**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 №7**

з дисципліни  
«Технології розроблення системних програм»

на тему  
«Модульне програмування. Використання процедур.»

Виконав: Перевірив:

студент групи ІП-84 доц. кафедри ОТ  
Ковалишин Олег Юрійович Павлов В. Г.  
номер залікової книжки: 8410

Київ 2020

**Мета роботи**

Вивчення прийомів модульного програмування, методів звернення до процедур і передачі в них параметрів.

**Порядок виконання роботи**

1. Вивчити методи звернення до процедур і передачі в них параметрів.
2. Для свого варіанту індивідуального завдання до лабораторної роботи 6 розробити програму на мові Асемблер, в якій використовувати три процедури з різними способами передачі параметрів:
   * через регістри;
   * через стек;
   * за допомогою директив EXTRN та PUBLIC.
3. Для цього чисельник дробу зі свого варіанту індивідуального завдання до лабораторної роботи 6 розділити на два доданка, з яких для першого застосувати передачу параметрів і результату через регістри, а для другого – через стек. Для знаменника використовувати метод оголошення загальних змінних директивами public і extern. Виведення результату\* виконати в основній програмі.
4. Розрахунки (п. 3) повторити в програмі для 5 значень змінних\*\*, причому всі вхідні значення задати дійсними числами у вигляді одновимірних масивів.
5. Для перевірки правильності виконання розрахунків і результатів, що виводяться, заздалегідь виконати контрольні розрахунки, які повинні охоплювати різноманітні сполучення вхідних даних, на які програма повинна надавати вірну відповідь. Проміжні і остаточні результати контрольних розрахунків привести в звіті по лабораторній роботі. Точність розрахунків така ж, як і у лаб. роботі 6.
6. Виконати відладку програми шляхом порівняння розрахованих програмою результатів з контрольними прикладами. Лістинг розробленої програми і скріншоти розрахунків по всіх контрольних прикладах привести в звіті по лабораторній роботі.
7. У протоколі по лабораторній роботі для першого і другого способів передачі параметрів поруч з відповідними командами у лістингу відобразити в графічному вигляді стани стека при зверненні до процедур, виконання у них команд та повернення з процедур до основної програми.
8. Зробити висновки по лабораторній роботі.

**Номер у списку групи:** 8

**Варіянт:** (tg(c) - d \* 23) / (2 \* b - a)

**Контрольні приклади:**

1. **a = 1,23, b = -2,34, c = 3,14159, d = 4,56**  
   (tg(3,14159) - 4,56 \* 23) / (2 \* (-2,34) - 1,23)  
   4,56 \* 23 = 104,880000000000000000000000000000000  
   tg(3,14159) = -0,0000026535897932446910952  
   tg(3,14159) - 4.56 \* 23 = -104,88000265358979336207  
   2 \* (-2,34) = -4,680000000000000000000000000000000000000000  
   2 \* (-2,34) - 1,23 = -5,9100000000000000000000000000000000000000  
   (tg(3,14159) - 4,56 \* 23) / (2 \* (-2,34) - 1,23) = -104,88000265358979336207 / -5,91 = 17,74619334240098026431
2. **a = 10,42, b = -2,39, c = 12,5, d = -1,43**  
   (tg(12,5) - (-1,43) \* 23) / (2 \* (-2,39) - 10,42)  
   -1,43 \* 23 = -32,89  
   tg(12,5) = -0,06646824186327419610  
   tg(12,5) - (-1,43) \* 23 = 32,8235317581367258039  
   2 \* (-2,39) = -4,7800000000000000000000000000000000000000000000  
   2 \* (-2,39) - 10,42 = -15,200000000000000000000000000000000000000  
   (tg(12,5) - (-1,43) \* 23) / (2 \* (-2,39) - 10,42) = 32,8235317581367258039 / (-15,2) = -2,15944287882478459236
3. **a = -110,54, b = 34,13, c = 0,00035, d = 0,51**  
   (tg(0,00035) - 0,51 \* 23)/(2 \* 34,13 - (-110,54))  
   0,51 \* 23 = 11,73  
   tg(0,00035) = 0,00035000001429166737  
   tg(0,00035) - 0,51 \* 23 = -11,72964999998570833263  
   2 \* 34,13 = 68,260000000000000000000000000000000000000000000  
   2 \* 34,13 - (-110,54) = 178,8‬0000000000000000000000000000000000  
   (tg(0,00035) - 0,51 \* 23)/(2 \* 34,13 - (-110,54)) = -0,06560206935115049403
4. **a = 12,09, b = 6,045, c = -9, d = -1,2**  
   (tg(-9) - (-1,2) \* 23) / (2 \* (6,045) - 12,09)  
   (-1,2) \* 23 = -27,6000000000000000000000000000000000000  
   tg(-9) = 0,45231565944180984059  
   (tg(-9) - (-1,2) \* 23) = 28,05231565944180984059  
   2 \* 6,045 = 12,090000000000000000000000000000000000000000  
   2 \* 6,045 - 12,09 =0,00000000000000000000000000000000000  
   (tg(-9) - (-1,2) \* 23) / (2\* 6,045 - 12,09) = ∞ (Ділення на нуль)
5. **a = 1209, b = 120, c = 1,0e+20, d = 928,2**  
   (tg(1,0e+20) - 928.2 \* 23) / (2 \* 120 - 1209)  
   928,2 \* 23 = 21348,60000000000000000000000000000  
   tg(1,0e+20) = -0,844602463019884254184  
   (tg(1,0e+20) - 928.2 \* 23) = -21349,44460246301988425‬  
   2 \* 120 = 240‬,000000000000000000000000000000000000  
   2 \* 120 - 1209 = -969,00000000000000000000000000000  
   (tg(1,0e+20) - 928.2 \* 23) / (2 \* 120 - 1209) = 22,032450570137275422342621259

**Вихідний код**

**Lab7.asm**

INCLUDE \masm32\include\masm32rt.inc

INCLUDE \masm32\include\Fpu.inc

INCLUDELIB \masm32\lib\Fpu.lib

EXTERN tan:PROTO

EXTERN den:PROTO

EXTERN extrn\_res:QWORD

EXTERN mult:PROTO

PUBLIC extrn\_a, extrn\_b

save MACRO buff, num, error

.IF error == 1

INVOKE crt\_sprintf, ADDR buff, ADDR msg\_err\_1

.ELSEIF error == 2

INVOKE crt\_sprintf, ADDR buff, ADDR msg\_err\_2

.ELSE

INVOKE crt\_sprintf, ADDR buff, ADDR msg\_num, num

.ENDIF

ENDM

copyDQ MACRO in, out ; copies dq value from in to out

MOV ESI, DWORD ptr in

MOV DWORD ptr out, ESI

MOV ESI, DWORD ptr in[4]

MOV DWORD ptr out[4], ESI

ENDM

; (tg(c) - d \* 23) / (2 \* b - a)

calc MACRO a, b, c\_, d, res

MOV error\_status, 0

; -23 \* d

**PUSH DWORD PTR d[4]**

**PUSH DWORD PTR d**

**PUSH offset temp2**

**CALL mult**

save buff\_temp\_1, temp2, error\_status

; tg(c)

LEA ESI, c\_

LEA ECX, temp1

**CALL tan**

MOV error\_status, EBX

save buff\_temp\_2, temp1, error\_status

finit

fld temp1 ; st(0) = tan(c)

fld temp2 ; st(0) = -23d, st(1) = tan(c)

fadd st, st(1) ; st(0) = tan(c) - d \* 23

fstp temp1

save buff\_temp\_3, temp1, error\_status

; 2 \* b - a

copyDQ a, extrn\_a

copyDQ b, extrn\_b

**CALL den**

save buff\_temp\_4, extrn\_res[8], error\_status

save buff\_temp\_5, extrn\_res, error\_status

.IF error\_status == 0

MOV error\_status, ESI

.ENDIF

.IF error\_status == 0

finit

fld temp1

fld extrn\_res

fdivp st(1), st

fstp res

.ENDIF

save buff\_temp\_6, res, error\_status

ENDM

getExpression MACRO i, buff

calc a[i\*8], b[i\*8], c\_[i\*8], d[i\*8], res[i\*8]

save buff\_temp\_7, res[i\*8], error\_status

INVOKE crt\_sprintf, buff, ADDR msg\_final, a[i\*8], b[i\*8], c\_[i\*8], d[i\*8], ADDR buff\_temp\_7,

ADDR buff\_temp\_1,

ADDR buff\_temp\_2,

ADDR buff\_temp\_3,

ADDR buff\_temp\_4,

ADDR buff\_temp\_5,

ADDR buff\_temp\_6

ENDM

.data

; STRINGS

msg\_title DB "Лабораторна робота 6", 0

msg\_last\_final DB "Результати обчислень:", 10,

"1. %s", 10,

"2. %s", 10,

"3. %s", 10,

"4. %s", 10,

"5. %s", 0

msg\_final DB "a = %.18f, b = %.18f, c = %.18f, d = %.18f", 10, "res = %s", 10, 0, ; uncomment 0 for shorter output

" -23 \* d = %s", 10,

" tg(c) = %s", 10,

" tg(c) - d \* 23 = %s", 10,

" b \* 2 = %s", 10,

" b \* 2 - a = %s", 10,

" (tg(c) - d \* 23) / (2 \* b - a) = %s", 0

msg\_num DB "%.18f", 0

msg\_err\_1 DB "Division by zero.", 0

msg\_err\_2 DB "tan: invalid argument.", 0

; BUFFERS

buff\_last\_final DB 40960 DUP (0)

buff\_final DB 10240 DUP (0)

buff\_final\_2 DB 10240 DUP (0)

buff\_final\_3 DB 10240 DUP (0)

buff\_final\_4 DB 10240 DUP (0)

buff\_final\_5 DB 10240 DUP (0)

buff\_size = $ - buff\_final\_5

buff\_temp\_1 DB 01280 DUP (0)

buff\_temp\_2 DB 01280 DUP (0)

buff\_temp\_3 DB 01280 DUP (0)

buff\_temp\_4 DB 01280 DUP (0)

buff\_temp\_5 DB 01280 DUP (0)

buff\_temp\_6 DB 01280 DUP (0)

buff\_temp\_7 DB 01280 DUP (0)

current\_buff\_ADDR DD 0

; VARIABLES

res DQ 5 DUP (0)

a DQ 1.23 , 10.42, -110.54, 12.09 , 1209.0

b DQ -2.34 , -2.39, 34.13 , 6.045 , 120.0

c\_ DQ 3.14159, 12.50, 0.00035, -9.0 , 1.0e20

d DQ 4.56 , -1.43, 0.51 , -1.2 , 928.2

temp1 DQ 0.0

temp2 DQ 0.0

error\_status DD 0

; PUBLICS

extrn\_a DQ 0.0

extrn\_b DQ 0.0

.code

start:

MOV EDI, 0

MOV current\_buff\_ADDR, offset buff\_final

hereWeGoAgain:

getExpression EDI, current\_buff\_ADDR

ADD current\_buff\_ADDR, buff\_size

INC EDI

CMP EDI, 5

JB hereWeGoAgain

INVOKE crt\_sprintf, ADDR buff\_last\_final, ADDR msg\_last\_final,

ADDR buff\_final,

ADDR buff\_final\_2,

ADDR buff\_final\_3,

ADDR buff\_final\_4,

ADDR buff\_final\_5

INVOKE MessageBox, 0, ADDR buff\_last\_final, ADDR msg\_title, MB\_OK

INVOKE ExitProcess, 0

END start

**Lab7-1.asm**

INCLUDE \masm32\include\masm32rt.inc

PUBLIC tan

.data

max DQ 922337203685477580.0

min DQ -922337203685477580.0

.code

; calculates tan(c)

; Input:

; ESI - address of DQ c

; EBX - number of error

; ECX - address of result DT variable

; Output: [ECX] - tan(c)

tan PROC

XOR EBX, EBX

finit

fld QWORD PTR [ESI] ; st(0) = c

fcom max

fstsw AX

SAHF

JNC tangens\_error

fcom min

fstsw AX

SAHF

JC tangens\_error

fptan ; st(0) = 1, st(1) = tan(c)

fdivp st(1), st ; st(0) = tan(c)

fstp QWORD PTR [ECX]

Зі стеку вийнято адресу повернення у основну програму

fin:

**RET**

tangens\_error:

MOV EBX, 2

JMP fin

tan ENDP

END

**Lab7-2.asm**

INCLUDE \masm32\include\masm32rt.inc

PUBLIC mult

.data

const DQ -23.0

.code

; Calculates -23 \* d

; Input:

; [EBP + 8] - address of result

; [EBP + 12] - first 32 bits of d

; [EBP + 16] - last 32 bits of d

;Output:

; QWORD PTR [EBP + 8] - -23d

mult PROC

**PUSH EBP**

MOV EBP, ESP

**PUSH EAX**

MOV EAX, [EBP + 8]

finit

fld const

fld QWORD PTR [EBP + 12]

fmul

fstp QWORD PTR [EAX]

**POP EAX**

**POP EBP**

**RET 12**

mult ENDP

END

**Lab7-3.asm**

INCLUDE \masm32\include\masm32rt.inc

PUBLIC den, extrn\_res

EXTERN extrn\_a:QWORD, extrn\_b:QWORD

.data

extrn\_res DQ 0.0, 0.0

zero DQ 0.0

.code

; calculates 2 \* b - a and checks if it's 0

Зі стеку вийнято адресу повернення у основну програму та видалено 12 байт.

; INPUT:

; extrn\_a - QWORD

; extrn\_b - QWORD

; OUTPUT:

; [extrn\_res + 0] - QWORD result of calculations 2 \* b - a

; [extrn\_res + 8] - QWORD result of calculations 2 \* b

; ESI = 1 if 2 \* b - a = 0

den PROC

XOR ESI, ESI

finit

fld1

fld1

faddp st(1), st

fld extrn\_b

fmulp st(1), st

fst extrn\_res[8]

fld extrn\_a

fsubp st(1), st

fcom zero

fstsw AX

SAHF

JZ division\_by\_zero

fin:

Зі стеку вийнято адресу повернення у основну програму

fstp extrn\_res

**RET**

division\_by\_zero:

MOV ESI, 1

JMP fin

den ENDP

END

**.bat-file**

@echo off

    set filename="7-8-IP84-Ковалишин"

    set exec\_filename="7-8-IP84-Ковалишин.exe"

    if exist "%filename%.obj" del "%filename%.obj"

    if exist "%filename%-1.obj" del "%filename%-1.obj"

    if exist "%filename%-2.obj" del "%filename%-2.obj"

    if exist "%filename%-3.obj" del "%filename%-3.obj"

    if exist "%filename%.exe" del "%filename%.exe"

    \masm32\bin\ml /c /coff "%filename%.asm"

    \masm32\bin\ml /c /coff "%filename%-1.asm"

    \masm32\bin\ml /c /coff "%filename%-2.asm"

    \masm32\bin\ml /c /coff "%filename%-3.asm"

    if errorlevel 1 goto errasm

    \masm32\bin\Link.exe /SUBSYSTEM:WINDOWS /out:%exec\_filename% "%filename%.obj" "%filename%-1.obj" "%filename%-2.obj" "%filename%-3.obj"

    if errorlevel 1 goto errlink

    dir "%filename%.\*"

    goto TheEnd

  :errlink

    echo \_

    echo Link error

    goto TheEnd

  :errasm

    echo \_

    echo Assembly Error

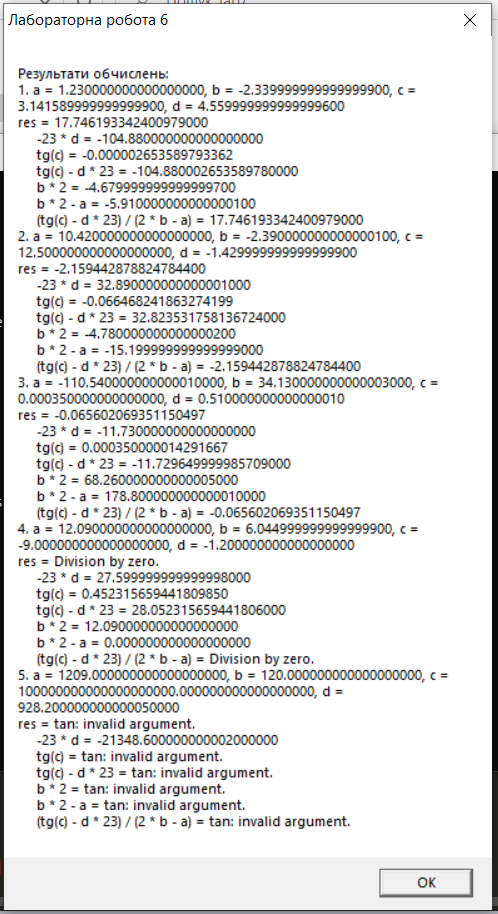
    goto TheEnd

  :TheEnd

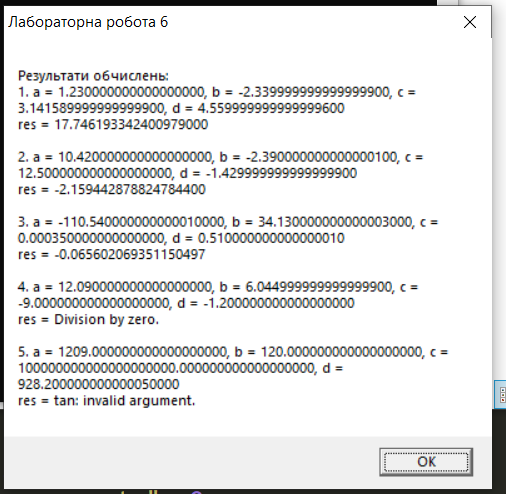
%filename%.exe

pause

**Скріншоти**Версія для відлагодження:

****

Скорочена версія:



**Примітка.** Приклад 5 виводить помилку “tan: invalid argument” через те, що за специфікацією функції FTAN вона повертає правильний результат лише в тому випадку, якщо модуль вхідного значення |x|<2^63.

**Висновок**

Вивчено прийоми модульного програмування, методи звернення до процедур і передачі в них параметрів. Розроблено програму, що рахує контрольні приклади за варіантом і виводить їх на екран у віконному інтерфейсі. Під час обрахунку використано процедури з передачею аргументів через регістри, стек і за допомогою директив **EXTERN**/**PUBLIC**. Заздалегіть виконано контрольні обчислення; виконано відлагодження програми шляхом порівняння результатів контрольних обчислень та виводу програми. Досліджено стани стеку під час звернення до процедур, виконання в них команд та виходу з процедур.