**Міністерство освіти і науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра обчислювальної техніки**

**Лабораторна робота №4**

з дисципліни «Системне програмування»

на тему «Макровизначення і макроси в MASM»

Виконав: Перевірив:

Студент 2 курсу ФІОТ доц. Павлов В. Г.

групи ІМ-22

Довженко Антон Андрійович

номер в списку групи: 9

Київ 2024

**Мета роботи:**

Вивчити технології створення і вживання макросів. Дослідження результатів роботи макрогенератору MASM.

Порядок виконання роботи

1. Вивчити правила використання макросів в програмах.

2. Переробити програму, підготовлену в лабораторній роботі 3 так, щоб виведення кожного виду інформації (дата народження, номер залікової книжки і т. п.) виконувалося у вигляді багатократного звернення до макросу (макрос 1), в якому у віконному інтерфейсі здійснювалося б виведення тексту, який передається в макрос як параметр.

3. Оформити також у вигляді окремих макросів фрагменти програми, де здійснюється шифрування введеного рядка символів (макрос 2), а також порівняння її з хеш-кодом оригіналу пароля, що зберігається у програмі (макрос 3). Макровизначення для всіх макросів розмістити в тому ж файлі, що і програма. У всіх макросах обов'язково використовувати звичайні і приховані коментарі, а в останньому макросі також використовувати механізм оголошення локальних міток.

4. Виконати компіляцію і компоновку файлу програми.

5. Перевірити роботу програми шляхом введення як правильного, так і невірного паролів.

6. Отримати розширений лістинг програми за допомогою опції /Fl компілятора ML.

7. Провести дослідження отриманого лістингу: визначити, яким чином компілятор виконав трансляцію кожній з команд макросів і помітити їх в тексті розширеного лістингу кольоровими олівцями або кольоровими фломастерами.

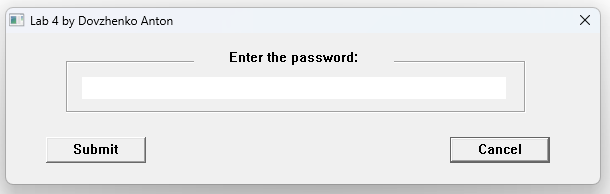
8. Повторити дослідження, які виконані в п. 4 – 7 для другої версії програми, де всі перераховані в п. 2 і 3 макровизначення для макросів повинні розташовуватися в окремому файлі.

9. Порівняти тексти розширених лістингів для обох варіантів розміщення макровизначень, знайти схожість і відмінності. Відобразити їх в звіті по лабораторній роботі.

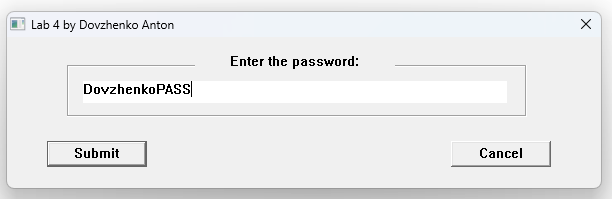
10. Зробити висновки по лабораторній роботі

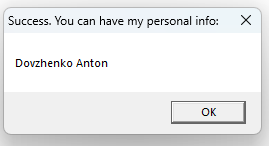
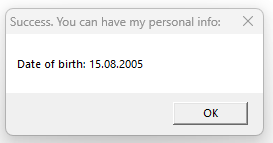
**5. Перевірити роботу програми шляхом введення як правильного, так і невірного паролів.**

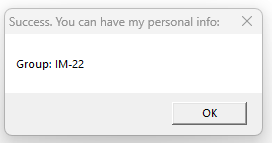
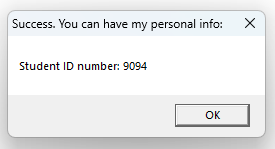
Запуск прогами:



Введення правильного паролю:

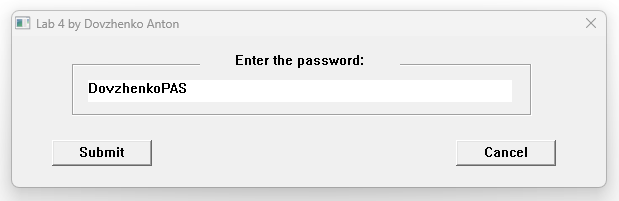


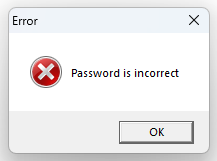
 

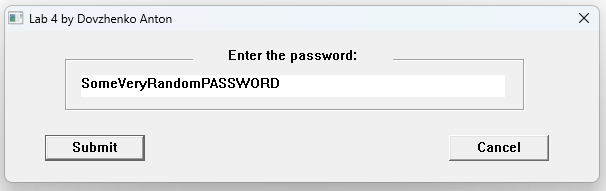
 

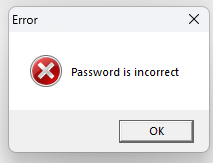
Тобто тепер персональна інформація виконується у вигляді багаторазового звернення до **DovzhenkoDisplayMessageMc** (макросу 1) і виводить інформацію по черзі, кожен раз в новому рядку, а не як в попередній лабораторній роботі, де виводилася вся інформація в одному вікні.

Введення неправильного паролю:









Вікно про помилку у зв’язку з неправильним паролем також створюється за допомогою **DovzhenkoDisplayMessageMc** (макросу 1).

Отже, у нас є три макроси:

**DovzhenkoDisplayMessageMc** (макрос 1) – цей макрос призначений для виводу повідомлень у вікні діалогу. Використовується в програмі і для виводу персональної інформації у випадку введення правильного паролю, і для виводу помилки, якщо пароль неправильний.

**DovzhenkoEncryptMc –** цей макрос відповідає за шифрування.

**DovzhenkoCompareHashMc –** цей макрос призначений для порівняння паролів.

**Шифрування паролю**

Пароль – DovzhenkoPASS  
Ключ - sECRETKEY1479

Переведення

|  |  |
| --- | --- |
| D – 010001002 – 4416 | s – 011100112 – 7316 |
| o – 011011112 - 6F16 | E – 010001012 – 4516 |
| v - 011101102 - 7616 | C – 010000112 – 4316 |
| z – 011110102 - 7A16 | R – 010100102 – 5216 |
| h - 011010002 - 6816 | E – 010001012 - 4516 |
| e – 011001012 - 6516 | T – 010101002 – 5416 |
| n - 011011102 - 6E16 | K – 010010112 – 4B16 |
| k – 011010112 - 6B16 | E – 010001012- 4516 |
| o – 011011112 - 6F16 | Y – 010110012 – 5916 |
| P – 010100002 - 5016 | 1 – 001100012 – 2116 |
| A – 010000012 - 4116 | 4 – 001101002 – 3416 |
| S – 010100112 - 5316 | 7 – 001101112 – 3716 |
| S – 010100112 - 5316 | 9 – 001110012 - 3916 |

Шифрування

D XOR s = 010001002  XOR 010100112 = 001101112 = 3716 = 7o XOR E = 011011112  XOR 010001012 = 001010102 = 2A16 = \*  
v XOR C = 011101102  XOR 010000112 = 001101012 = 3516 = 5  
z XOR R = 011110102  XOR 010100102 = 001010002 = 2816 = (  
h XOR E = 011010002  XOR 010001012 = 001011012 = 2D16 = -  
e XOR T = 011001012  XOR 010101002 = 001100012 = 3116 = 1  
n XOR K = 011011102  XOR 010010112 = 001001012 = 2516 = %  
k XOR E = 011010112  XOR 010001012 = 001011102 = 2E16 = .  
o XOR Y = 011011112  XOR 010110012 = 001101102 = 3616 = 6  
P XOR 1 = 010100002  XOR 001100012 = 011000012 = 6116 = a  
A XOR 4 = 010000012  XOR 001101002 = 011101012 = 7516 = u  
S XOR 7 = 010100112  XOR 001101112 = 011001002 = 6416 = d  
S XOR 9 = 010100112  XOR 001110012 = 011010102 = 6A16 = j

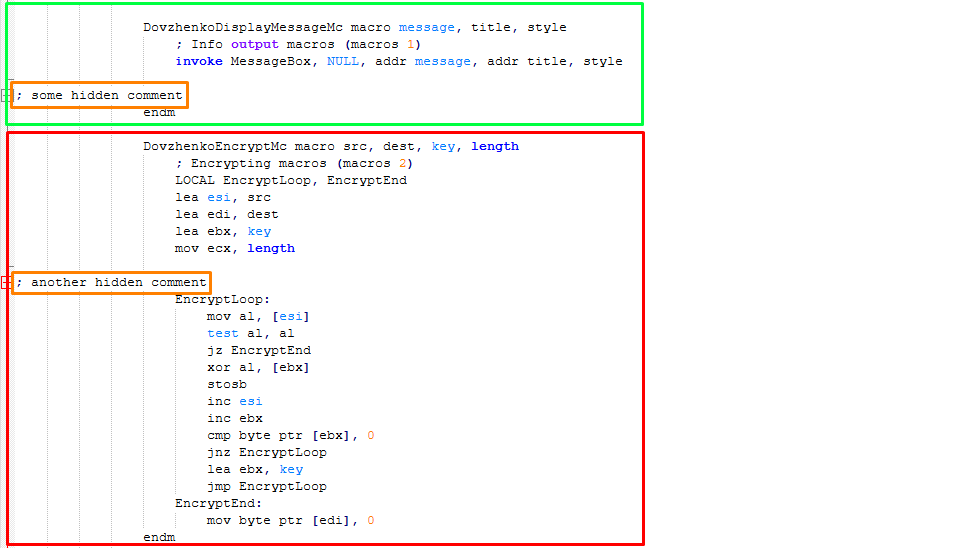
Отже, наш зашифрований пароль - 7\*5(-1%.6audj

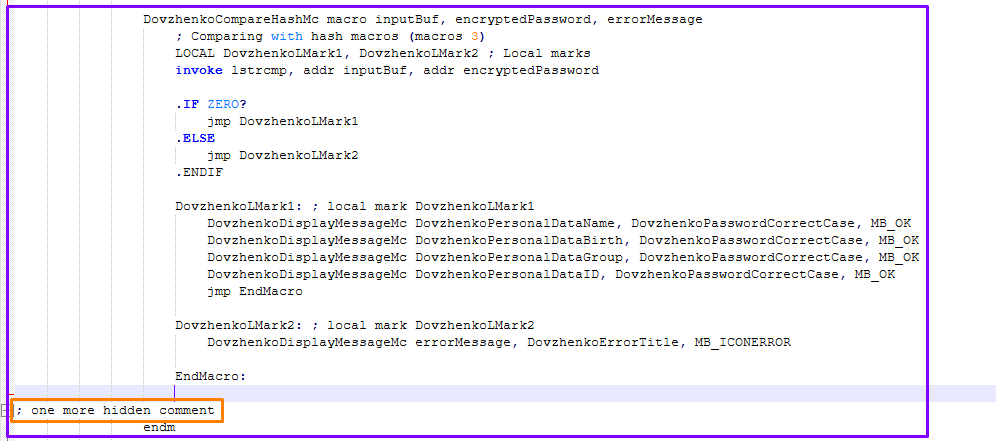
**Дослідження лістингу**

Спочатку програма, де макроси в одному файлі з рештою коду:

**Макровизначення**

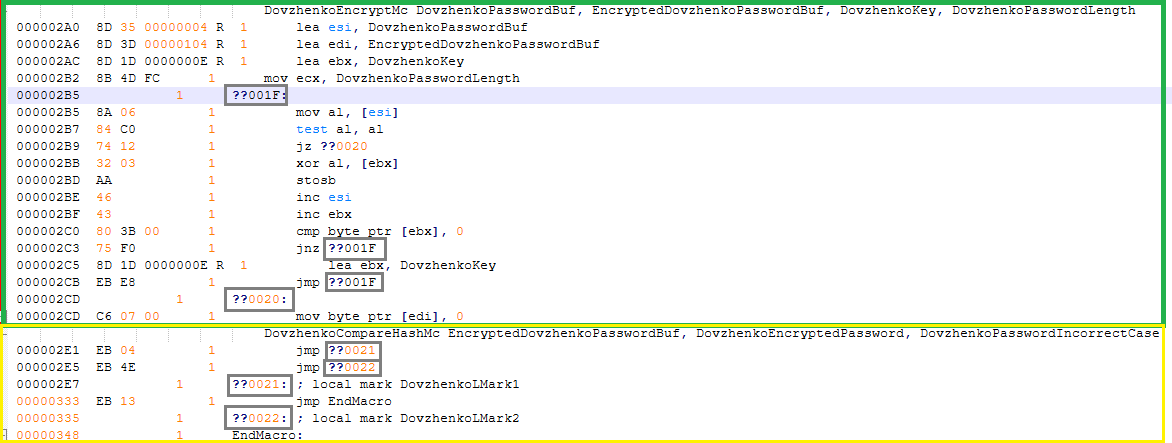
Після блоку .data було визначено макровизначення, які нижче виділив





Виділено зеленим на скріншоті макровизначення виведення вікна з повідомленнями, червоним кольором макровизначення шифрування пароля, та фіолетовим макровизначення порівняння паролів. Також можна помітити, що ми можемо бачити звичайні коментарі, і навіть приховані (виділено помаранчевим кольором).

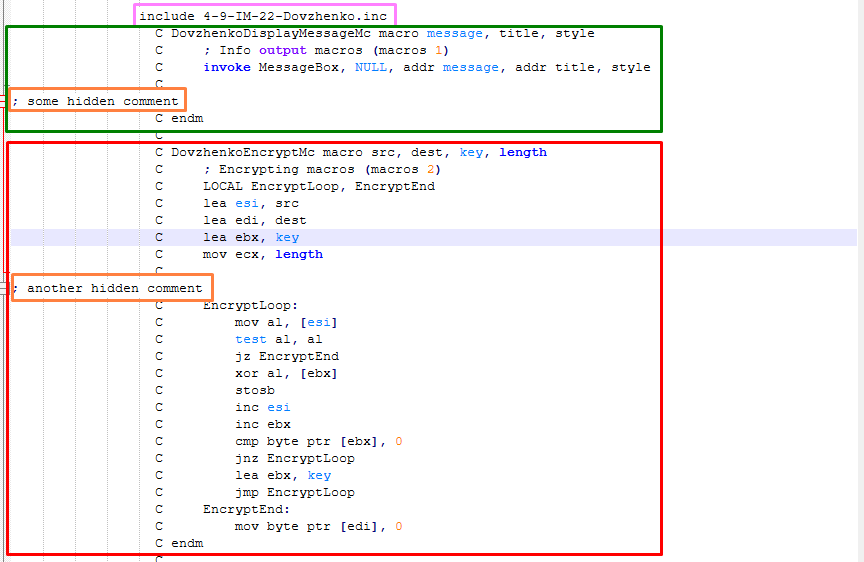
**Трансляції макросів**

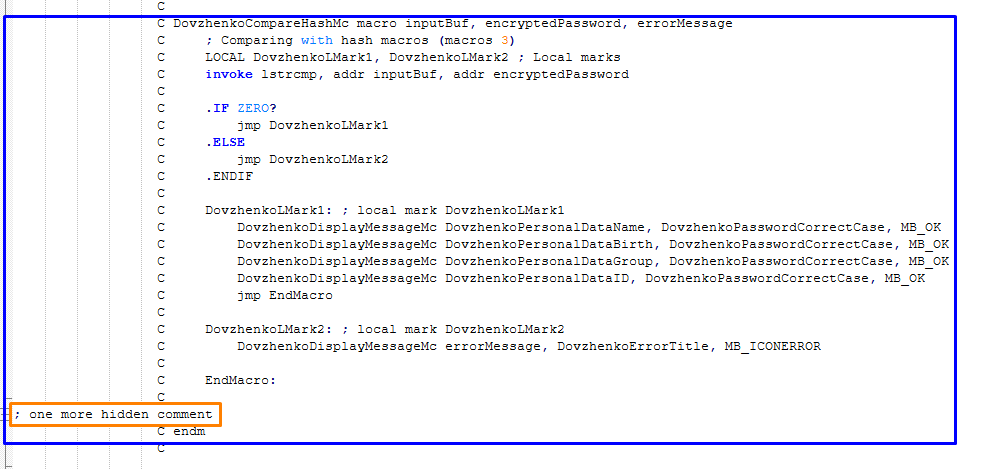


На скріншотах вище виділено трансляцію макросу для шифрування (другий макрос) зеленим кольором, трансляцію макросу для порівняння (третій макрос) жовтим кольором. Також виділено сірим кольором коди, які з’являються замість назв локальних міток при трансляції. Важливо зауважити, що через те що макрос виведення вікна з повідомленнями (перший макрос) викликається у макросі і транслюється в макрос для порівняння (третій макрос), ми його назви не бачимо. Ще важливо помітити, що ми тепер не бачимо приховані коментарі, проте бачимо звичайні коментарі. Також при трансляції приховано команди invoke, що містяться в макросах. Через це ми не бачимо перший макрос при трансляції, бо він складається з єдиного рядка з invoke і до того ж використовується в іншому макросі.

Тепер програма, де макроси винесено в окремий файл та імпортовано.

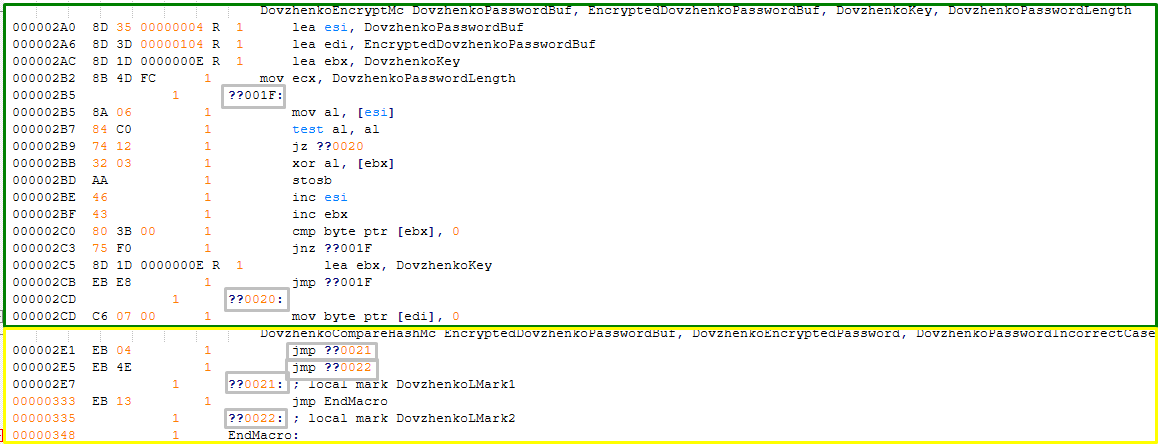
**Макровизначення**





У лістингу другої програми ми бачимо макровизначення одразу після include, що використовуємо для їхнього імпорту (виділено рожевим кольором). Виділено зеленим макровизначення для виводу тексту у вікні, червоним макровизначення для шифрування та синім для порівняння паролів. Також ми знову бачимо як звичайні, так і приховані коментарі (приховані виділені помаранчевим кольором).

**Трансляція**



Бачимо тут аналогічну ситуацію, а саме показано зеленим кольором трансляцію макросу для шифрування (другого макросу) та жовтим показано трансляцію для макросу порівняння (третього макросу). Назви макросу для виведення повідомлення у вікні (першого макросу) ми не бачимо, бо макрос складається з єдиного рядка invoke, який тут прихований, а викликається цей макрос з іншого макросу (третього макросу), і транслюється туди. Також бачимо, що замість назв локальних міток знову з’явилися коди, які виділено сірим.

**Висновок**

Отже, під час виконання лабораторної роботи було досліджено та застосовано макроси, також було створено дві програми з двома різними підходами (визначення макросів в файлі з рештою коду та використання макросів в окремому файлі). Було помічено спільні та відмінні риси.

Більше в цих підходах буде спільного, наприклад: перетворення в код локальних міток , приховано виклики invoke, сховані «приховані коментарі», а звичайні видні. Трансляції в обох програмах по суті ідентичні. А з відмінностей можу виокремити тільки розташування макровизначень.

Щодо макросів, то корисна річ, на мою думку, можна уникнути повторення коду та додати читабельність, що дуже корисно в контексті MASM32, що є по своїй суті складною мовою.