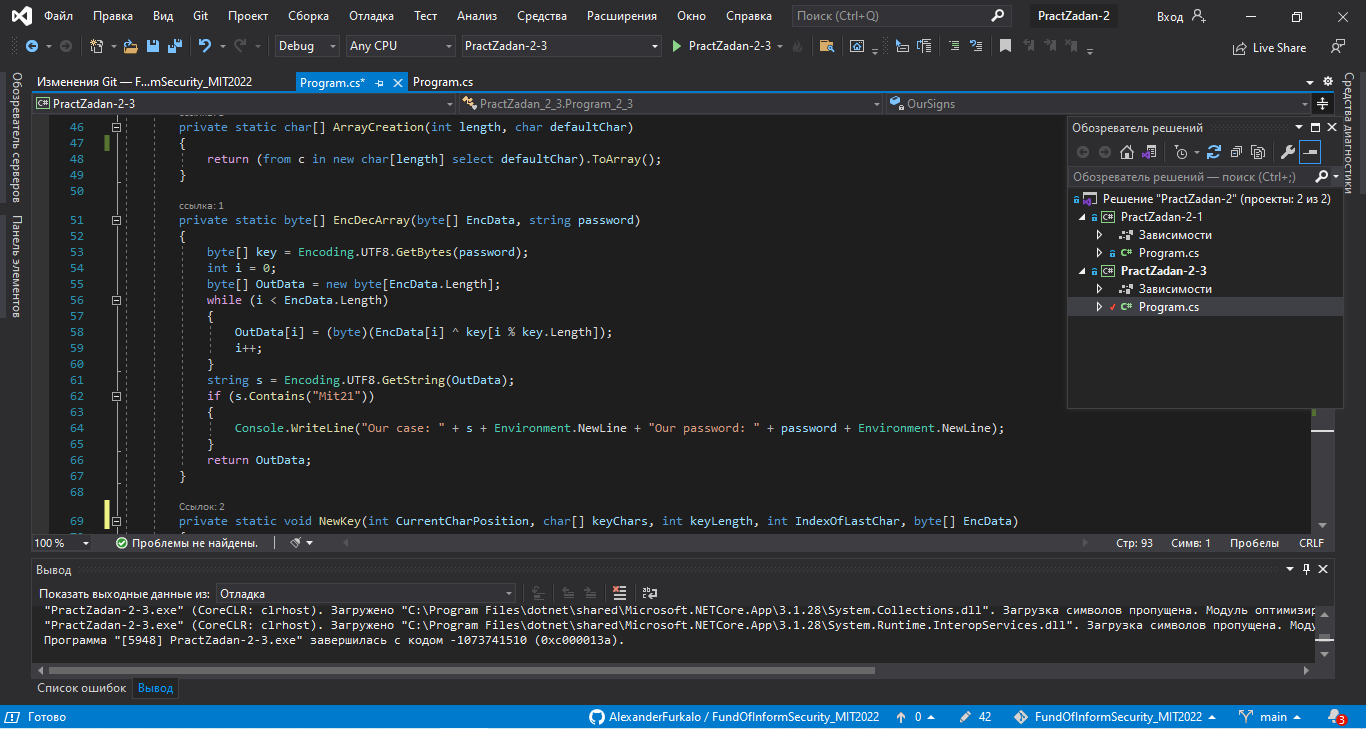
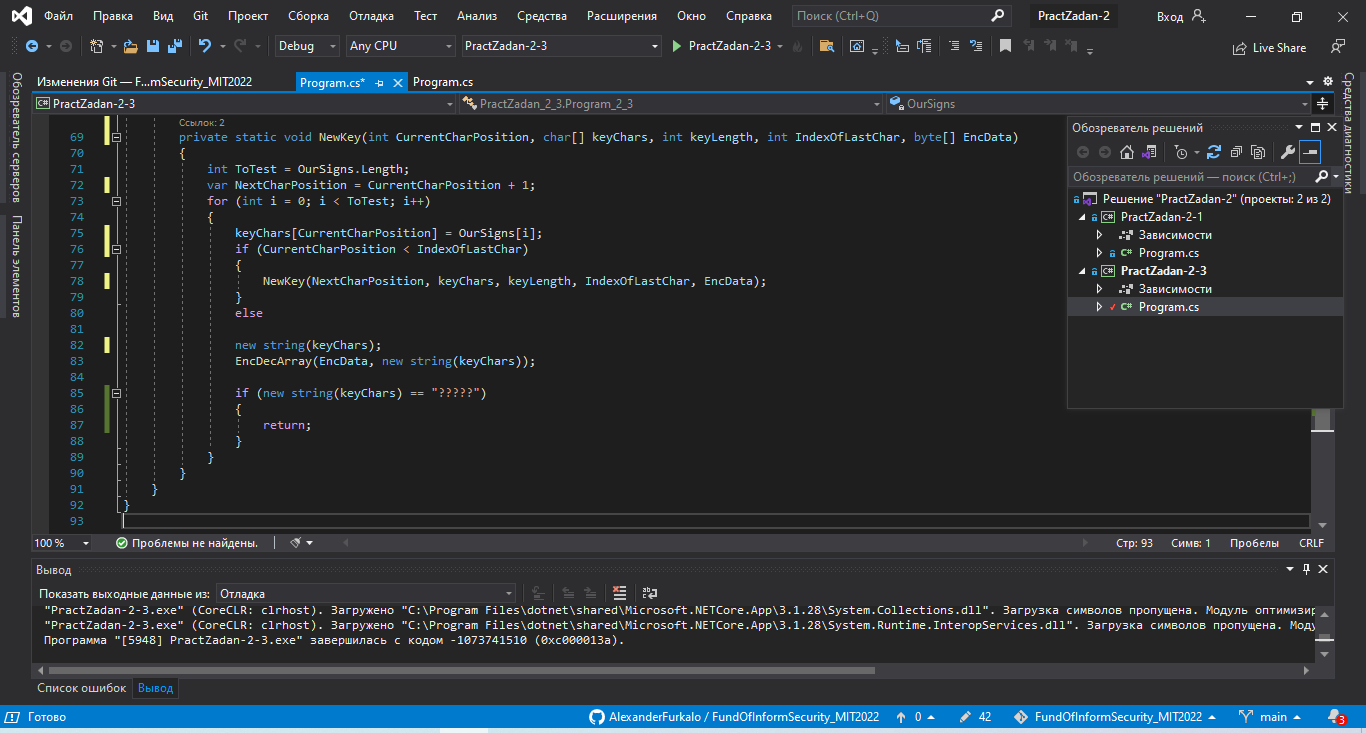
“mit2022Encrypted.dat” та “mit2022Decrypted.dat” зберігаються в папці PractZadan-2\bin\Debug\netcoreapp3.1, оскільки не були вказані конкретніші координати для їх збереження.  
mit2022.dat:  
  
mit2022Encrypted.dat:  
  
mit2022.Decrypted:  


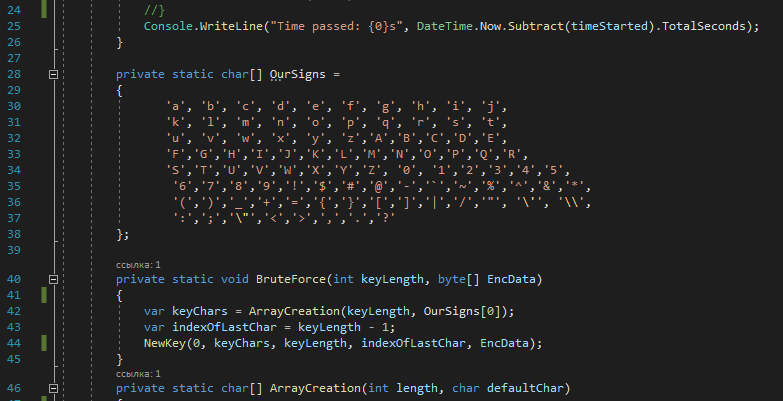
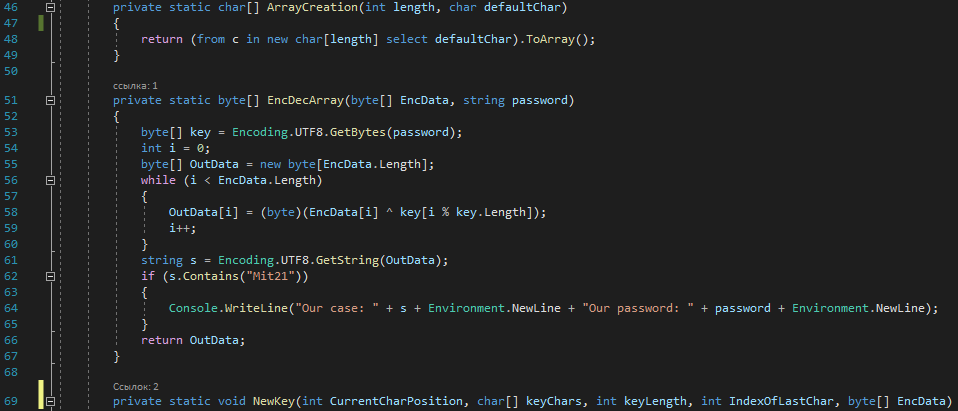
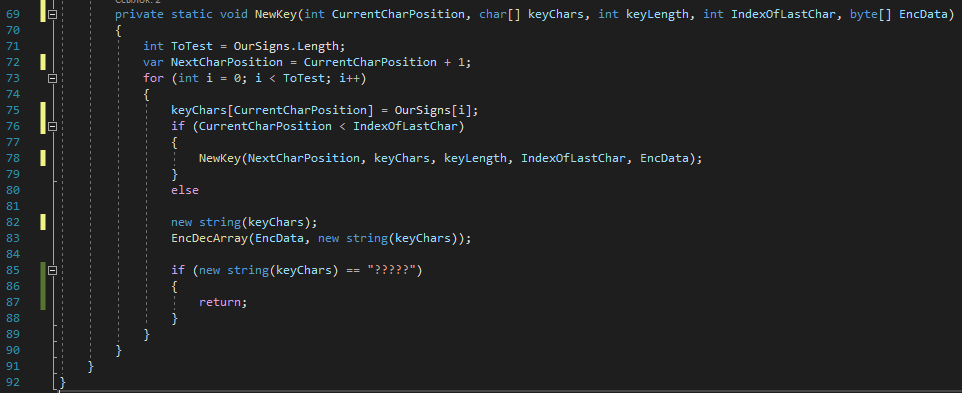
Для цього завдання також є інший варіант для шифрування, який на знімках коду позначений зеленим кольором

Обидві реалізації вдало шифрують файл, проте останній, як здається, має менше можливих варіантів – “key” буде десь у діапазоні від 0 до 127  
Основна команда – операція XOR. Вона записується за допомогою ^ - для шифрування повідомлення потрібно виконати її над текстом та секретним ключем, наприклад, ось так:  
- byte encryptedMessage = (byte)(message ^ key);  
- encrData[i] = (byte)(origData[i] ^ secret[i % secret.Length]);

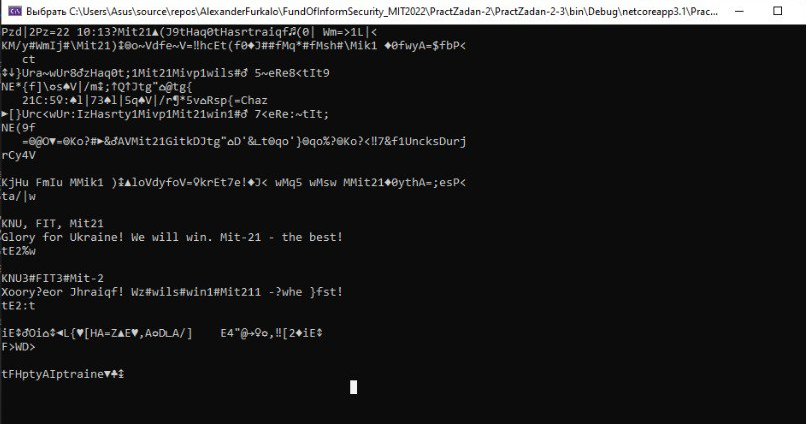
На базі цієї операції XOR, Вернам у 1917 році запатентував шифр, так і названий – “Шифр Вернама” або ж “одноразовий шифр-блокнот”.   
Для шифрування рядка (бітів) необхідно мати ключ, що складається з тієї ж самої кількості двійкових знаків, що й повідомлення. Коли цих умов дотримано, кожен біт рядка, що підлягає шифруванню, додається за модулем 2 (операція XOR) до відповідного біта ключа й отримуємо шифрограму.  
Ключ **повинен** мати три властивості: бути справедливо випадковим, має збігатися за розміром із заданим відкритим текстом та одноразовим.  
Цей метод, однак, мав критичний недолік - хоч саме повідомлення і було добре зашифровано, спочатку потрібно було передати ключ.   
Якщо ключ передавати у відкритому вигляді, його можна буде перехопити, і тоді сила шифру не матиме значення. Якщо передавати його за захищеними каналами зв'язку, тоді таким же чином можна було б передати і саме повідомлення - шифр, знову, не має сенсу.  
Якщо ключ не одноразовий, можна використовувати метод брутфорсингу, щоб знайти пароль та розшифрувати повідомлення – про це друге завдання:

Наблизим:   
  
  
  


З особливостей:  
- main веде лічильник часу, щоб наприкінці виконання вивести підсумковий витрачений час. Для того, щоб перейти від “aaaaa” до “bbbbb”, потрібно приблизно 104 секунди. На перебір усіх перерахованих на поточний момент варіацій символів у п'ятизначній комбінації пішло 15534 секунд або 258 хвилин

https://cdn.discordapp.com/attachments/903399019995340890/1024285381916639272/unknown.png  
- BruteForce має два параметри – EncData (ці дані потрібні далі в програмі) та довжину пароля, яка вказана як 5. Якщо кількість символів у паролі невідома, цей фрагмент можна переробити, поставивши лічильник count і зробивши число цифр невідомим (замість п'яти), збільшуючи це значення на 1 кожного разу, коли програма повертається.  
- OurSigns - символи, які використовуватимуться для розрахунку пароля. Усього елементів десь 95, що дає 7737809375 можливих комбінацій.  
- У самому BruteForce додається ще кілька значень: keyChars через ArrayCreation та IndexOfLastChar (обидва використовують у своїх розрахунках поточну довжину пароля, яка в нашому випадку завжди дорівнює п'яти)  
- NewKey використовується для генерації паролів - починаючи з “aaaaa”, вона через рекурсію змінює це значення, починаючи з останнього символу, після чого переходить на передостанній символ, і так доти, доки не дійде до останнього можливого варіанту у вигляді “?????”.  
- Кожне з цих значень переходить в метод EncDecArray - у ньому отримане значення використовується як ключ операції XOR, потім результат переводиться в string, і якщо там є потрібний нам рядок – виводиться на екран.  
За допомогою такої програми можна зламати будь-який багатозначний код, що складається із зазначених символів. Тому й були придумані раніше перелічені необхідні властивості ключа шифра Вернама  
Приклад виконання (з розшифрованим рядком, пароль для якого був tE2%w)



Висновки

Під час цієї лабораторної роботи ми навчилися створювати  
програми для зчитування файлу, шифрування та розшифрування його вмісту та запису зашифрованих файлів з розширенням \*.dat. Ми також навчилися створювати програми для дешифрування (не плутати з розшифруванням) зашифрованих файлів методом «грубої сили» або методом перебору із використанням апріорної інформації про зашифрований текст. Ми дізналися про Шифр ​​Вернама, про плюси цього шифрування і деякі мінуси, які виявляються на практиці, а також переконалися, наскільки просто обчислити п'ятизначний пароль.