НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ ТА ПРОГРАМУВАННЯ»

«Системне програмування »

*Звiт з лабораторної роботи №4*

*Тема: «*Структури, макроси*»*

Виконав:

ст. гр. KIT-119D

Гряник Г.В.

Перевірив:

Межерицький С.Г.

Харків – 2020

**Мета :**

набути практичних навичок складання, налагодження та виконання програм з використанням структур, написаних мовою ассемблера для платформи х64 в середовищі masm64..

набути практичних навичок складання, налагодження та виконання програм з використанням макросів, написаних мовою ассемблера для співпроцесора платформи х64 в середовищі masm64.

***Постановка задачі***

Виконати завдання на masm64 з використанням повноцінного Windows-вікна, функції MessageBoxIndirect з відображенням своєї фотографії як ікони, свого прізвища, номером своєї навчальної групи, а також завдання та отримані результати. За бажанням можете організувати розрахунки рівняння зі зміною однієї змінної з певним кроком.

***Індивідуальне завдання***

Варіант 6.

* Задана матриця 4 х 6. Визначити суму елементів кожного стовпця.
* Згідно номеру студента в групі вибрати варіант завдання і написати на асемблері програму обчислення одного з виразів, в якому одна з змінних змінюється кілька разів: (1,1ab – 3)/ab;
* Задана послідовність структур. Структура містить поля з торгової діяльності комп'ютерної фірми: назва товару (комп'ютер, модем, гвинт), ціна (в доларах), кількість проданих одиниць. Обчислити прибуток фірми (підсумовування "кількості проданих одиниць помножену на ціну").

**Текст програми**

Програма 1: (LR4-1.1)

include \masm64\include64\masm64rt.inc ; библиотеки для подключения

IDI\_ICON EQU 1001

MSGBOXPARAMSA STRUCT

cbSize DWORD ?,?

hwndOwner QWORD ?

hInstance QWORD ?

lpszText QWORD ?

lpszCaption QWORD ?

dwStyle DWORD ?,?

lpszIcon QWORD ?

dwContextHelpId QWORD ?

lpfnMsgBoxCallback QWORD ?

dwLanguageId DWORD ?,?

MSGBOXPARAMSA ENDS

DATE1 STRUCT ; определения данные СТРУКТУРА с именем DATE1

Name1 dW ? ; имя 1 поля структуры

Name2 dw ? ; имя 2 поля структуры

Name3 dw ? ; имя 3 поля структуры

Name4 dw ? ; имя 4 поля структуры

Name5 dw ? ; имя 5 поля структуры

Name6 dW ? ; имя 6 поля структуры

DATE1 ENDS ; окончания данные СТРУКТУРА с именем Date1

.data ; директива определения данные

Car1 DATE1 <1,10,2,3,1,1> ; структура с именем Car1

Car2 DATE1 <2,-11,6,7,2,1> ; структура с именем Car2

Car3 DATE1 <3,12,10,11,3,1> ; структура с именем Car3

Car4 DATE1 <4,13,14,15,31,1> ; структура с именем Car4

\_res1 dq 0 ; операнд res1 размерностью 64 разряда; змінна результ

\_res2 dq 0 ; операнд res1 размерностью 64 разряда; змінна результ

\_res3 dq 0 ; операнд res1 размерностью 64 разряда; змінна результ

\_res4 dq 0 ; операнд res1 размерностью 64 разряда; змінна результ

\_res5 dq 0 ; операнд res1 размерностью 64 разряда; змінна результ

\_res6 dq 0 ; операнд res1 размерностью 64 разряда; змінна результ

params MSGBOXPARAMSA <>

;Текст для MessageBox

titl1 db "Лаб.4-1.1 СТРУКТУРИ. masm64",0

txt1 db "Завдання: Дана матриця 4 х 6. Визначити суму елементів кожного стовпця.", 10,10,

"Вхідний массив - вид матриці:",10,\

" |01|10|02|03|01|01|",10,\

" |01|10|02|03|01|01|",10,\

" |03|12|10|11|03|01|",10,\

" |04|13|14|15|31|01|",10,\

"-------------------------",10,\

"сума: |%d|%d|%d|%d|%d|%d| ",10,10,\

"Автор: Гряник Г.В., гр.КІТ-119Д",0

st2 db 20 dup(?),0 ;

.code

entry\_point proc

xor ebx,ebx

movzx ebx,Car1.Name1 ; заполнение нулями старшей части

add bx,Car2.Name1 ; bx := Car1.Name1 + Car2.Name1

add bx,Car3.Name1 ;

add bx,Car4.Name1 ;

movsxd r15,ebx

mov \_res1,r15 ;

movzx ebx,Car1.Name2 ; заполнение нулями старшей части

add bx,Car2.Name2 ; bx := Car1.Name2 + Car2.Name2

add bx,Car3.Name2 ;

add bx,Car4.Name2 ;

movsxd r15,ebx

mov \_res2,r15 ;

movzx ebx,Car1.Name3 ; заполнение нулями старшей части

add bx,Car2.Name3 ; bx := Car1.Name3 + Car2.Name3

add bx,Car3.Name3 ;

add bx,Car4.Name3 ;

movsxd r15,ebx

mov \_res3,r15 ;

movzx ebx,Car1.Name4 ; заполнение нулями старшей части

add bx,Car2.Name4 ; bx := Car1.Name4 + Car2.Name4

add bx,Car3.Name4 ;

add bx,Car4.Name4 ;

movsxd r15,ebx

mov \_res4,r15 ;

movzx ebx,Car1.Name5 ; заполнение нулями старшей части

add bx,Car2.Name5 ; bx := Car1.Name5 + Car2.Name5

add bx,Car3.Name5 ;

add bx,Car4.Name5 ;

movsxd r15,ebx

mov \_res5,r15 ;

movzx ebx,Car1.Name6 ; заполнение нулями старшей части

add bx,Car2.Name6 ; bx := Car1.Name6 + Car2.Name6

add bx,Car3.Name6 ;

add bx,Car4.Name6 ;

movsxd r15,ebx

mov \_res6,r15 ;

invoke wsprintf,ADDR st2,ADDR txt1,\_res1,\_res2,\_res3,\_res4,\_res5,\_res6 ; функция преобразования

mov params.cbSize,SIZEOF MSGBOXPARAMSA ; размер структуры

mov params.hwndOwner,0 ; дескриптор окна владельца

invoke GetModuleHandle,0 ; получение дескриптора программы

mov params.hInstance,rax ; сохранение дескриптора программы

lea rax,st2

mov params.lpszText,rax ; адрес сообщени¤

lea rax,titl1

mov params.lpszCaption,rax ; адрес заглави¤ окна

mov params.dwStyle,MB\_USERICON ; стиль окна

mov params.lpszIcon,IDI\_ICON ; ресурс значка

mov params.dwContextHelpId,0 ;контекст справки

mov params.lpfnMsgBoxCallback,0 ;

mov params.dwLanguageId,LANG\_NEUTRAL ; ¤зык сообщени¤

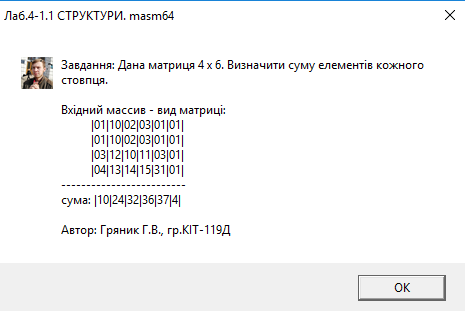
lea rcx,params

invoke MessageBoxIndirect

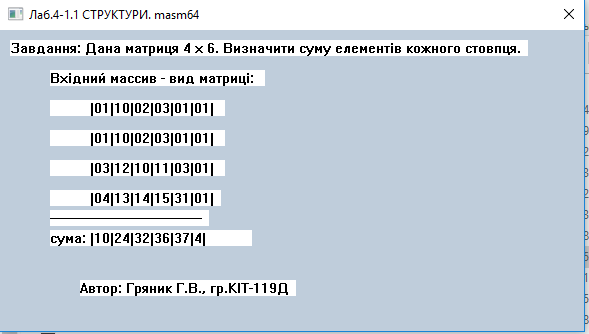
invoke ExitProcess, 0

entry\_point endp

end

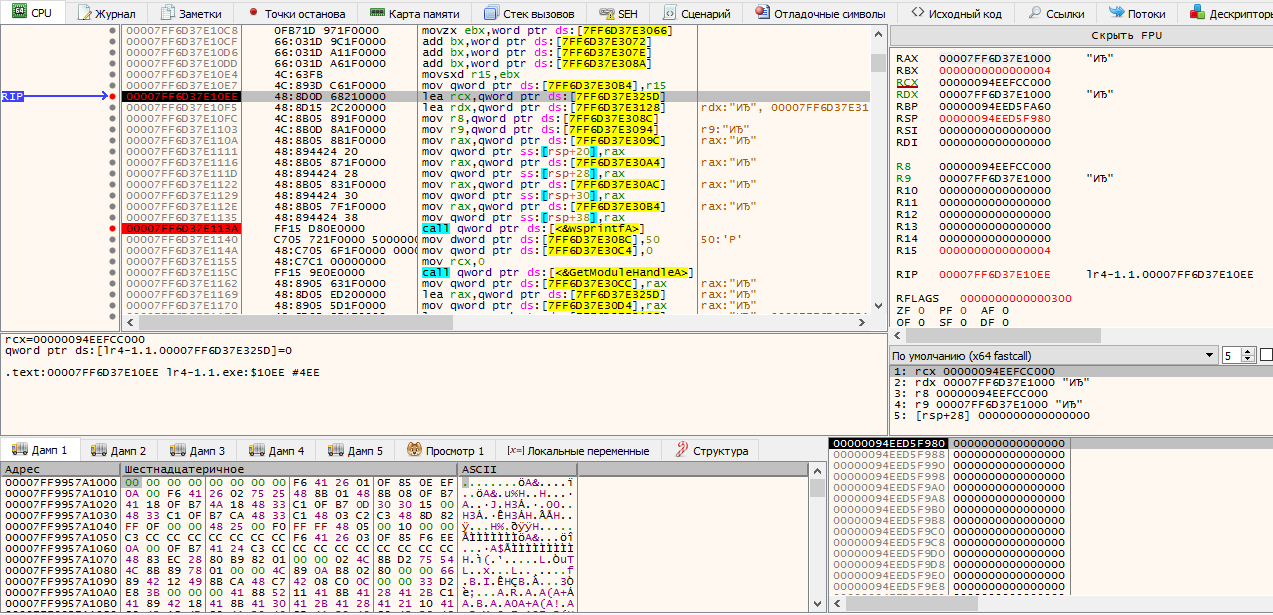


А)



Б)

**Рис 1 – результат виконання роботи програми** LR4-1.1



**Рис 2 – результат налагодження програми** LR4-1.1

Алгоритм: Програма послідовно звертається до кожного стовпця знаходить суму стовпця та записує результат у змінну. Після чого результат виводиться на екран за допомого MessageBoxIndirect та окремому window.

**Програма 2:** (LR4-1.2)

**include \masm64\include64\masm64rt.inc; библиотеки для подключения**

**IDI\_ICON EQU 1001**

**MSGBOXPARAMSA STRUCT**

**cbSize DWORD ?,?**

**hwndOwner QWORD ?**

**hInstance QWORD ?**

**lpszText QWORD ?**

**lpszCaption QWORD ?**

**dwStyle DWORD ?,?**

**lpszIcon QWORD ?**

**dwContextHelpId QWORD ?**

**lpfnMsgBoxCallback QWORD ?**

**dwLanguageId DWORD ?,?**

**MSGBOXPARAMSA ENDS**

**DATE2 STRUCT ; тип данных структура**

**\_firm db 22 dup(?) ; поле торговельнa діяльності комп'ютерної фірми**

**\_computer db 10 dup(?) ; поле Комп'ютер**

**\_modem db 10 dup(?) ; поле винт**

**\_price dq ? ; ціна (в доларах)**

**\_quantity dq ? ; кількість проданих одиниць**

**DATE2 ENDS ; окончания данных СТРУКТУРА**

**.data**

**ASUS DATE2 <"Комп'ютерна техніка", "ASUS" , "512 ГБ SSD",940,5> ; структура с именем ASUS**

**HP DATE2 <"Комп'ютерна техніка", "HP", "256 ГБ SSD",761,5> ; структура с именем HP**

**Acer DATE2 <"Комп'ютерна техніка", "Acer", "512 ГБ SSD",673,8> ; структура с именем ACER**

**MSI DATE2 <"Комп'ютерна техніка", "MSI", "256 ГБ SSD",672,7> ; структурас именем MSI**

**txt1 db"Задана послідовність структур.",10,**

**" Обчислити прибуток фірми (підсумовування 'кількості проданих одиниць помножену на ціну').",10,10,**

**"| торг. діяльності |Комп'ютер|винт |ціна|кількіс|",10,\**

**"|Комп'ютерна техніка| ASUS |512 ГБ SSD|940$| 5|",10,\**

**"|Комп'ютерна техніка| HP |256 ГБ SSD|761$| 5|",10,\**

**"|Комп'ютерна техніка| Acer |512 ГБ SSD|673$| 8|",10,\**

**"|Комп'ютерна техніка| MSI |256 ГБ SSD|672$| 7|",10,10,\**

**"Автор: Гряник Г.В., гр.КІТ-119Д",0**

**txt01 db"Задана послідовність структур.",10,0**

**txt2 db" Обчислити прибуток фірми (підсумовування 'кількості проданих одиниць помножену на ціну').",10,10,0**

**txt3 db"| торг. діяльності\_\_\_\_|Комп'ютер|винт\_\_\_\_\_\_|ціна|кількіс|",0**

**txt4 db"|Комп'ютерна техніка| ASUS\_\_\_\_|512 ГБ SSD|940$| 5|",0**

**txt5 db"|Комп'ютерна техніка| HP\_\_\_\_\_\_|256 ГБ SSD|761$| 5|",0**

**txt6 db"|Комп'ютерна техніка| Acer\_\_\_\_\_|512 ГБ SSD|673$| 8|",0**

**txt7 db"|Комп'ютерна техніка| MSI\_\_\_\_\_|256 ГБ SSD|672$| 7|",0**

**autor db"Автор: Гряник Г.В., гр.КІТ-119Д",0**

**hInstance dq ? ; дескриптор програми**

**hWnd dq ? ; дескриптор окна**

**hIcon dq ? ; дескриптор иконки**

**hCursor dq ? ; дескриптор курсора**

**sWid dq ? ; ширина монитора (колич. пикселей по x)**

**sHgt dq ? ; высота монитора (колич. пикселей по y)**

**classname db "template\_class",0**

**buf1 dq 5 dup(?),0**

**st1 db "Лаб.4-1.2 СТРУКТУРИ. masm64", 0**

**ifmt db "Прибуток фірми: %d$ ",0;Прибуток фірми**

**\_res1 dq 0**

**params MSGBOXPARAMSA <>**

**.code**

**entry\_point proc**

**mov rax, ASUS.\_price ; загрузка адреса первой строки структуры**

**mul ASUS.\_quantity ; ASUS.\_price\*ASUS.\_quantity**

**mov \_res1,rax; ; збереження даних**

**mov rax, HP.\_price ; загрузка адреса первой строки структуры**

**mul HP.\_quantity ; HP.\_price\*HP.\_quantity**

**add \_res1,rax; ; збереження даних**

**mov rax, Acer.\_price ; загрузка адреса первой строки структуры**

**mul Acer.\_quantity ; Acer.\_price\*Acer.\_quantity**

**add \_res1,rax; ; збереження даних**

**mov rax, MSI.\_price ; загрузка адреса первой строки структуры**

**mul MSI.\_quantity ; MSI.\_price\*MSI.\_quantity**

**add \_res1,rax; ; збереження даних**

**invoke wsprintf,ADDR buf1,ADDR ifmt,\_res1**

**mov params.cbSize,SIZEOF MSGBOXPARAMSA ; размер структуры**

**mov params.hwndOwner,0 ; дескриптор окна владельца**

**invoke GetModuleHandle,0 ; получение дескриптора программы**

**mov params.hInstance,rax ; сохранение дескриптора программы**

**lea rax,txt1**

**mov params.lpszText,rax ; адрес сообщени¤**

**lea rax,st1**

**mov params.lpszCaption,rax ; адрес заглави¤ окна**

**mov params.dwStyle,MB\_USERICON ; стиль окна**

**mov params.lpszIcon,IDI\_ICON ; ресурс значка**

**mov params.dwContextHelpId,0 ;контекст справки**

**mov params.lpfnMsgBoxCallback,0 ;**

**mov params.dwLanguageId,LANG\_NEUTRAL ; ¤зык сообщени¤**

**lea rcx,params**

**invoke MessageBoxIndirect**

**lea rax,buf1**

**mov params.lpszText,rax ; адрес сообщени¤**

**lea rax,st1**

**mov params.lpszCaption,rax ; адрес заглави¤ окна**

**lea rcx,params**

**invoke MessageBoxIndirect**

**mov hInstance,rv(GetModuleHandle,0) ; получение и сохранение дескрипторa програми**

**mov hIcon, rv(LoadIcon,IDI\_ICON,IDI\_ICON) ; загрузка и сохранение дескрипторa иконки**

**mov hCursor,rv(LoadCursor,0,IDC\_ARROW) ; загрузка курсора и сохранение**

**mov sWid,rv(GetSystemMetrics,SM\_CXSCREEN) ; получение кол. пикселей по х монитора**

**mov sHgt,rv(GetSystemMetrics,SM\_CYSCREEN) ; получение кол. пикселей по y монитора**

**call main**

**invoke ExitProcess,0**

**ret**

**entry\_point endp**

**main proc**

**LOCAL wc :WNDCLASSEX ; объявление локальных переменных**

**LOCAL lft :QWORD ; Лок. переменные содержатся в стеке**

**LOCAL top :QWORD ; и существуют только во время вып. проц.**

**LOCAL wid :QWORD**

**LOCAL hgt :QWORD**

**mov wc.cbSize,SIZEOF WNDCLASSEX ; колич. байтов структуры**

**mov wc.style,CS\_BYTEALIGNCLIENT or CS\_BYTEALIGNWINDOW ; стиль окна**

**mov wc.lpfnWndProc,ptr$(WndProc) ; адрес процедуры WndProc**

**mov wc.cbClsExtra,0 ; количество байтов дл¤ структуры класса**

**mov wc.cbWndExtra,0 ; количество байтов дл¤ структуры окна**

**mrm wc.hInstance,hInstance ; заполнение поля дескриптора в структуре**

**mrm wc.hIcon, hIcon ; хэндл иконки**

**mrm wc.hCursor,hCursor ; хэндл курсора**

**mrm wc.hbrBackground,4 ; цвет окна**

**mov wc.lpszMenuName,0 ; заполнение пол¤ в структуре с именем ресурса меню**

**mov wc.lpszClassName,ptr$(classname) ; им¤ класса**

**mrm wc.hIconSm,hIcon**

**invoke RegisterClassEx,ADDR wc ; регистрация класса окна**

**mov wid, 750 ; ширина пользовательского окна в пиксел¤х**

**mov hgt, 340 ; высота пользовательского окна в пиксел¤х**

**mov rax,sWid ; колич. пикселей монитора по x**

**sub rax,wid ; дельта Х = Х(монитора) - х(окна пользовател¤)**

**shr rax,1 ; получение середины Х**

**mov lft,rax ;**

**mov rax, sHgt ; колич. пикселей монитора по y**

**sub rax, hgt ;**

**shr rax, 1 ;**

**mov top, rax ;**

**; ---------------------------------**

**; centre window at calculated sizes**

**invoke CreateWindowEx,WS\_EX\_LEFT or WS\_EX\_ACCEPTFILES, \**

**ADDR classname,ADDR st1 , \**

**WS\_OVERLAPPED or WS\_VISIBLE or WS\_SYSMENU,\**

**lft,top,wid,hgt,0,0,hInstance,0**

**mov hWnd,rax ; сохранение дескриптора окна**

**call msgloop**

**ret**

**main endp**

**msgloop proc**

**LOCAL msg :MSG**

**LOCAL pmsg :QWORD**

**mov pmsg,ptr$(msg) ; получение адреса структуры сообщения**

**jmp gmsg ; jump directly to GetMessage()**

**mloop:**

**invoke TranslateMessage,pmsg**

**invoke DispatchMessage,pmsg ; отправка на обслуживание к WndProc**

**gmsg:**

**test rax, rv(GetMessage,pmsg,0,0,0) ; пока GetMessage не вернет ноль**

**jnz mloop**

**ret**

**msgloop endp**

**WndProc proc hWin:QWORD,uMsg:QWORD,wParam:QWORD,lParam:QWORD**

**LOCAL hdc:HDC ; резервирование стека дл¤ дескриптора окна**

**LOCAL ps:PAINTSTRUCT ; дл¤ структуры PAINTSTRUCT**

**LOCAL rect:RECT ; дл¤ структуры координат RECT**

**.switch uMsg**

**.case WM\_DESTROY ; если есть сообщение про уничтожение окна**

**invoke PostQuitMessage,NULL**

**.case WM\_PAINT ; если есть смс о перерисовании**

**invoke BeginPaint,hWnd, ADDR ps ; вызов подготовительной процедуры**

**mov hdc,rax ; сохранение контекста**

**invoke GetClientRect,hWnd,ADDR rect ; занесение в структуру rect**

**; характеристик окна**

**invoke DrawText, hdc,ADDR txt1,0, ADDR rect, \ ; рисование текста**

**DT\_SINGLELINE or DT\_CENTER or DT\_VCENTER**

**invoke TextOut,hdc,10,10,ADDR txt01,32**

**invoke TextOut,hdc,50,40,ADDR txt2,92**

**invoke TextOut,hdc,50,70,ADDR txt3,57**

**invoke TextOut,hdc,50,90,ADDR txt4,55**

**invoke TextOut,hdc,50,110,ADDR txt5,55**

**invoke TextOut,hdc,50,130,ADDR txt6,56**

**invoke TextOut,hdc,50,150,ADDR txt7,55**

**invoke TextOut,hdc,80,250,ADDR autor,31**

**invoke TextOut,hdc,80,200,ADDR buf1,25**

**invoke EndPaint,hWnd, ADDR ps ; завершение рисования**

**.endsw**

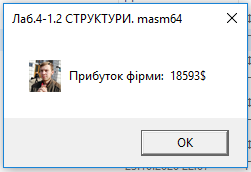
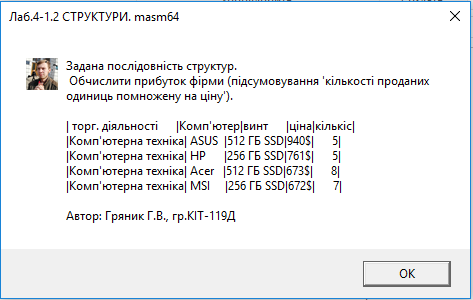
**invoke DefWindowProc,hWin,uMsg,wParam,lParam**

**ret**

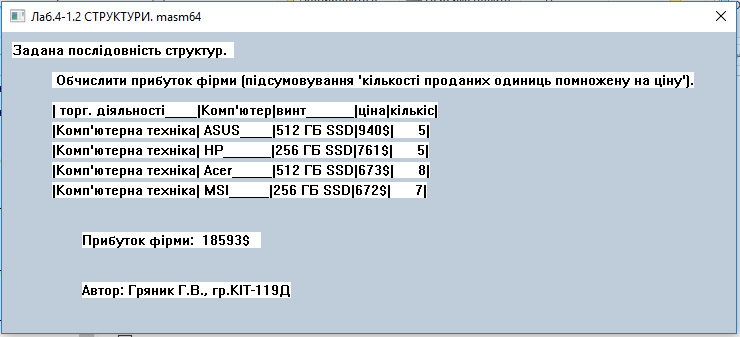
**WndProc endp**

**end**

**Результат**

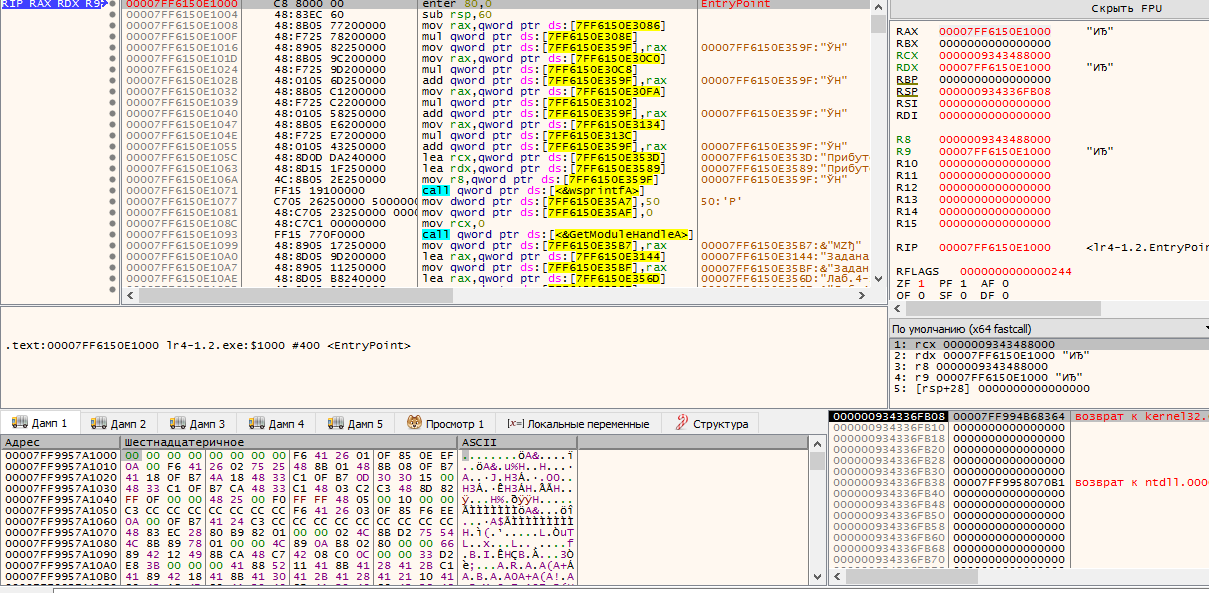


А) Б)



В)

**Рис 3 – результат виконання роботи програми** LR4-1.2



**Рис 4 – результат налагодження програми** LR3-1.2

Алгоритм: Програма звертається до структури та перемножує два поля :ціна ,кількість. Після чого це число додається до результату. Далі результат виводиться на екран за допомого MessageBoxIndirect та окремому window.

**Програма 3:** (LR4-2)

include \masm64\include64\masm64rt.inc ; библиотеки

IDI\_ICON EQU 1001

MSGBOXPARAMSA STRUCT

cbSize DWORD ?,?

hwndOwner QWORD ?

hInstance QWORD ?

lpszText QWORD ?

lpszCaption QWORD ?

dwStyle DWORD ?,?

lpszIcon QWORD ?

dwContextHelpId QWORD ?

lpfnMsgBoxCallback QWORD ?

dwLanguageId DWORD ?,?

MSGBOXPARAMSA ENDS

mMul macro x, y ;; макрос с именем mSub

fld x ;; загрузка x

fld y ;; загрузка y

fmul ;; x\*y

endm ;; окончание макроса

.data ;

a real4 2.7 ; операнд а1 размерностью 64 разряда константа 2

b real4 5.2 ; операнд b1 размерностью 64 разряда ;змзмінна d

const1 real4 1.1

const2 real4 3.0

res1 dd 0,0; операнд res1 размерностью 64 разряда; змінна результ

;Текст для MessageBox

title1 db "Лаб.4-2.1 макроси. masm64",0

txt1 db "Вираз: (1,1ab – 3)/ab",10,

"Змінні: a=2.7, b=5.2",10,10,

"Результат: %d",10,"Адрес змінної в памяти: %ph",10,10,

"Автор: Гряник Г.В., гр.КІТ-119Д",0

buf1 dq 25 dup(0),0

buf2 dq 25 dup(0),0

;Текст для window

txt01 db "Вираз: (1,1ab – 3)/ab",0

txt02 db"Змінні: a=2.7, b=5.2",0

txt03 db"Результат: %d "

txt04 db"Адрес змінної в памяти: %ph",0

autor db"Автор: Гряник Г.В., гр.КІТ-119Д",0

params MSGBOXPARAMSA <>

hInstance dq ? ; дескриптор програми

hWnd dq ? ; дескриптор окна

hIcon dq ? ; дескриптор иконки

hCursor dq ? ; дескриптор курсора

sWid dq ? ; ширина монитора (колич. пикселей по x)

sHgt dq ? ; высота монитора (колич. пикселей по y)

classname db "template\_class",0

caption db "“екст в Windows-окне",0

Hello db "ѕервое сообщение пользовател¤ в windows окне!!!",10,

" masm64",0

.code;cекция кода

entry\_point proc

finit ; инициализация FPU

;mov ecx,4;

mMul [a],[b]

fmul const1

fsub const2

fdiv a

fmul b

fisttp res1

;movsxd rax,res1

;Створення MessageBox

invoke wsprintf,ADDR buf1,ADDR txt1,res1,ADDR res1

invoke wsprintf,ADDR buf2,ADDR txt03,res1

invoke wsprintf,ADDR txt03,ADDR txt04,ADDR res1

invoke MessageBox,0,ADDR buf1,ADDR title1,MB\_ICONINFORMATION

mov params.cbSize,SIZEOF MSGBOXPARAMSA ; размер структуры

mov params.hwndOwner,0 ; дескриптор окна владельца

invoke GetModuleHandle,0 ; получение дескриптора программы

mov params.hInstance,rax ; сохранение дескриптора программы

lea rax,buf1

mov params.lpszText,rax ; адрес сообщения

lea rax,title1

mov params.lpszCaption,rax ; адрес заглавия окна

mov params.dwStyle,MB\_USERICON ; стиль окна

mov params.lpszIcon,IDI\_ICON ; ресурс значка

mov params.dwContextHelpId,0 ;контекст справки

mov params.lpfnMsgBoxCallback,0 ;

mov params.dwLanguageId,LANG\_NEUTRAL ; язык сообщения

lea rcx,params

invoke MessageBoxIndirect

mov hInstance,rv(GetModuleHandle,0) ; получение и сохранение дескрипторa програми

mov hIcon, rv(LoadIcon,IDI\_ICON,IDI\_ICON) ; загрузка и сохранение дескрипторa иконки

mov hCursor,rv(LoadCursor,0,IDC\_ARROW) ; загрузка курсора и сохранение

mov sWid,rv(GetSystemMetrics,SM\_CXSCREEN) ; получение кол. пикселей по х монитора

mov sHgt,rv(GetSystemMetrics,SM\_CYSCREEN) ; получение кол. пикселей по y монитора

call main

invoke ExitProcess,0

ret

entry\_point endp

main proc

LOCAL wc :WNDCLASSEX ; объявление локальных переменных

LOCAL lft :QWORD ; Лок. переменные содержатся в стеке

LOCAL top :QWORD ; и существуют только во время вып. проц.

LOCAL wid :QWORD

LOCAL hgt :QWORD

mov wc.cbSize,SIZEOF WNDCLASSEX ; колич. байтов структуры

mov wc.style,CS\_BYTEALIGNCLIENT or CS\_BYTEALIGNWINDOW ; стиль окна

mov wc.lpfnWndProc,ptr$(WndProc) ; адрес процедуры WndProc

mov wc.cbClsExtra,0 ; количество байтов дл¤ структуры класса

mov wc.cbWndExtra,0 ; количество байтов дл¤ структуры окна

mrm wc.hInstance,hInstance ; заполнение поля дескриптора в структуре

mrm wc.hIcon, hIcon ; хэндл иконки

mrm wc.hCursor,hCursor ; хэндл курсора

mrm wc.hbrBackground,4 ; цвет окна

mov wc.lpszMenuName,0 ; заполнение пол¤ в структуре с именем ресурса меню

mov wc.lpszClassName,ptr$(classname) ; им¤ класса

mrm wc.hIconSm,hIcon

invoke RegisterClassEx,ADDR wc ; регистрация класса окна

mov wid, 450 ; ширина пользовательского окна в пиксел¤х

mov hgt, 190 ; высота пользовательского окна в пиксел¤х

mov rax,sWid ; колич. пикселей монитора по x

sub rax,wid ; дельта Х = Х(монитора) - х(окна пользовател¤)

shr rax,1 ; получение середины Х

mov lft,rax ;

mov rax, sHgt ; колич. пикселей монитора по y

sub rax, hgt ;

shr rax, 1 ;

mov top, rax ;

; ---------------------------------

; centre window at calculated sizes

invoke CreateWindowEx,WS\_EX\_LEFT or WS\_EX\_ACCEPTFILES, \

ADDR classname,ADDR title1 , \

WS\_OVERLAPPED or WS\_VISIBLE or WS\_SYSMENU,\

lft,top,wid,hgt,0,0,hInstance,0

mov hWnd,rax ; сохранение дескриптора окна

call msgloop

ret

main endp

msgloop proc

LOCAL msg :MSG

LOCAL pmsg :QWORD

mov pmsg,ptr$(msg) ; получение адреса структуры сообщения

jmp gmsg ; jump directly to GetMessage()

mloop:

invoke TranslateMessage,pmsg

invoke DispatchMessage,pmsg ; отправка на обслуживание к WndProc

gmsg:

test rax, rv(GetMessage,pmsg,0,0,0) ; пока GetMessage не вернет ноль

jnz mloop

ret

msgloop endp

WndProc proc hWin:QWORD,uMsg:QWORD,wParam:QWORD,lParam:QWORD

LOCAL hdc:HDC ; резервирование стека дл¤ дескриптора окна

LOCAL ps:PAINTSTRUCT ; дл¤ структуры PAINTSTRUCT

LOCAL rect:RECT ; дл¤ структуры координат RECT

.switch uMsg

.case WM\_DESTROY ; если есть сообщение про уничтожение окна

invoke PostQuitMessage,NULL

.case WM\_PAINT ; если есть смс о перерисовании

invoke BeginPaint,hWnd, ADDR ps ; вызов подготовительной процедуры

mov hdc,rax ; сохранение контекста

invoke GetClientRect,hWnd,ADDR rect ; занесение в структуру rect

; характеристик окна

invoke DrawText, hdc,ADDR txt1,0, ADDR rect, \ ; рисование текста

DT\_SINGLELINE or DT\_CENTER or DT\_VCENTER

invoke TextOut,hdc,10,10,ADDR txt01,22

invoke TextOut,hdc,50,40,ADDR txt02,21

invoke TextOut,hdc,50,70,ADDR buf2,13

invoke TextOut,hdc,50,90,ADDR txt03,41

invoke TextOut,hdc,80,130,ADDR autor,31

invoke EndPaint,hWnd, ADDR ps ; завершение рисования

.endsw

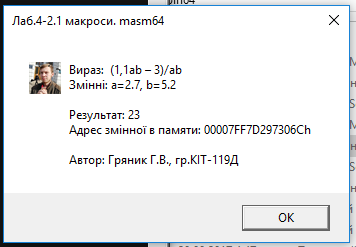
invoke DefWindowProc,hWin,uMsg,wParam,lParam

ret

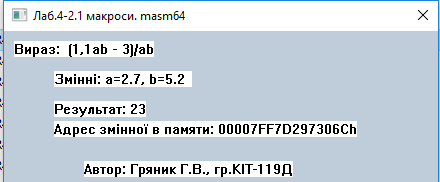
WndProc endp

end

**Результат**

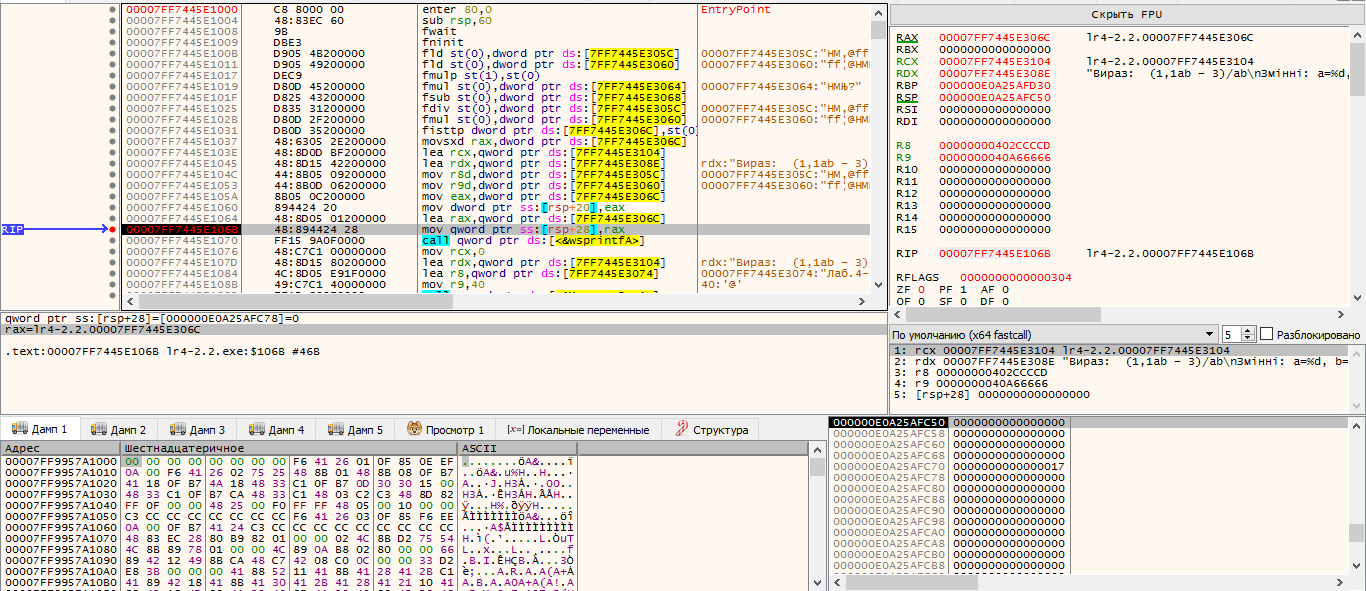


А)



Б)

**Рис 5 – результат виконання роботи програми** LR4-2



**Рис 6 – результат налагодження програми** LR3-2

**Алгоритм:** Програма вирішує заданий завданням вираз вирішується за тією логікою ,що і звичайний за винятком того ,що крок множення a\*b робиться за допомогою макросу.

Далі результат виводиться на екран за допомого MessageBoxIndirect та окремому window.

**Висновок:** Під час лабораторної роботи було набуто практичних навичок з роботи mas64, а саме , навичок складання, налагодження та виконання програм з використанням макросів, написаних мовою ассемблера для співпроцесора платформи х64 в середовищі masm64. налагодження та виконання програм з використанням структур, використання допомого MessageBoxIndirect та окремоме window.