**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 у MASM32”

Виконав: Перевірив:

Студент 2-го курсу групи ІМ-13 Павлов Валерій Георгійович

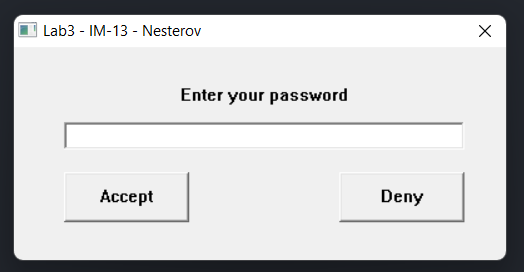
Нестеров Дмитро Васильович  
номер у списку групи: 16

Номер залікової книжки:IM-1320

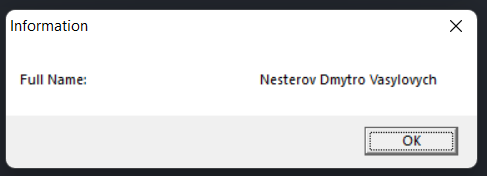
**Київ 2023**

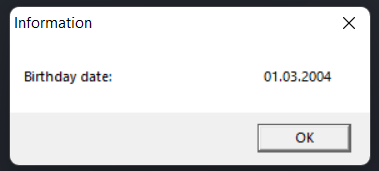
**Скріншоти виконання програми**

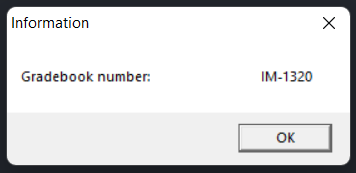
Спочатку з’являється вікно, де ми вводимо пароль:



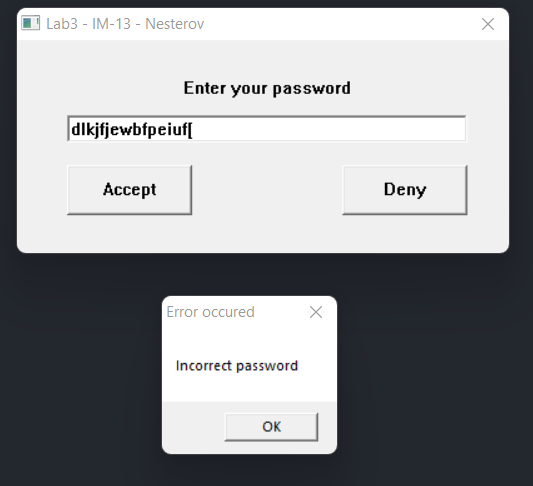
Після введення правильного паролю, порівняно з минулою лабораторною роботою, дані виводяться не усі разом в одному вікні, а послідовно у 3 різних вікнах, кожне з яких являє собою виклик макроса 1 з відповідною частиною тексту.







Повідомлення виводиться також за допомогою макроса 1 у випадку введення невірного паролю.



**Я за завданням визначив три макроси:**

1. **NESTER\_DISPLAY\_MESSAGE** – виводить на екран вікно з вмістом message та заголовком TITLE, де message і TITLE - параметри.
2. **NESTER\_XOR\_ENCRYPTING** – шифрування паролю
3. **NESTER\_XOR\_CHECKING** – порівняння паролів (правильного та зашифрованого)

**Також я додатково використав ще два макроси:**

1. **NESTER\_EQUAL\_PASSWORDS** **-** у випадку вірного введеного паролю виводить послідовно всю необхідну інформацію.
2. **NESTER\_NOT\_EQUAL\_PASSWORDS** - у випадку неправильного пароля виводить вікно з повідомленням про помилку.

**Шифрування**

У циклі шифруються усі символи пароля за одним і тим же ключем за допомогою базового метода - XOR

Ключ: **vaLera**

Пароль: **NEsTER**

**Переведення символів у двійкову систему використовуючи ASCII таблицю:**

v = 11810 = 011101102  N = 7810 = 010011102

a = 9710 = 011000012  E = 6910 = 010001012

L = 7610 = 010011002 s = 11510 = 011100112

e = 10110 = 011001012  T = 8410 = 010101002

r = 11410 = 011100102  E = 6910 = 010001012

a = 9710 = 011000012 R = 8210 = 010100102

**Процес шифрування:**

v = 011101102 XOR N = 010011102 => 001110002 = 5610 = 8

a = 011000012 XOR E = 010001012 => 001001002 = 3610 = $

L = 010011002 XOR s = 011100112 => 001111112 = 6310 = ?

e = 011001012 XOR T = 010101002 => 001100012 = 4910 = 1

r = 011100102 XOR E = 010001012 => 001101112 = 5510 = 7

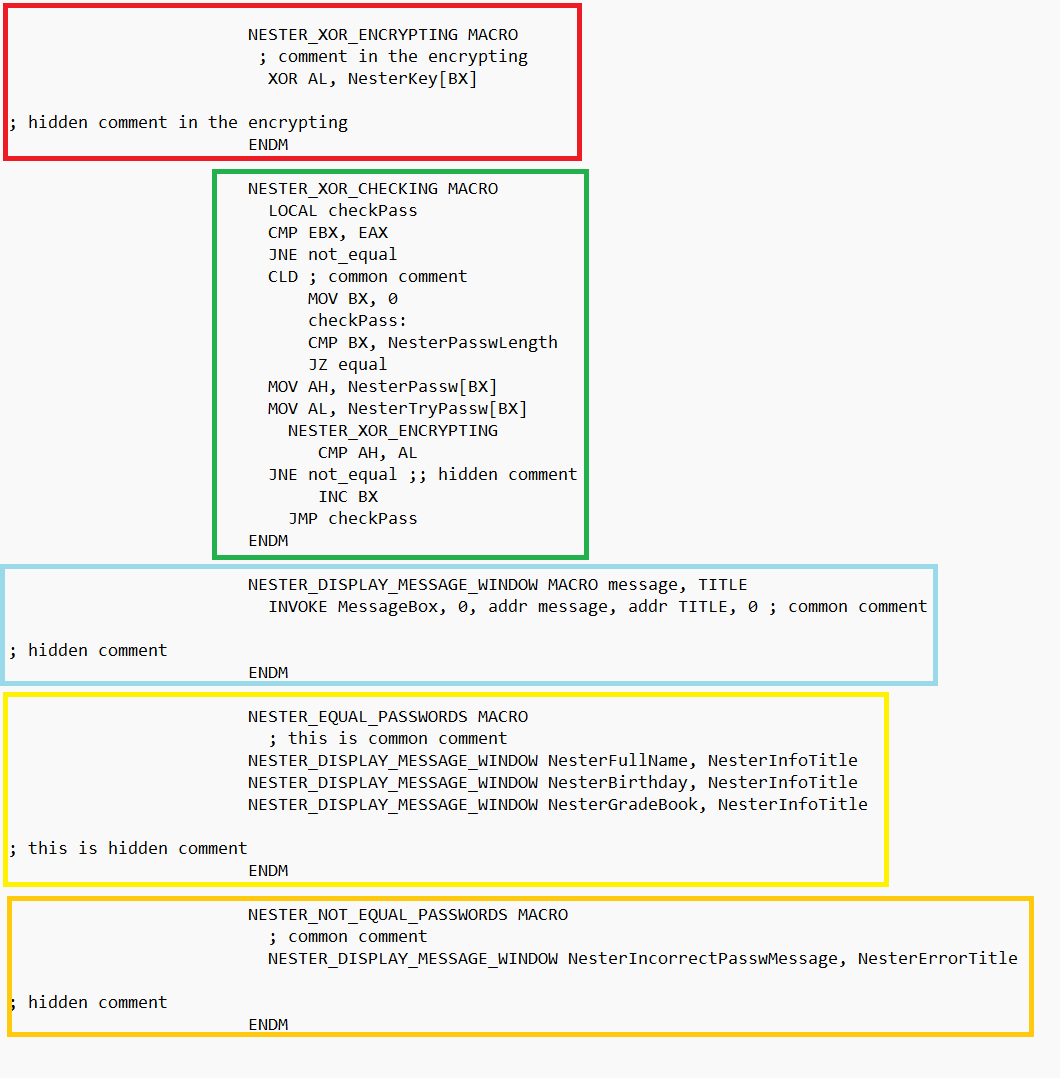
a = 011000012 XOR R = 010100102 => 001100112 = 5110 = 3

Отже зашифрований пароль - **8$?173**

**Дослідження лістингу**

**Макровизначення**

Макровизначення я визначив одразу після підключення бібліотек перед сигнатурою функції, яка відповідає за поведінку вікна, і в лістингу у мене їх також показано після підключення бібліотек, також ми зможемо побачити як звичайні, так і приховані коментарі.

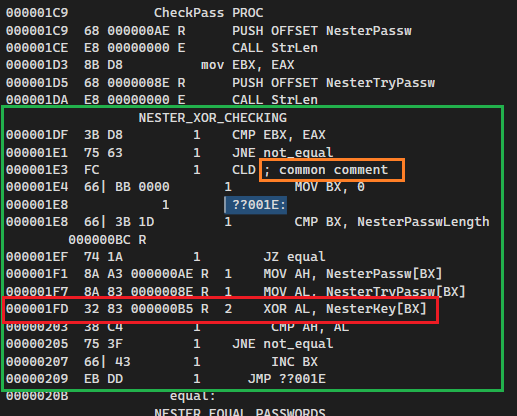


На скріншоті показані макровизначення макросу виводу вікна ( **голубий** ), шифрування (**червоний**), порівняння паролів (**зелений**). А також я використовував макроси для виведення вікон з інформацією за допомогою макросу виводу вікна після введення правильного, та виводу вікна з помилкою після неправильного паролів (**жовтий** та **помаранчевий** відповідно) Також варто зазначити, що у макросі порівняння використано локальну мітку, за її поведінкою при трансляції можна буде спостерігати далі.

**Трансляції макросів:**

Дослідження лістингу, якщо макроси написані прямо у файлі з кодом

На скріншоті нижче бачимо трансляції макросів шифрування (**червоний**) та порівняння (**зелений**) паролів. Оскільки я викликаю макрос шифрування в макросі порівняння, то він транслюється у останній і тому ми не бачимо тут NESTER\_XOR\_ENCRYPTING. Бачимо, що назви локальних міток були замінені на коди (виділено курсором). Прихованих коментарів не видно, але звичайні ми побачити можемо. Я обвів коментар помаранчевим.



Я створив два додаткових макроса які реагують на правильний (**жовтий**) і

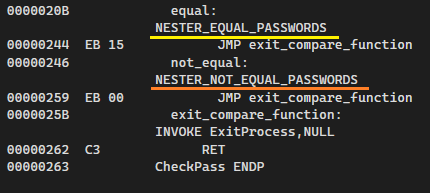
неправильний (**помаранчевий**) паролі відповідно, а вони вже в свою

чергу викликають макроси виводу вікна з необхідними параметрами, які

як і в попередньому випадку, транслюються у макроси, що їх викликали,

але окрім цього в усіх трансляціях приховано рядки з викликами функцій

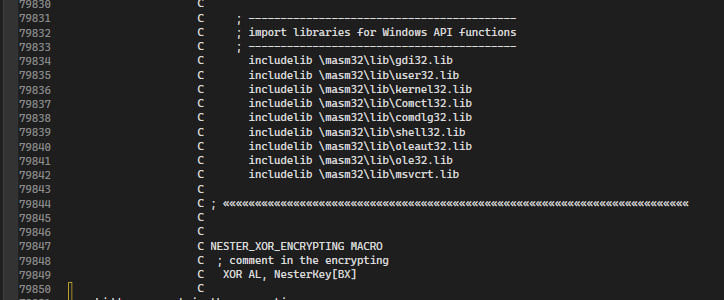
через invoke, тому ми бачимо у лістинг файлу таку картину:



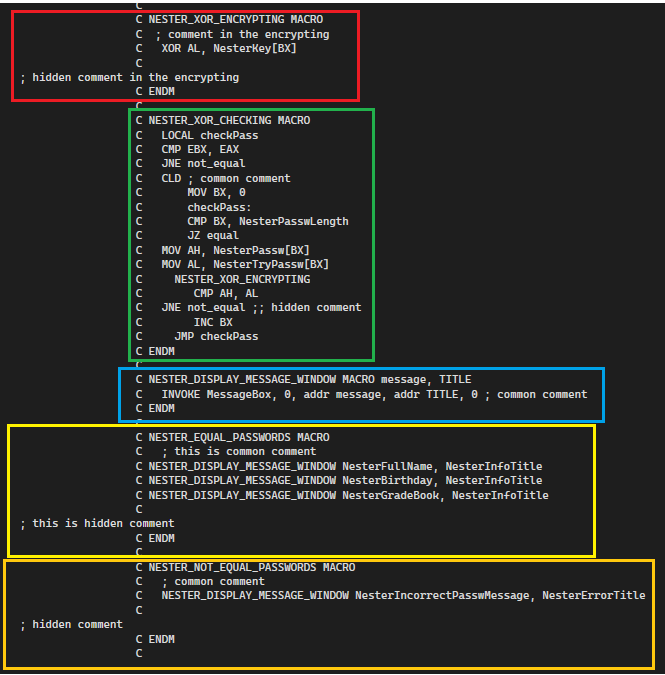
Дослідження лістингу, якщо макроси підключено за допомогою include

**Макровизначення:**



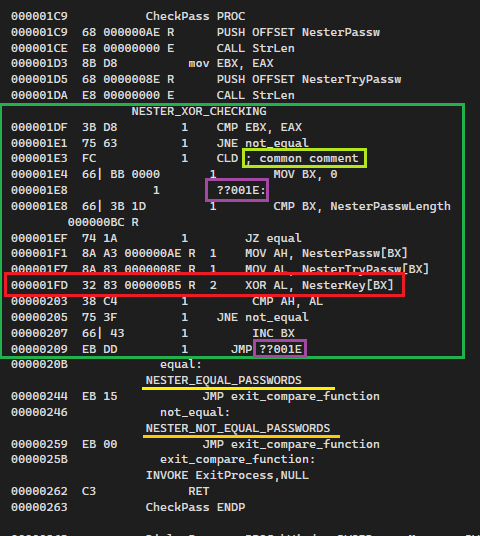


Макровизначення показано після підключення файлу з макросами. І знову можемо побачити звичайні та приховані коментарі На скріншоті я навів макровизначення макросу виводу вікна (**голубий**), шифрування (**червоний**), порівняння паролів (**зелений**), а також двох моїх для реагування на правильній та неправильний паролі, **жовтий** та **помаранчевий** відповідно.



**Трансляція:**

На скріншоті нижче я навів трансляцію макросу шифрування (**червоний**), порівняння паролів (**зелений**), а також двох моїх для реагування на правильній та неправильний паролі, **жовтий** та **помаранчевий** відповідно. У трансляціях знову відсутні приховані коментарі, **салатовим** кольором я відмітив звичайний коментар. Також приховано виклики функцій за допомогою invoke і назви локальних міток замінені на коди, я обвів **фіолетовим** кольором два приклади цього. Також можна було б побачити, що замість параметрів підставлено передані аргументи, але через те, що макроси, які викликають діалогове вікно, у мене самі викликаються у іншому макросі, вони транслюються у них і тому ми бачимо наступну картину:



**Висновок**

Під час виконання цієї лабораторної роботи я дослідив та використав на практиці таку сутність у MASM32, як макроси. Я зрозумів принцип їх роботи та навчився писати власноруч, спробував 2 способи їх використання: за допомогою оголошення макросів в коді та імпортуванням їх із файлу. Я ще у минулій лабораторній використовував локальні мітки, але зараз я більш детально дослідив їх поведінку. Як виявилося, назви цих локальних міток при трансляції приймають у розширеному лістингу вигляд унікальних кодів.

У цій лабораторній мною було використано макроси двома різними способами: безпосередньо у файлі та за допомогою підключення їх до файлу, після дослідження було знайдено дуже багато спільного, аде були і відмінності.

Відмінне:

* місцезнаходження макровизначень (там, де вони визначались у 1 способі та після підключення файлу з визначенням макросів, у другому)

Схоже:

* у макровизначеннях коментарі видно як звичайні, так і приховані в обох способах
* у трансляціях макросів одразу під викликом макросу вставлено їх код
* коментарі у трансляції видно тільки звичайні
* замість параметрів підставляються передані аргументи (це я помітив коли починав робити лабораторну, коли ще не створював додаткові макроси NESTER\_EQUAL\_PASSWORDS і NESTER\_NOT\_EQUAL\_PASSWORDS)
* трансляція при виклику 1 макроса з іншого ідентична.
* назви локальних міток представлені кодами, приховано рядки з викликом функцій за допомогою invoke.

У результаті, можу сказати, що трансляції в обох випадках ідентичні, у макровизначеннях відмінне тільки розташування у розширеному лістингу. Макроси дуже зручні у використанні та нескладні у написанні. Вони значно підвищують зрозумілість коду та зменшують кількість його повторення (що без сумнівів варто нашої уваги) при цьому не змінюючи логіку.