**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 групи ІП-93 доцент

Марченко Максим Олегович Павлов В.Г.

номер залікової книжки: 9316

Київ 2020

Мета роботи : Вивчити технології створення і вживання макросів. Дослідження результатів роботи макрогенератору masm32.

Макроси - це символьне ім’я, що замінює собою певну послідовність команд, і робить це шляхом генерування заданого коду у місці, де викликається макрос.

Для створення макросу перед частиною з кодом та даними програми, записується назва макросу, далі ключове слово macro, та необхідні вхідні дані. Після цього записується код, що повинен виконуватися у макросі. Кінець макросу позначається ключовим словом endm.

Приклад макросу (з програми 4-13-IP93-Marczenko):

outMessage macro info

invoke MessageBox, 0, addr info, addr info\_title, 0

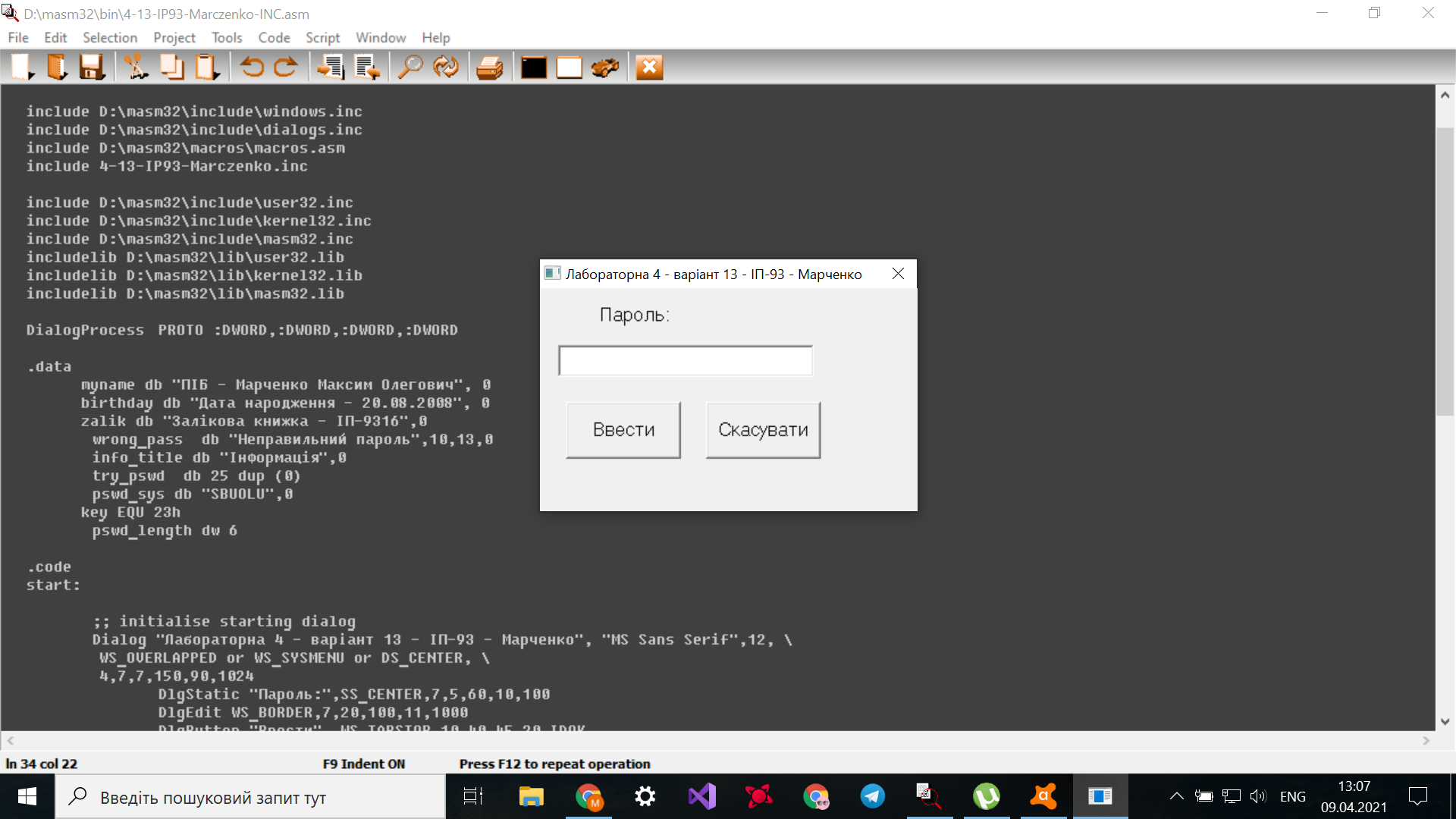
endm

Тут, зокрема info - вхідні дані макросу, а info\_title - глобально визначена змінна.

Макроси використовуються для спрощення написання та розуміння програми, вони підвищують її читабельність. У порівнянні з процедурами, макроси є більш швидкодійними, проте у певних випадках вимагають виділення більшої пам’яті.

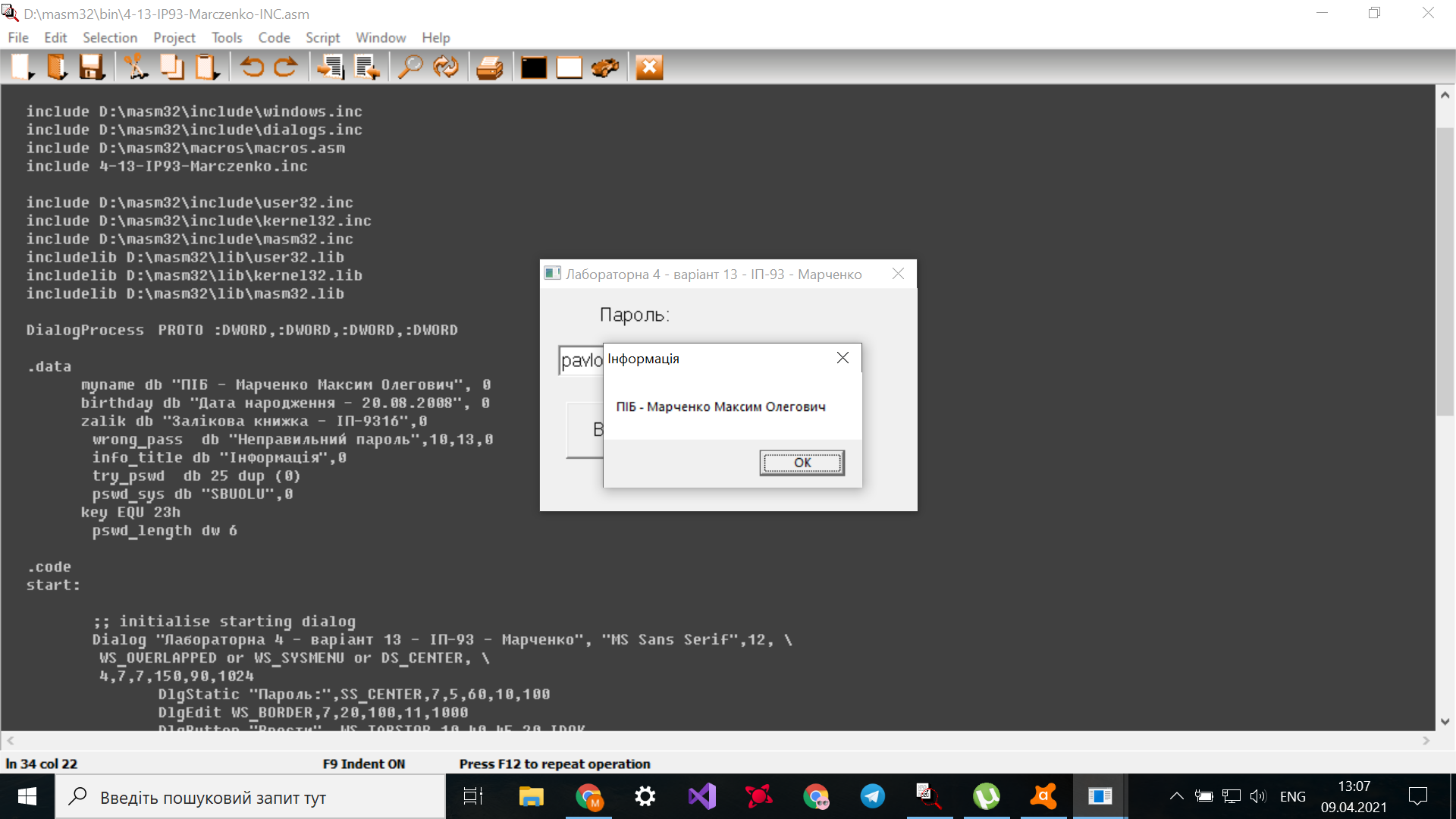
Скріншоти виконання програми

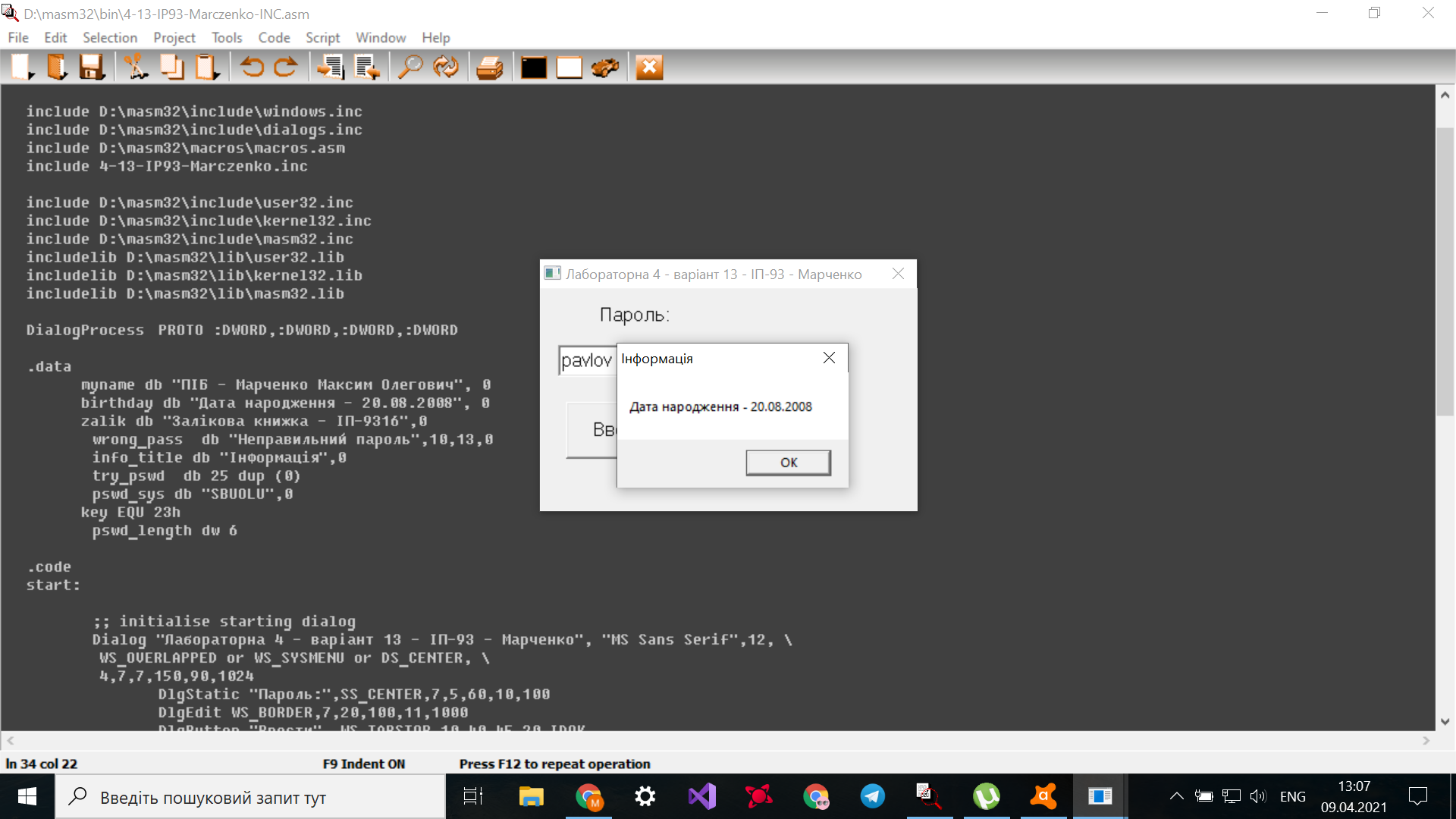
Вікно введення паролю

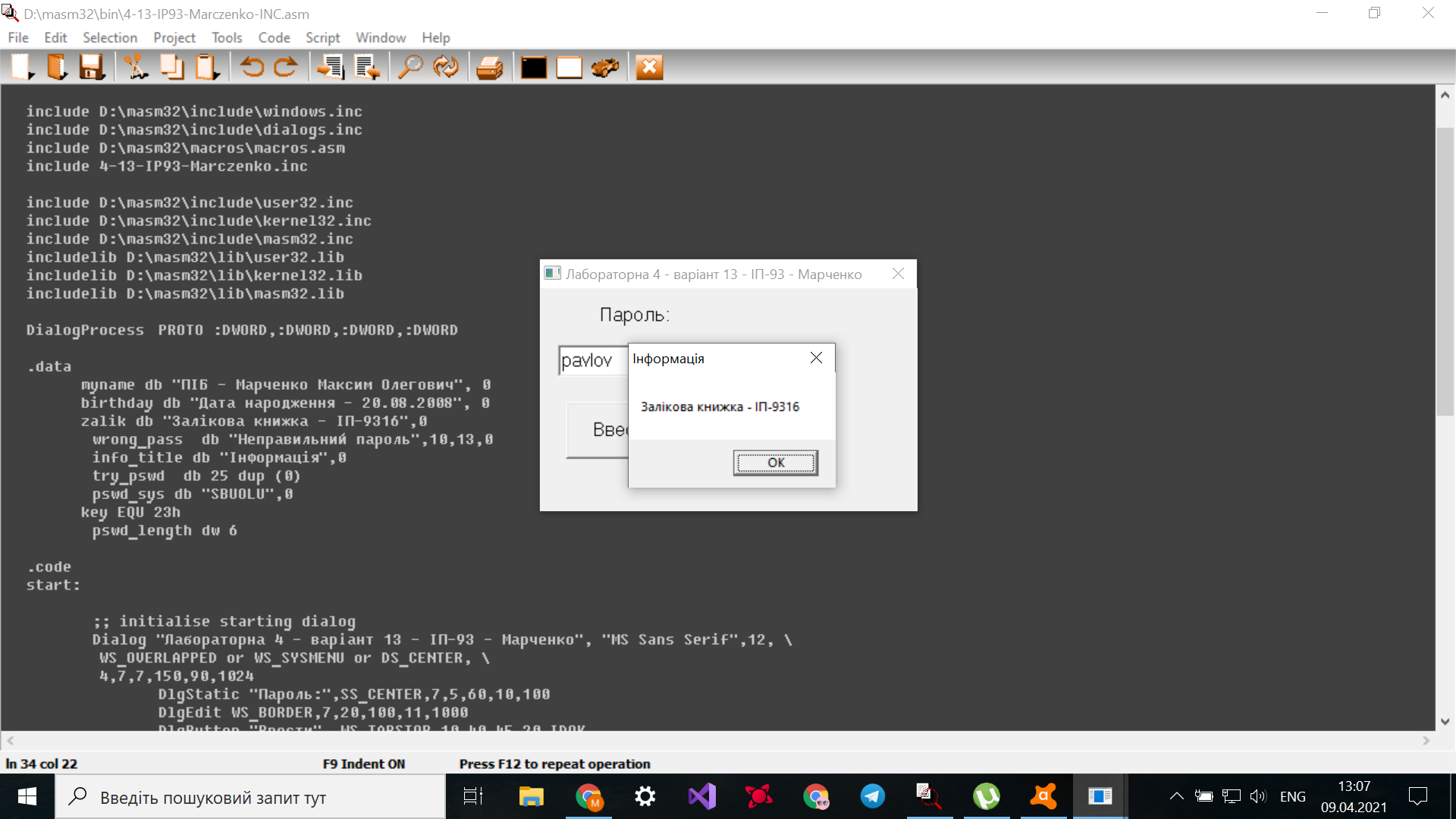


Після введення правильного паролю

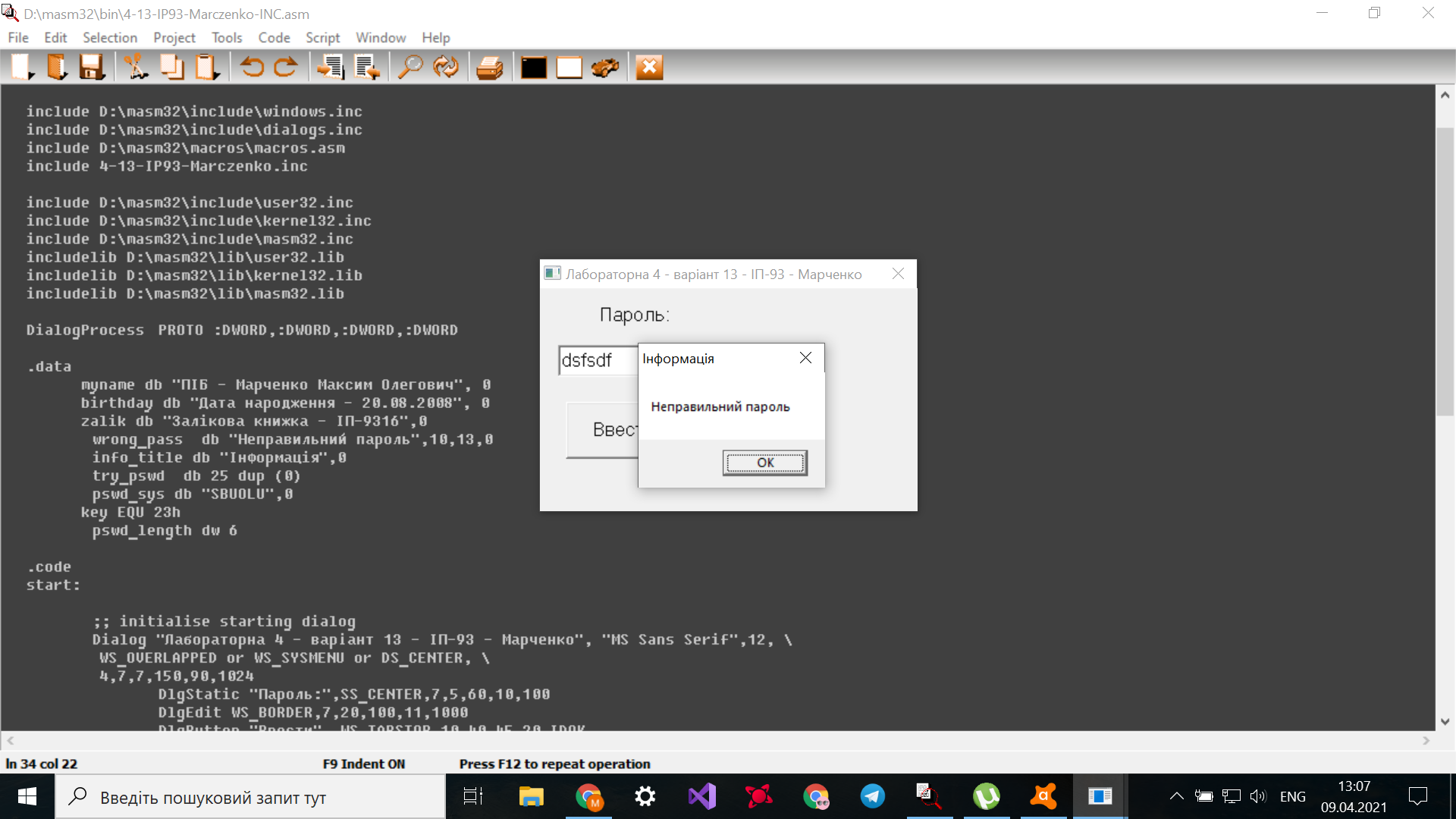
Послідовно відображаються три вікна, кожне ж яких появляється після закриття попереднього.





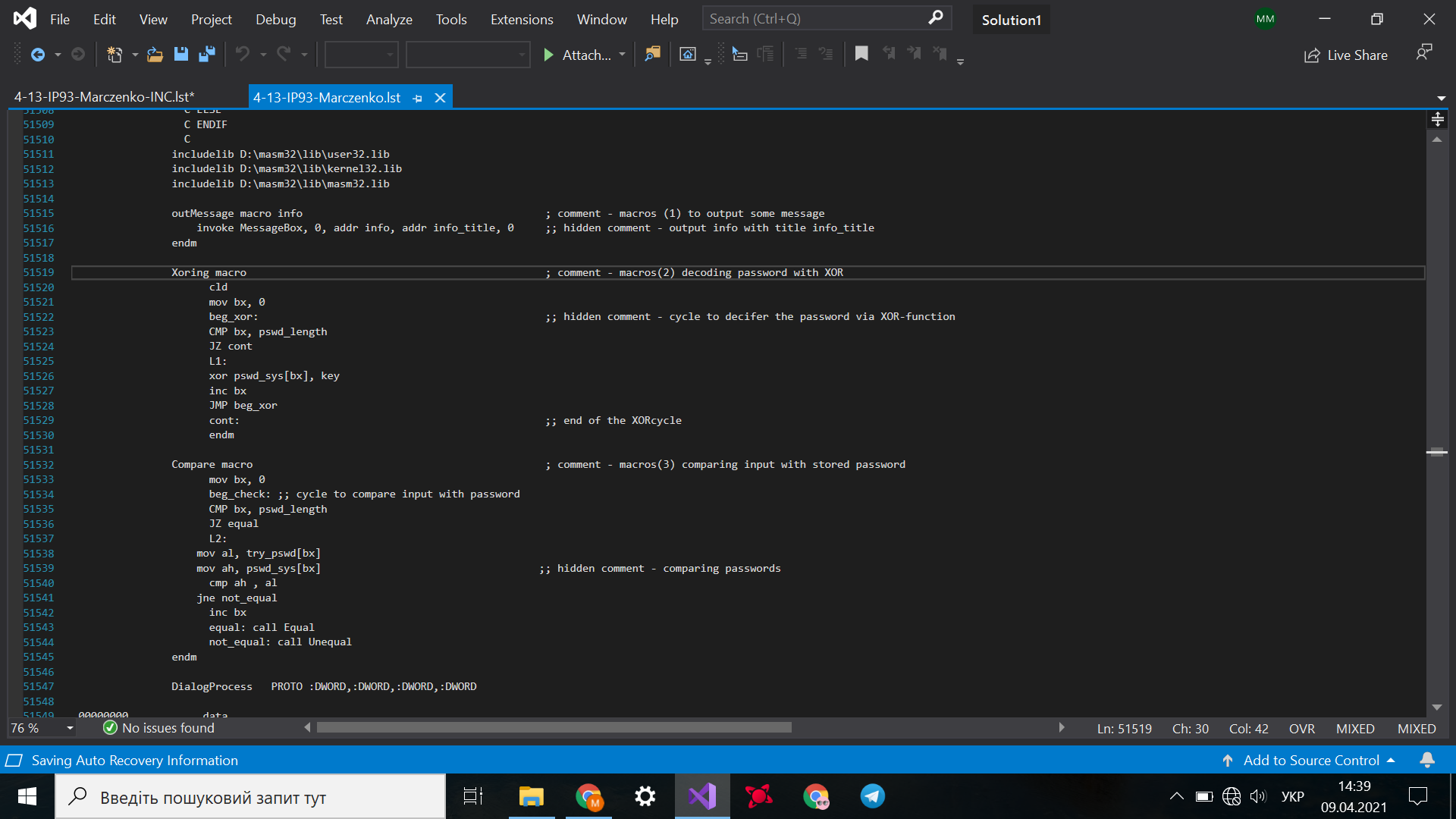


Після введення неправильного паролю

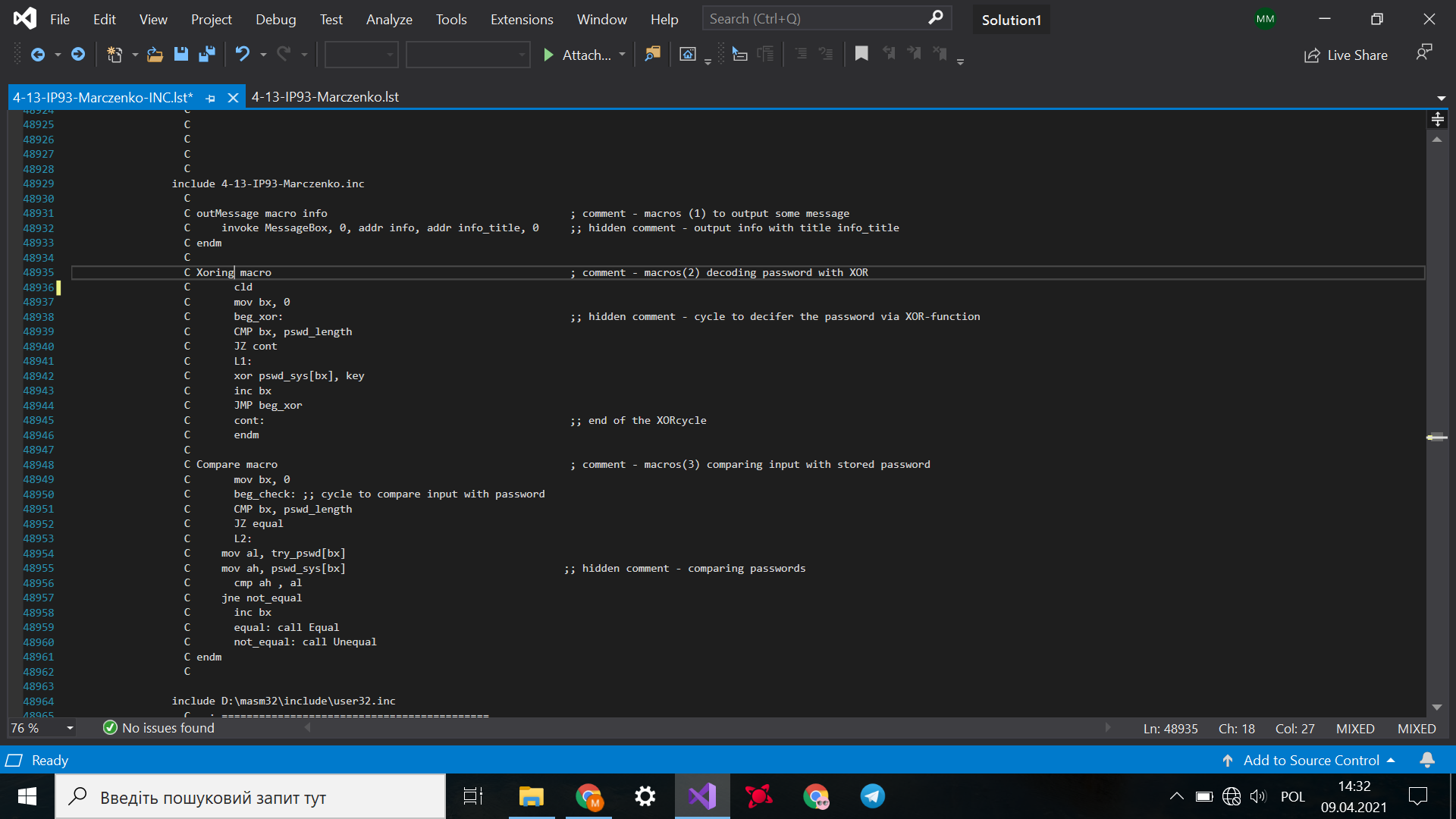


Дослідження лістингу

Версія з макросами у файлі разом з кодом



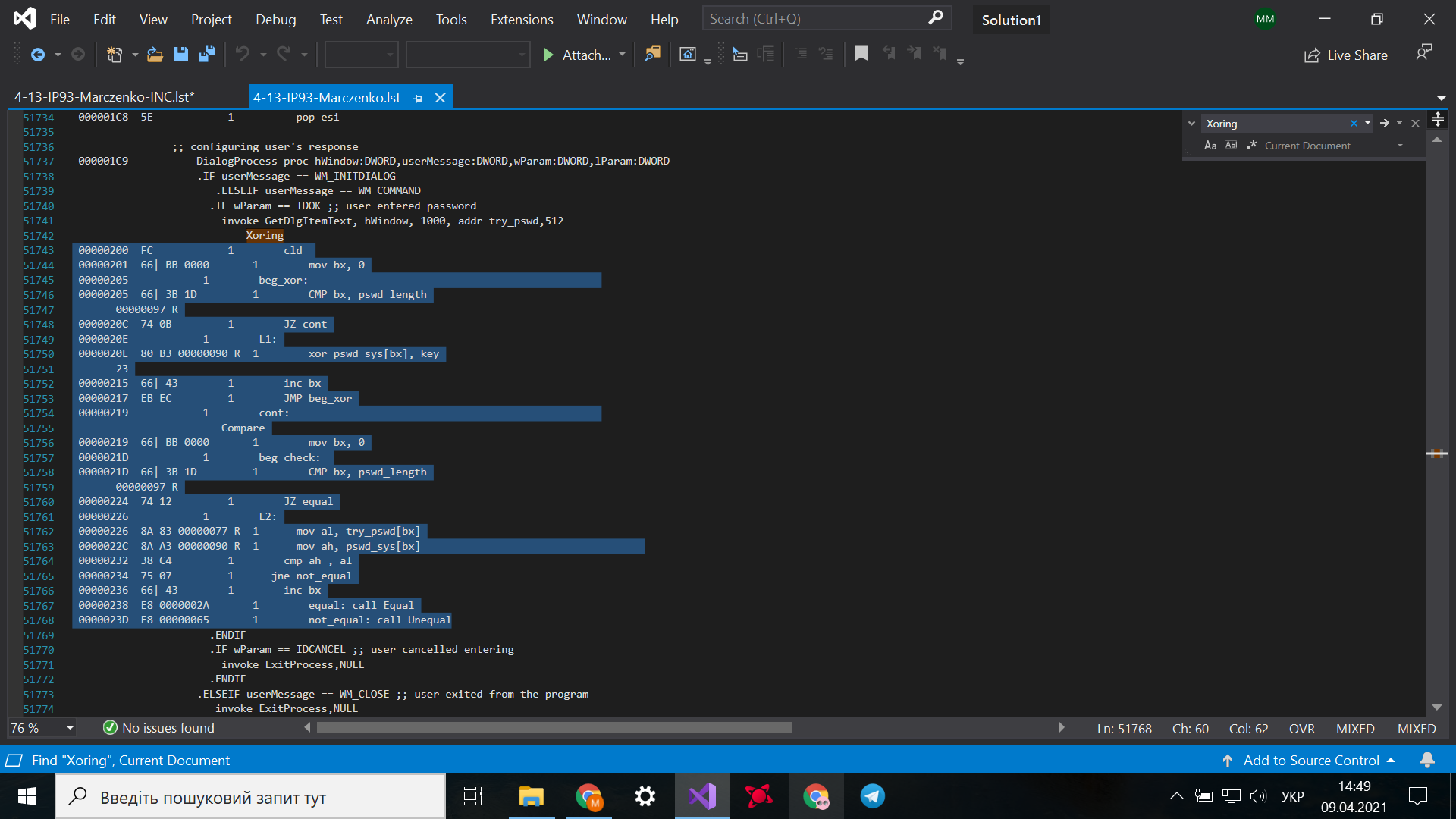
Версія з макросами у окремому файлі



Як видно, основна відмінність лістингів - те, що означення макросів розташоване у різних місцях.

У версії, де макроси розташовані разом з кодом, їх місце у лістингу таке ж, як і у файлі .asm-розширення, тобто перед секцією .data та означенням функції DialogProcess.

У іншій ж версії, визначення макросів показане у секції, де відбувається включення файлу, що їх містить (з розширенням .inc).



На скріншоті зображено, як макрос відображається вже при виконанні програми.

Висновки

Під час виконання лабораторної роботи я ознайомився з структурою файлів формату .exe - розширення виконуваного файлу, що широко застосовується в системах DOS та Microsoft Windows. Файл EXE складається з двох частин: заголовок (інформація до завантажувача програми) та модуль завантаження. Також було досліджено деякі параметри файлу, та способи їх знаходження та зміни, зокрема за допомогою програми PE Tools. За допомогою цього ж застосунку було досліджено імпортовані бібліотеки та функції.

Додаток: Код програми 3-13-IP93-MarczenkoXOR

.386

.model flat, stdcall

option casemap :none

include D:\masm32\include\windows.inc

include D:\masm32\include\dialogs.inc

include D:\masm32\macros\macros.asm

include D:\masm32\include\user32.inc

include D:\masm32\include\kernel32.inc

include D:\masm32\include\masm32.inc

includelib D:\masm32\lib\user32.lib

includelib D:\masm32\lib\kernel32.lib

includelib D:\masm32\lib\masm32.lib

outMessage macro info ; comment - macros (1) to output some message

invoke MessageBox, 0, addr info, addr info\_title, 0 ;; hidden comment - output info with title info\_title

endm

Xoring macro ; comment - macros(2) decoding password with XOR

cld

mov bx, 0

beg\_xor: ;; hidden comment - cycle to decifer the password via XOR-function

CMP bx, pswd\_length

JZ cont

L1:

xor pswd\_sys[bx], key

inc bx

JMP beg\_xor

cont: ;; end of the XORcycle

endm

Compare macro ; comment - macros(3) comparing input with stored password

mov bx, 0

beg\_check: ;; cycle to compare input with password

CMP bx, pswd\_length

JZ equal

L2:

mov al, try\_pswd[bx]

mov ah, pswd\_sys[bx] ;; hidden comment - comparing passwords

cmp ah , al

jne not\_equal

inc bx

equal: call Equal

not\_equal: call Unequal

endm

DialogProcess PROTO :DWORD,:DWORD,:DWORD,:DWORD

.data

myname db "ПIБ - Марченко Максим Олегович", 0

birthday db "Дата народження - 20.08.2008", 0

zalik db "Залiкова книжка - IП-9316",0

wrong\_pass db "Неправильний пароль",10,13,0

info\_title db "Iнформацiя",0

try\_pswd db 25 dup (0)

pswd\_sys db "SBUOLU",0

key EQU 23h

pswd\_length dw 6

.code

start:

;; initialise starting dialog

Dialog "Лабораторна 4 - варіант 13 - ІП-93 - Марченко", "MS Sans Serif",12, \

WS\_OVERLAPPED or WS\_SYSMENU or DS\_CENTER, \

4,7,7,150,90,1024

DlgStatic "Пароль:",SS\_CENTER,7,5,60,10,100

DlgEdit WS\_BORDER,7,20,100,11,1000

DlgButton "Ввести", WS\_TABSTOP,10,40,45,20,IDOK

DlgButton "Скасувати", WS\_TABSTOP,65,40,45,20,IDCANCEL

CallModalDialog 0,0,DialogProcess,NULL ;; calling starting dialog

;; configuring user's response

DialogProcess proc hWindow:DWORD,userMessage:DWORD,wParam:DWORD,lParam:DWORD

.IF userMessage == WM\_INITDIALOG

.ELSEIF userMessage == WM\_COMMAND

.IF wParam == IDOK ;; user entered password

invoke GetDlgItemText, hWindow, 1000, addr try\_pswd,512

Xoring

Compare

.ENDIF

.IF wParam == IDCANCEL ;; user cancelled entering

invoke ExitProcess,NULL

.ENDIF

.ELSEIF userMessage == WM\_CLOSE ;; user exited from the program

invoke ExitProcess,NULL

.ENDIF

return 0

DialogProcess endp

Equal proc

outMessage myname

outMessage birthday

outMessage zalik

invoke ExitProcess,NULL

Equal endp

Unequal proc

outMessage wrong\_pass

invoke ExitProcess,NULL

Unequal endp

end start