**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»

Виконала: Перевірив:

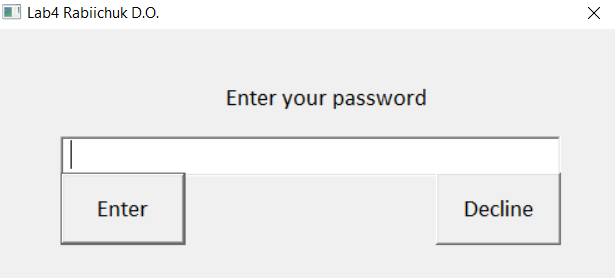
Студентка 2-го курсу групи ІМ-21 Павлов В.Г.

Рабійчук Дар’я Олександрівна   
номер у списку групи: 18

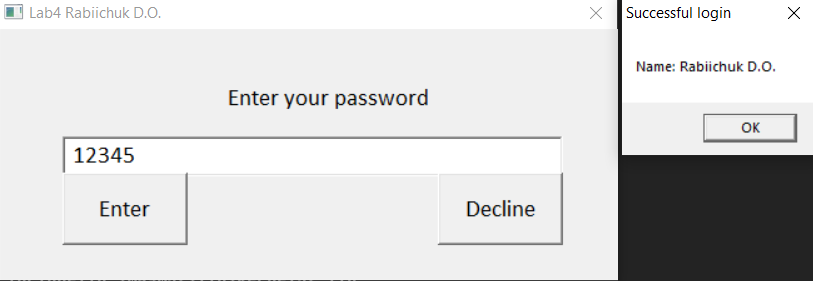
Київ 2024

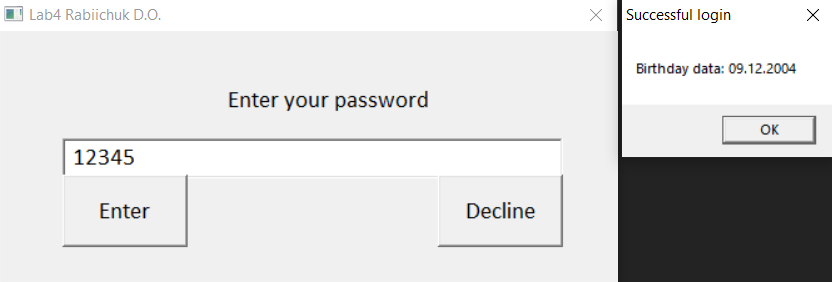
**Скріншоти виконання програми:**

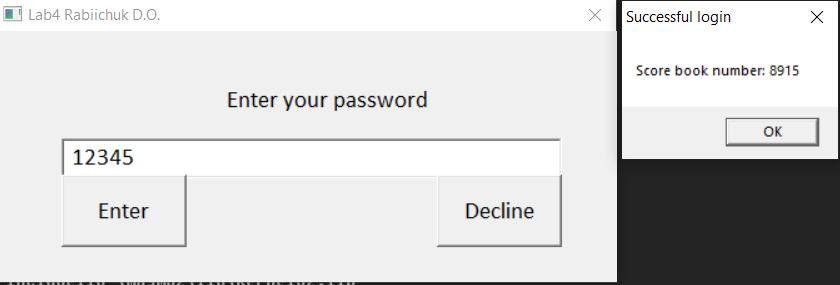
Вікно введення паролю:



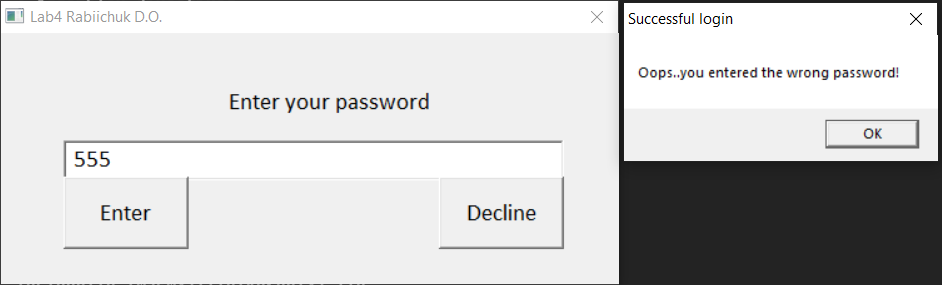
Введення правильного паролю:







Введення неправильного паролю:



У даній програмі були створені наступні макроси:

1. ShowMessageBox: Цей макрос використовується для відображення повідомлення в діалоговому вікні. Приймає текст повідомлення як параметр і викликає функцію MessageBox з відповідними параметрами.

2. EncryptInputString: Цей макрос використовується для шифрування введеного рядка. У цьому випадку, це приклад реалізації шифрування за допомогою XOR з ключем.

3. CompareEncryptedString: Цей макрос використовується для порівняння зашифрованого рядка з хешем пароля. Він порівнює кожен символ зашифрованого рядка з символом хешу пароля і підраховує кількість співпадаючих символів.

Ці макроси роблять програму більш модульною і спрощують її розуміння та супровід. Вони дозволяють використовувати один і той же код в різних частинах програми, уникати дублювання коду.

**Шифрування паролю за допомогою ключа:**

У програмі ми шифруємо введений користувачем пароль методом XOR. Ось детальний процес шифрування:

Наш пароль 12345:

ASCII-код для "1" - 49, "2" - 50, "3" - 51, "4" - 52, "5" - 53.

В бінарному вигляді числа 12345:

1: 00110001

2: 00110010

3: 00110011

4: 00110100

5: 00110101

Таким чином, в бінарному вигляді число 12345 буде мати вигляд:

00110001 00110010 00110011 00110100 00110101

Наше слово-ключ zEBra:

ASCII значення для кожного символу "zEBra":

z = 122

E = 69

B = 66

r = 114

a = 97

Тепер переведемо кожне з цих чисел у восьмирозрядний бінарний формат:

122 = 01111010

69 = 01000101

66 = 01000010

114 = 01110010

97 = 01100001

Таким чином, рядок "zEBra" у бінарному форматі буде:

01111010 01000101 01000010 01110010 01100001

Тепер виконаємо операцію XOR для кожної пари відповідних бітів:

1. Результат XOR для '1' та 'z': 00110001 XOR 01111010 = 01001011

2. Результат XOR для '2' та 'E': 00110010 XOR 01000101 = 01110111

3. Результат XOR для '3' та 'B': 00110011 XOR 01000010 = 01110001

4. Результат XOR для '4' та 'r': 00110100 XOR 01110010 = 01000110

5. Результат XOR для '5' та 'a': 00110101 XOR 01100001 = 01010100

Зашифрований текст у вигляді бінарного рядка: 01001011 01110111 01110001 01000110 01010100

1. 01001011 = 75 (в десятковій системі)

2. 01110111 = 119 (в десятковій системі)

3. 01110001 = 113 (в десятковій системі)

4. 01000110 = 70 (в десятковій системі)

5. 01010100 = 84 (в десятковій системі)

Тепер переведемо ці числа у відповідні символи ASCII:

1. 75 = 'K'

2. 119 = 'w'

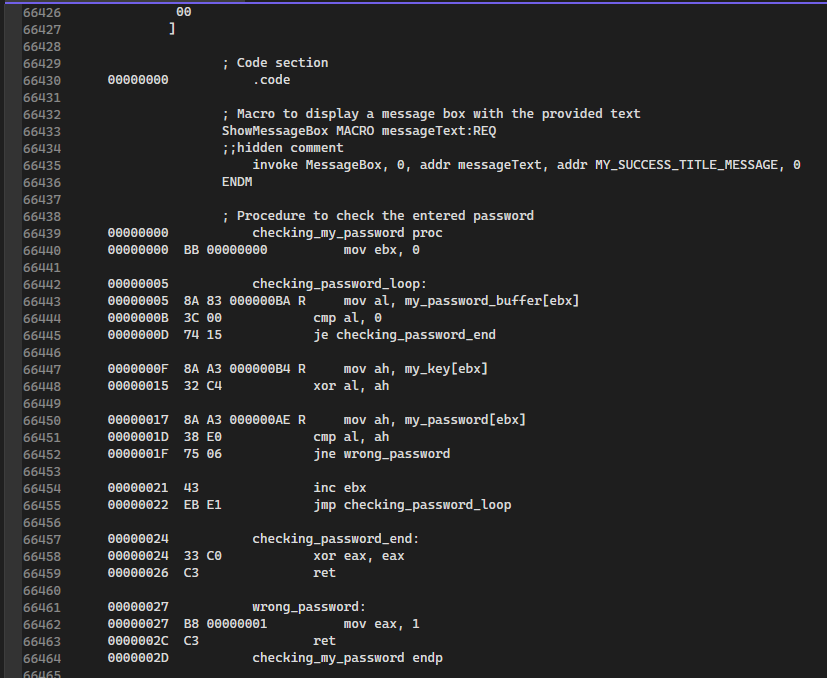
3. 113 = 'q'

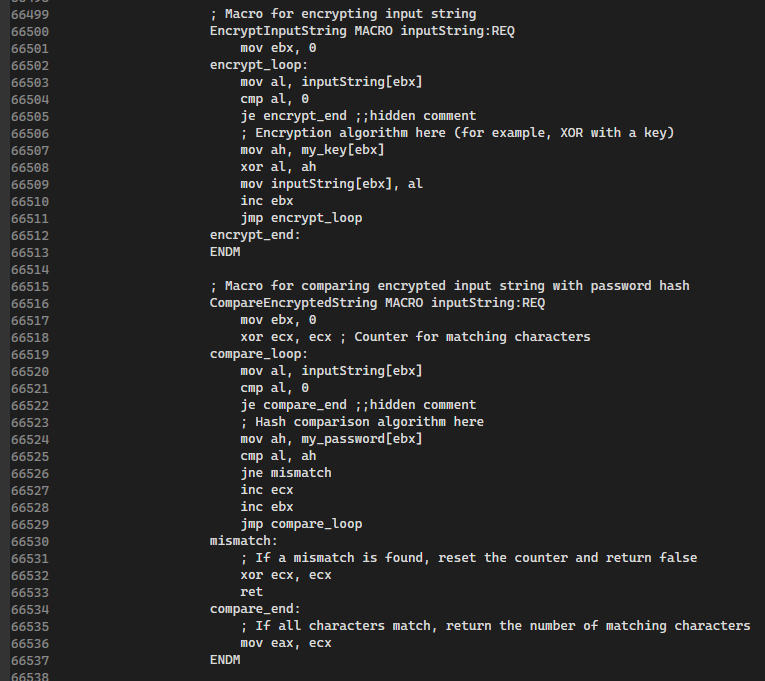
4. 70 = 'F'

5. 84 = 'T'

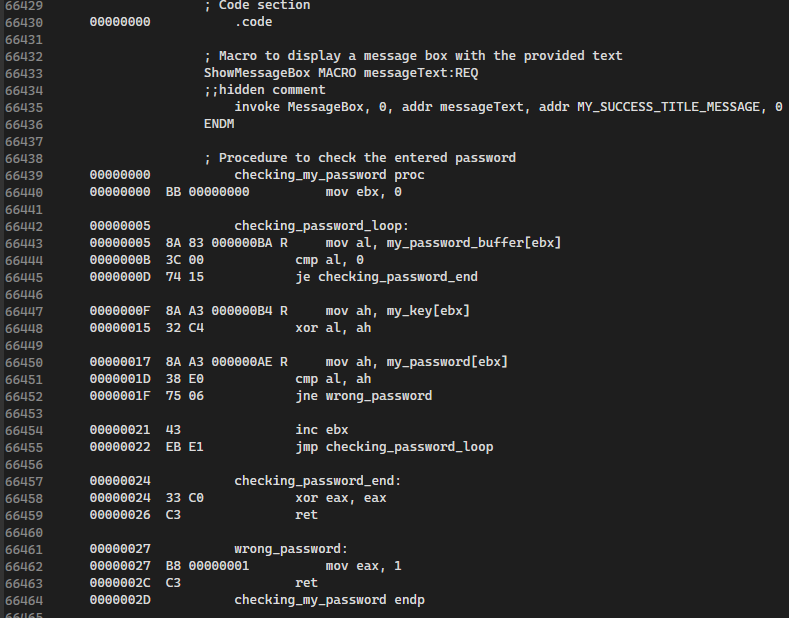
Отже, зашифрований текст у вигляді ASCII буде: "KwqFT".

**Дослідження файлів лістингу:**



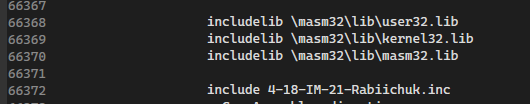


При дослідженні файлу 4-18-ІМ-21-Rabiichuk.lst файлу 4-18-ІМ-21-Rabiichuk.asm я побачила, що макроси визначені вже після константних даних та у файлі видно як звичайні так і приховані коментарі. На скріншотах червоним кольором я виділила усі 3 макроси, зеленим звичайні коментарі та жовтим приховані.

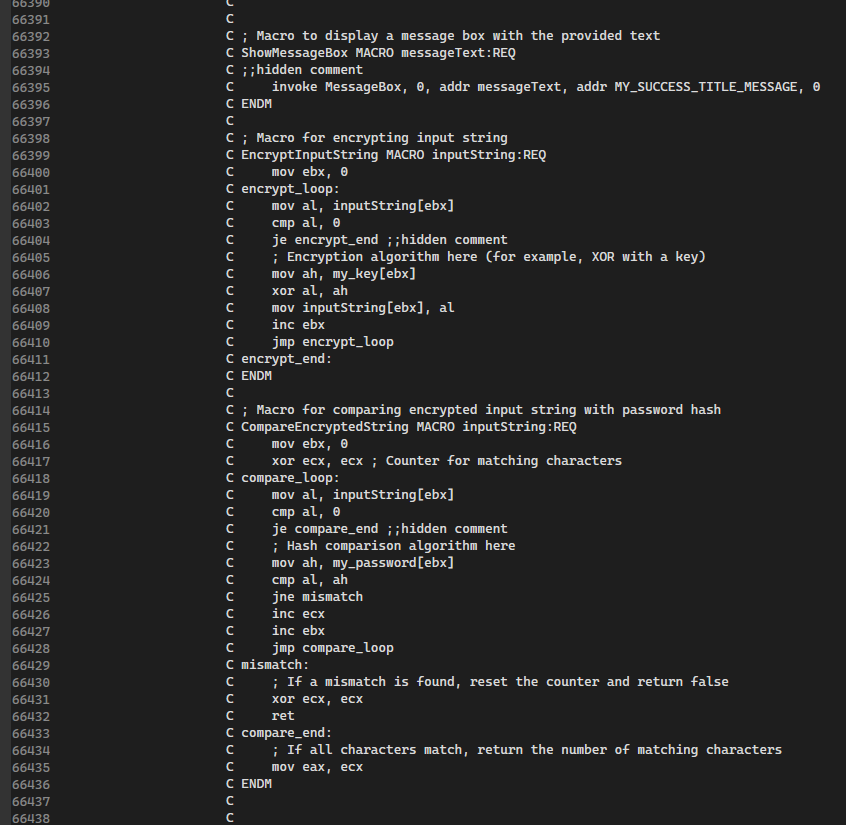


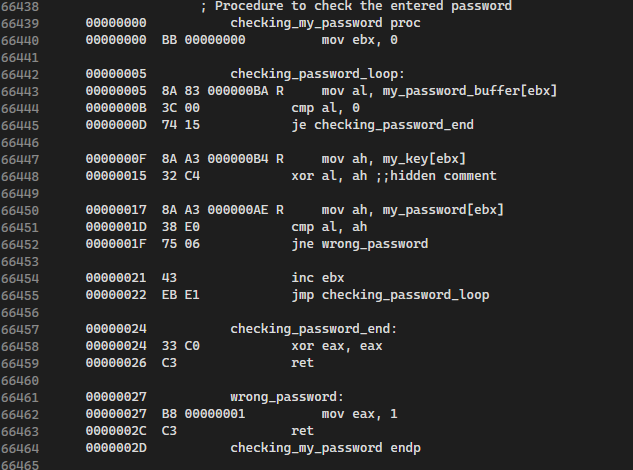
На цьому скріншоті показано трансляцію макросів, що виділена блакитним кольором.

Макровизначення:



Досліджуючи файл 4-18-ІМ-21-Rabiichuk-INC.lst файлу 4-18-ІМ-21-Rabiichuk-INC.asm можемо побачити, що макроси розташовані одразу після їх визначення (виділено блакитним кольором).





Дивлячись на трансляцію даного файлу (обведено зеленим кольором) бачимо, що відмінностей між програмою з підключенням окремого .inc файлу та програмою з прописаними макросами у виділеній частині коду немає.

При порівняні обидвох версій програм я зрозуміла, що принципової різниці у виконанні програм та у лістингу немає. Щодо відмінностей, файли .lst відрізняються лише розташуванням макровизначень. У першій програмі вони знаходяться після визначення констант, а у другому на початку файлу. Щодо схожості, в обох випадках бачимо як приховані так і звичайні коментарі.

**Висновок:**

У цій лабораторній роботі я вперше спробувала створити та вжити у своїх програмах таку технологію як макровизначення на мові Ассемблер. Також я навчилась аналізувати лістинг-файли. При їх досліджені я побачила, що незалежно від того чи макроси прописані напряму у .asm файлі чи підключенні через include з .inc файла, лістинги майже не відрізняються. Головна відмінність полягає в розташуванні макровизначень.

У результаті можу сказати, що макроси є дуже зручними та нескладними у їх написанні та використані. При розробці їх користь полягає в тому, що вони зменшують кількість повторюваного коду без змінення логіки програми, внаслідок чого його буде легше читати та видозмінювати в майбутньому.