Міністерство освіти і науки

Київський національний університет імені Тараса Григоровича Шевченка

Кафедра Мережевих та Інтернет технологій

**Звіт**

з практичної роботи **№3-4**

з дисципліни: “Основи інформаційної безпеки”

**«Хеш-функції та перевірка цілісності інформації»**

Виконав:

Студент групи МІТ-21

Горбун О. М.

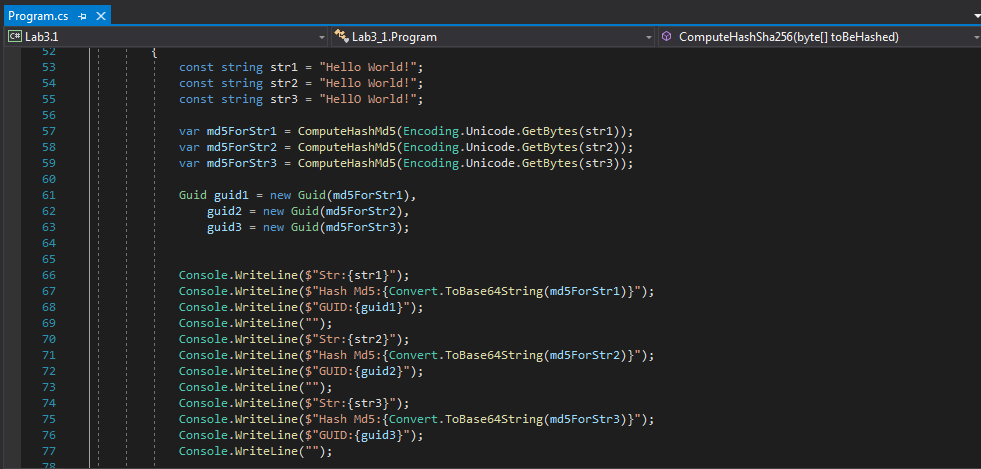
**Результати виконання завдань роботи:**

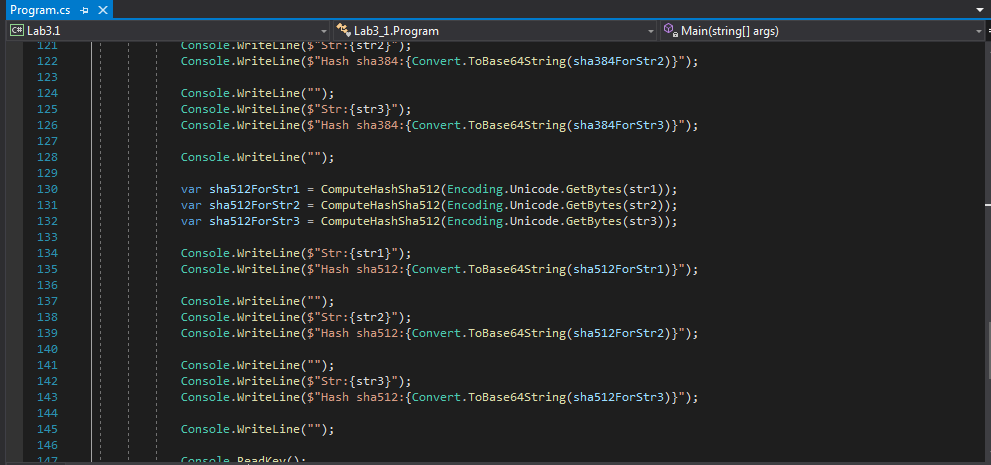
За завданнями практичної роботи було створено програми, які можна переглянути за посиланням: <https://github.com/OleksandrHorbun/Basics_Cybersecurity.git> (Посилання на відкритий репозиторій у веб-сервісі GitHub) завдання у вітці (Branch) Homework\_3\_4

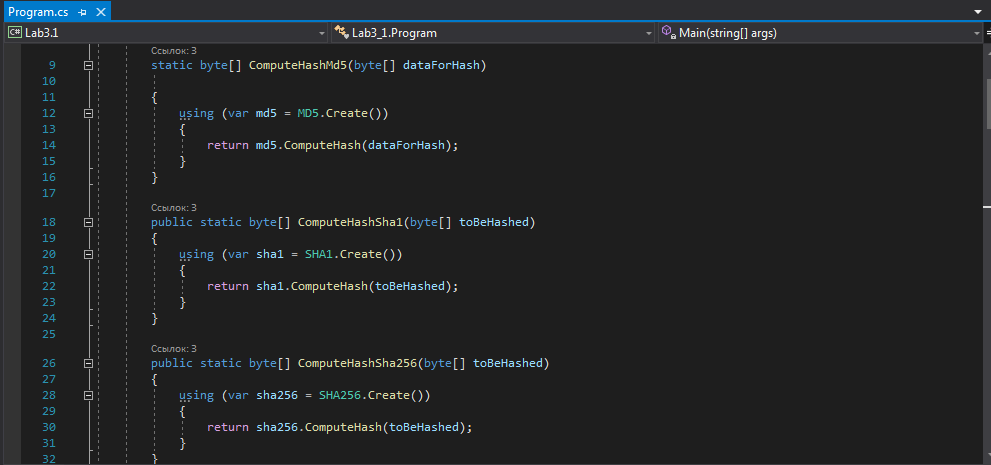
**I частина (перше завдання)**

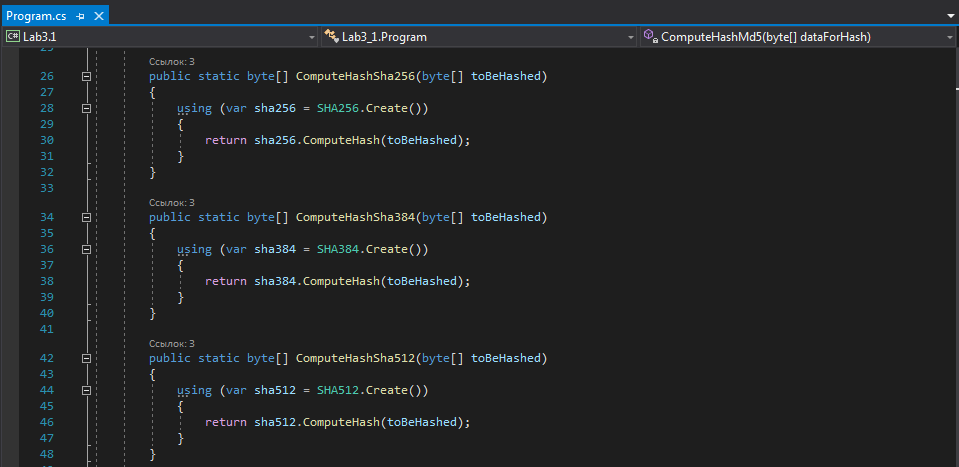
Як можемо побачити, у програмі наявні три рядка, один з яких відрізняється, кожний з цих рядків почергово передається у функцію *ComputeHashMd5* (хешування за алгоритмом MD5 - Message Digest Algorithm). У функції виконується хешування за MD5, після повертається в основну програму, де зберігається як екземпляр структури GUID (Globally Unique IDentifier). Хеш та GUID кожного рядка виводиться користувачу на екран. Після цього три вищевказані рядки передаються у функцію *ComputeHashSha1* (хешування за алгоритмом SHA1 - Secure Hashing Algorithm), де вираховується хеш за SHA1. Очевидно що це застаріла технологія, але за завданням її необхідно розглянути на практиці. Отже, вищевказана функція *ComputeHashSha1* вираховує хеш та повертає отриманий результат в Main, де, в свою чергу, результат одразу записується у змінну й хеш виводиться користувачу на екран. Увесь вищеописаний процес повторюється з тими ж рядками (виступають у ролі деяких «повідомлень», які треба захешувати). В програмі окрім хешування за MD5 і SHA1 розглянуто хешування за допомогою алгоритмів SHA256, SHA384, SHA512, для яких (по порядку) було створено функції *ComputeHashSha256*, *ComputeHashSha384*, *ComputeHashSha512*. Останні два є більш сучасними й використовуються для більш-менш безпечного хешування даних, адже в них найбільший розмір хешу. Процес використання функцій для хешування за цими алгоритмами абсолютно ідентичний до перших двох описаних. Звісно усі результати виводяться на екран.

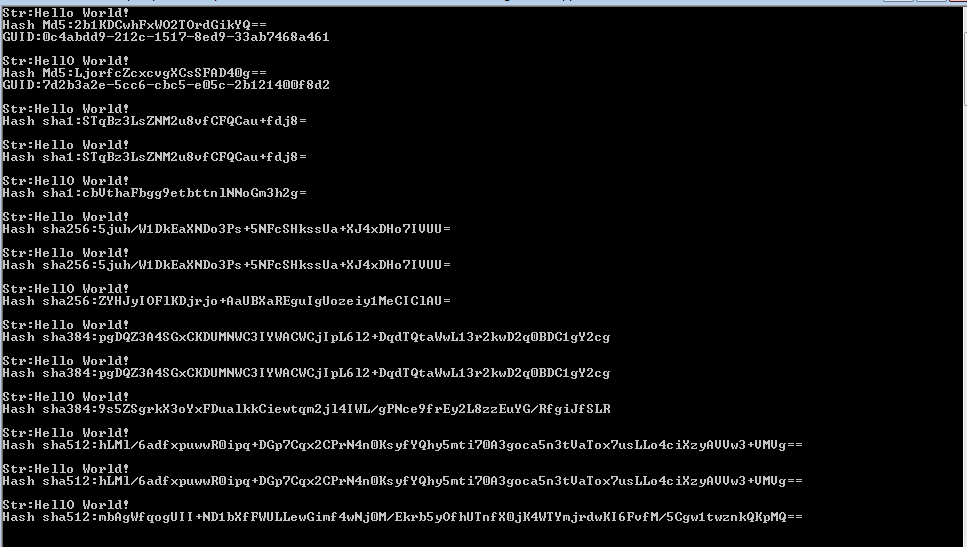
Все описане вище можна побачити на знімках екрану:







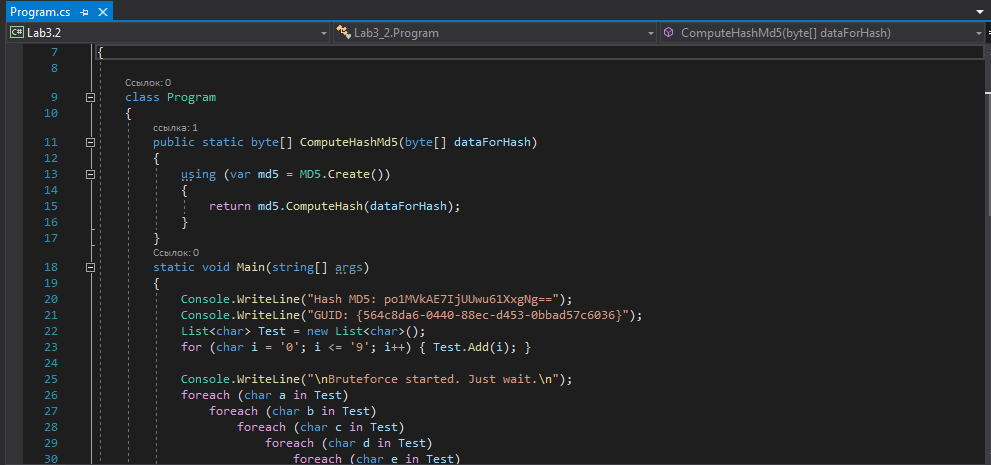


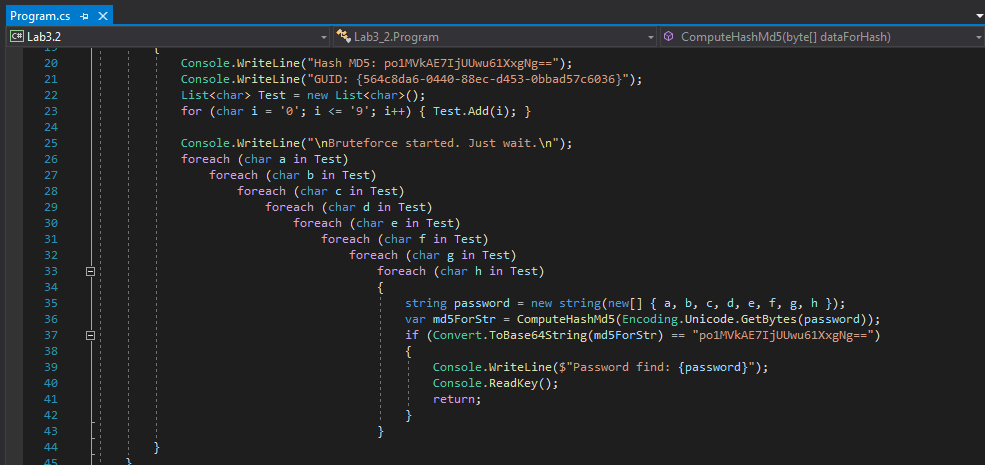


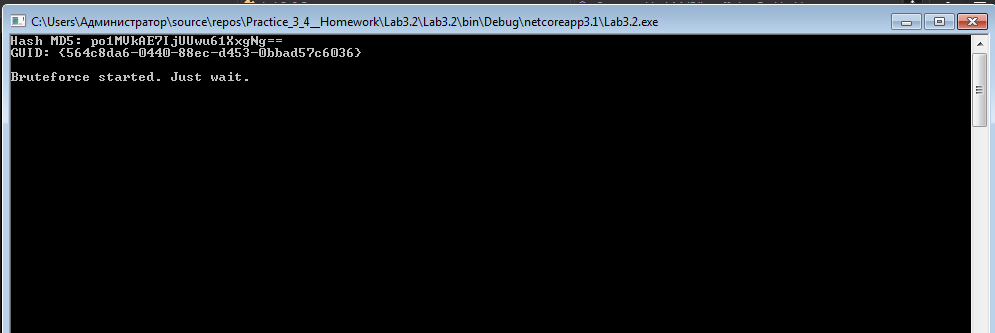
**IІ частина (друге завдання)**

Друга програма робить навпаки: з одержаних хешу MD5 та GUID підбирає пароль користувача. Можна робити підбір як за самим хешом, так і за GUID. Очевидно що підбір за хешом MD5 є більш раціональним, адже доведеться підбирати набагато менше можливих символів, що прискорить отримання бажаного результату. Загалом відбувається підбір за допомогою методу «грубої сили», тобто кожний символ підбирається за допомогою «каскаду» циклів. Після кожного отримання можливого хешу додатково проводиться перевірка: рядок передається у функцію *ComputeHashMd5* з минулої програми, тобто підібраний рядок перетворюється у хеш MD5, після чого відбувається перевірка і, якщо необхідний хеш знайдено, цикли перериваються, а результат (підібраний рядок - пароль) виводиться на екран. Виконання програми може зайняти деякий час, але, як можна побачити, пароль з хешу можна відновити методом підбору. Більше того розмір хешу в MD5 невеликий (16 байтів) через що такий спосіб зберігання паролів є ненадійним. Такий метод хешування недоцільно застосовувати у кібербезпеці.

Все описане вище можна побачити на знімках екрану:



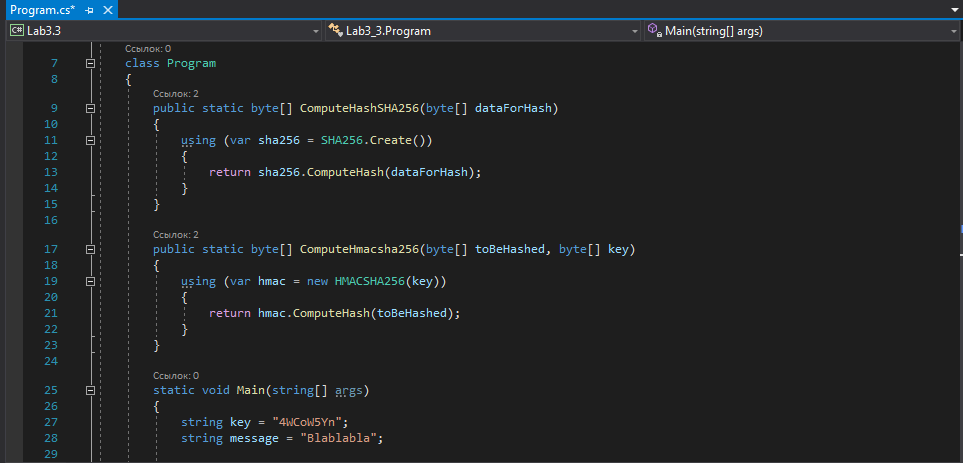


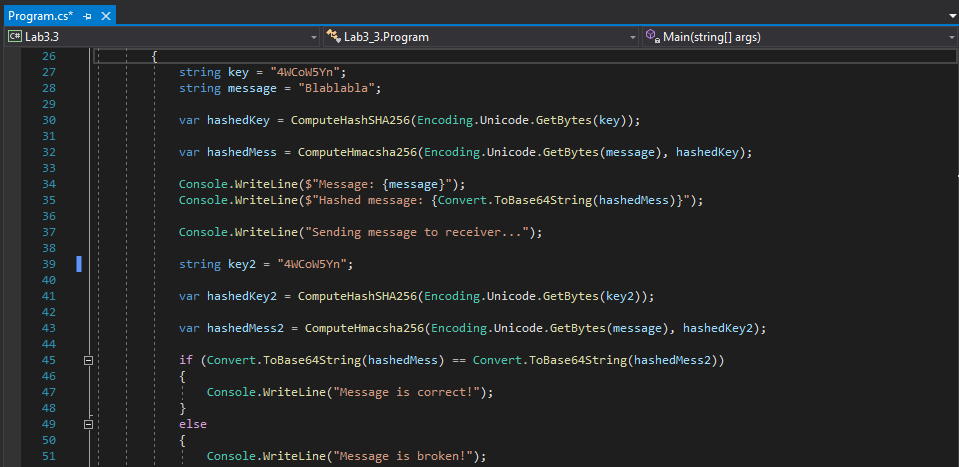


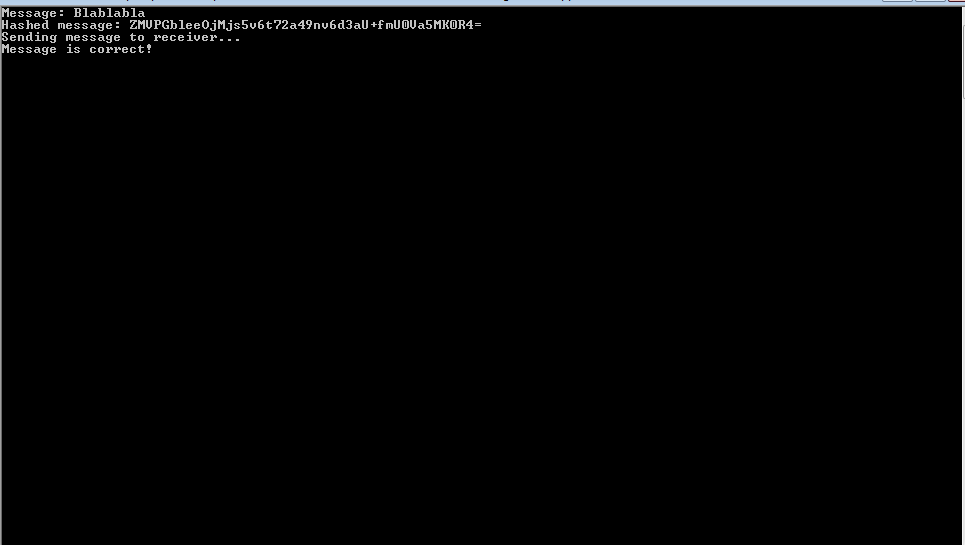
**IІІ частина (третє завдання)**

У третій програмі наявне деяке повідомлення та ключ для обчислення автентифікованого хешування. Спочатку для звичайного хешування за алгоритмом SHA256 використовується функція *ComputeHashSha256*, після цього отриманий звичайний хеш разом з ключем передається до функції *ComputeHmacsha256* (автентифіковане хешування за SHA256), яка є ідентичною до *ComputeHashSha256* за виключенням того, що при розрахунці де-факто хешу з хешу ще використовується ключ. Після цього проводиться перевірка автентифікації: тобто, так само розраховується автентифікований хеш для повідомлення, але вже за другим ключем. Якщо другий ключ співпадає з першим, то при перевірці вийде однаковий хеш, про що буде виведено повідомлення на екран. В іншому випадку (другий ключ неправильний) при перевірці хеш не співпаде про що буде виведено повідомлення на екран.

Все описане вище можна побачити на знімках екрану:









**Висновок:**

Виконуючи цю роботу я навчився обчислювати хеш за всіма відомими основними алгоритмами хешування, отримувати з хешу повідомлення методом підбору, обчислювати автентифікований хеш та виконувати його перевірку, тобто спів ставляти кожний новий хеш з вже обчисленим, що є основою для автентифікації та логування користувачів.