**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  
Голубов Іван Олегович Павлов В. Г.  
номер залікової книжки: 8404

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.

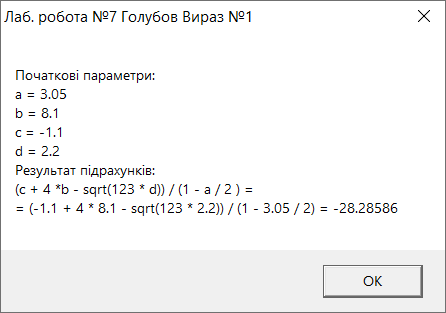
5. Для перевірки правильності виконання розрахунків і результатів, що виводяться, заздалегідь виконати контрольні розрахунки, які повинні охоплювати різноманітні сполучення вхідних даних, на які програма повинна надавати вірну відповідь\*\*\*. Проміжні і остаточні результати контрольних розрахунків привести в звіті по лабораторній роботі. Точність розрахунків така ж, як і у лаб. роботі 6.

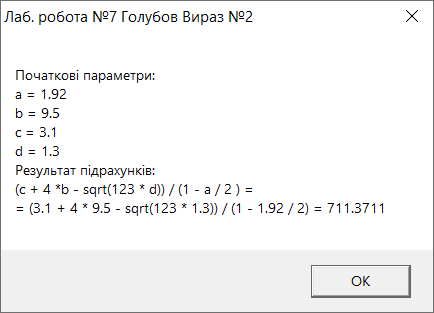
6. Виконати відладку програми шляхом порівняння розрахованих програмою результатів з контрольними прикладами. Лістинг розробленої програми і скріншоти розрахунків по всіх контрольних прикладах привести в звіті по лабораторній роботі.

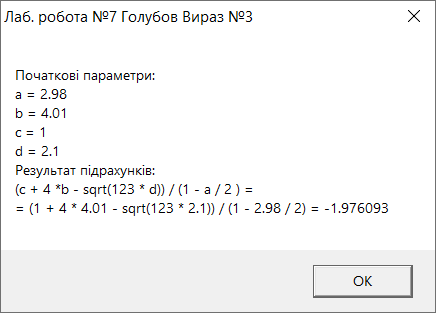
7. У протоколі по лабораторній роботі для першого і другого способів передачі параметрів поруч з відповідними командами у лістингу відобразити в графічному вигляді стани стека при зверненні до процедур, виконання у них команд та повернення з процедур до основної програми.

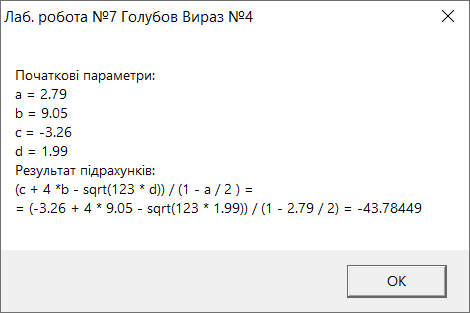
8. Зробити висновки по лабораторній роботі.

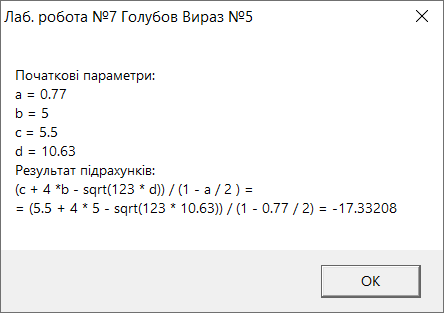
**Скріншоти**



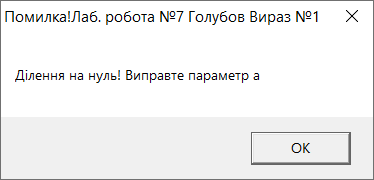




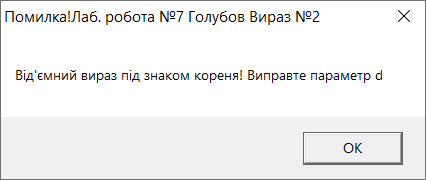




**Скріншот роботи програми при а=2**



**Скріншот роботи програми при d<0**



**Лістинг програми**

**7-03-IP84-Golubov.asm**

.686

.model flat,stdcall

option casemap:none

include \masm32\include\windows.inc ;MB\_OK,NULL

include \masm32\include\kernel32.inc ;ExitProcess

include \masm32\include\user32.inc ;wsprintf, MessageBox

include \masm32\include\masm32.inc ;FloatToStr

include \masm32\include\fpu.inc ;FpuFLtoA

includelib \masm32\lib\kernel32.lib ;ExitProcess

includelib \masm32\lib\user32.lib ;MessageBox

includelib \masm32\lib\masm32.lib ;FloatToStr

includelib \masm32\lib\fpu.lib ;FpuFLtoA

;розкоментувати для виводу проміжних результатів

extrn third@0:near

;public var\_a, var3, temp6, temp7

public var\_a, var3; закоментувати для виводу проміжних результатів

.data

var\_a DQ ?;

var\_b DQ ?;

var\_c DQ ?;

var\_d DQ ?;

Res DQ ?;

printRes DQ ?;

temp1 DT ?;

temp2 DT ?;

temp3 DT ?;

temp4 DT ?;

temp5 DT ?;

temp6 DT ?;

temp7 DT ?;

RES DT ?;

it DW 1;

iter DW 0;

arr\_a DQ 3.05, 1.92, 2.98, 2.79, 0.77;

arr\_b DQ 8.10, 9.50, 4.01, 9.05, 5.0;

arr\_c DQ -1.1, 3.1, 1.0, -3.26, 5.5;

arr\_d DQ 2.2, 1.3, 2.1, 1.99, 10.63;

buffa DB 25 DUP (?);

buffb DB 25 DUP (?);

buffc DB 25 DUP (?);

buffd DB 25 DUP (?);

buff8 DB 250 DUP (?);

buff1 DB 250 DUP (?);

buff2 DB 250 DUP (?);

buff3 DB 250 DUP (?);

buff4 DB 250 DUP (?);

buff5 DB 250 DUP (?);

buff6 DB 250 DUP (?);

buff7 DB 250 DUP (?);

buff9 DB 250 DUP (?);

buff DB 1000 DUP(?), 0;

mainbuff DB 1000 DUP(?), 0;

linetemplate DB 5 DUP("%s"), 0;

;template DB "%s",13,"%s",13, "%s",13, "%s",13, "%s",13, "%s",13, "%s",13, "%s",13

template DB "%s",13; закоментувати для виводу проміжних результатів

maintemplate DB "%s",13

li DW 50 DUP(?), 0;

line1 DW 50 DUP(?), 0;

line2 DW 50 DUP(?), 0;

line3 DW 50 DUP(?), 0;

line4 DW 50 DUP(?), 0;

line5 DW 50 DUP(?), 0;

line6 DW 50 DUP(?), 0;

line7 DW 50 DUP(?), 0;

line8 DW 50 DUP(?), 0;

line9 DW 100 DUP(?), 0;

counter DB 50 DUP(?);

.const

var1 DQ 4.0;

var2 DQ 123.0;

var3 DQ 2.0;

var4 DQ 0.0;

head DB "Лаб. робота №7 Голубов Вираз №%d", 0;

headerr DB "Помилка!Лаб. робота №7 Голубов Вираз №%d",0

text2 DB "Від'ємний вираз під знаком кореня! Виправте параметр d", 0;

text1 DB "Ділення на нуль! Виправте параметр a", 0;

headtask DB "N", 0;

head1 DB "Початкові параметри:" , 0;

head2 DB "Проміжні підрахунки:", 0;

head3 DB "Результат підрахунків:", 0;

head4 DB "(c + 4 \*b - sqrt(123 \* d)) / (1 - a / 2 ) =", 0;

letter DB "%s",13, "a = %s",13,"b = %s", 13, "c = %s", 13,"d = %s", 0;

formate DB "%s", 0;

template1 DB "%s",13, "1. 4 \* b = 4 \* %s = %s", 0

template2 DB "2. c + 4 \* b = %s + 4 \* %s = %s", 0;

template3 DB "3. 123 \* d = 123 \* %s = %s", 0;

template4 DB "4. sqrt(123 \* d) = sqrt(123 \* %s) = %s", 0;

template5 DB "5. c + 4 \* b - sqrt(123 \* d) = %s + 4 \* %s - sqrt(123 \* %s) = %s", 0;

template6 DB "6. a / 2 = %s / 2 = %s", 0;

template7 DB "7. 1 - a / 2 = 1 - %s / 2 = %s", 0;

template8 DB "%s",13, "%s",13,"= (%s + 4 \* %s - sqrt(123 \* %s)) / (1 - %s / 2) = %s ", 0;

.code

|  |  |
| --- | --- |
| one proc  fld qword ptr[eax]; | eax |
| fld qword ptr[edx]; | edx  eax |
| fmul ; | edx\*eax |
| fld qword ptr[ecx] | ecx  edx\*eax |
| fadd ; | ecx+edx\*eax |
| ret  one endp |  |
| second proc  ;prolog  push ebp  mov ebp, esp  ;prolog  xor eax, eax  mov eax, [ebp + 12]  xor ecx, ecx  mov ecx, [ebp + 8]  fld qword ptr [eax]; | eax  c+4\*b |
| fld qword ptr [ecx] | ecx  eax  c+4\*b |
| fmul ; | eax\*ecx  c+4\*b |
| fsqrt; | sqrt(eax\*ecx)  c+4\*b |
| fsub; | c+4\*b-sqrt(eax\*ecx) |
| ;epilog  pop ebp  ;epilog  ret 8  two endp |  |

mainStart:

finit

mainLoop:

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

xor ebx, ebx;

xor eax, eax

mov bx, iter;

fld arr\_a[ebx \* 8];

fcom var3

fstsw ax

sahf

jz eror1

xor eax, eax

jmp firstStart

eror1:

invoke wsprintf, addr counter, addr headerr, it;

invoke MessageBox,NULL, addr text1, addr counter, MB\_OK;

jmp ending

firstStart:

fstp var\_a;

fld arr\_b[ebx \* 8];

fstp var\_b;

fld arr\_c[ebx \* 8];

fstp var\_c;

fld arr\_d[ebx \* 8];

fcom var4

fstsw ax

sahf

jc eror2

xor eax, eax

jmp secondStart

eror2:

invoke wsprintf, addr counter, addr headerr, it;

invoke MessageBox,NULL, addr text2, addr counter, MB\_OK;

jmp ending

secondStart:

fstp var\_d;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

xor eax, eax

xor ecx, ecx

xor edx, edx

mov eax, offset var1

mov ecx, offset var\_c

mov edx, offset var\_b

call one;

push offset var\_d

push offset var2

call two;

call third@0

;fstp temp6

;fstp temp7

;fld temp7

fdiv;

fstp Res; DQ

invoke wsprintf, addr counter, addr head, it;

invoke FloatToStr, var\_a, addr buffa;

invoke FloatToStr, var\_b, addr buffb;

invoke FloatToStr, var\_c, addr buffc;

invoke FloatToStr, var\_d, addr buffd;

invoke FloatToStr, Res, addr buff8;

;invoke FpuFLtoA,offset temp1, 15,offset buff1, SRC1\_REAL

;invoke FpuFLtoA,offset temp2, 15,offset buff2, SRC1\_REAL

;invoke FpuFLtoA,offset temp3, 15,offset buff3, SRC1\_REAL

;invoke FpuFLtoA,offset temp4, 15,offset buff4, SRC1\_REAL

;invoke FpuFLtoA,offset temp5, 15,offset buff5, SRC1\_REAL

;invoke FpuFLtoA,offset temp6, 15,offset buff6, SRC1\_REAL

;invoke FpuFLtoA,offset temp7, 15,offset buff7, SRC1\_REAL

invoke wsprintf, addr line1, addr letter, addr head1, addr buffa, addr buffb, addr buffc, addr buffd

; invoke wsprintf, addr line2, addr template1, addr head2, addr buffb, addr buff1

; invoke wsprintf, addr line3, addr template2, addr buffc, addr buffb, addr buff2;

; invoke wsprintf, addr line4, addr template3, addr buffd, addr buff3;

; invoke wsprintf, addr line5, addr template4, addr buffd, addr buff4;

; invoke wsprintf, addr line6, addr template5, addr buffc, addr buffb, addr buffd, addr buff5;

; invoke wsprintf, addr line7, addr template6, addr buffa, addr buff6;

; invoke wsprintf, addr line8, addr template7, addr buffa, addr buff7;

invoke wsprintf, addr line9, addr template8, addr head3, addr head4, addr buffc, addr buffb, addr buffd, addr buffa, addr buff8

; invoke wsprintf, addr buff, addr template, addr line1, addr line2, addr line3, addr line4, addr line5, addr line6, addr line7, addr line8, addr line9;

invoke wsprintf, addr buff, addr template, addr line1, addr line9; закоментувати для виводу проміжних результатів

invoke MessageBox,NULL, addr buff, addr counter, MB\_OK;

jmp ending

ending:

inc it;

inc iter;

cmp iter, 5

jnz mainLoop

invoke ExitProcess, NULL;

end mainStart

**7-03-IP84-Golubov-Procedure.asm**

.686

.model flat, stdcall

option casemap: none

;extern var\_a:QWORD, var3:QWORD, temp6:QWORD , temp7:QWORD

extern var\_a:QWORD, var3:QWORD; закоментувати для виводу проміжних результатів

public third

.data

.code

third proc

fld1;

fld var\_a;

fld var3;

fdiv;

; fstp temp6;

; fld temp6

fsub;

; fld temp6

ret

third endp

end

**Перевірка результатів розрахунків**

Похибки обчислень виділені **жирним шрифтом**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | №1 | №2 | №3 |
| a=3.05  b=8.1  c=-1.1  d=2.2 | a=1.92  b=9.5  c=3.1  d=1.3 | a=2.98  b=4.01  c=1  d=2.1 |
| 4\**b* | Власні розрахунки | 32.4 | 38 | 16.04 |
| Програма | 32.4 | 38 | 16.04 |
| *c*+4\**b* | Власні розрахунки | 31.3 | 41.1 | 17.04 |
| Програма | 31.3 | 41.1 | 17.04 |
|  | Власні розрахунки | 270.6 | 159.9 | 258.3 |
| Програма | 270.6 | 159.9 | 258.3 |
|  | Власні розрахунки | 16.44992401198255**01** | 12.64515717577287**2** | 16.07171428317465**1** |
| Програма | 16.44992401198255 | 12.64515717577287 | 16.07171428317465 |
|  | Власні розрахунки | 14.85007598801745**2** | 28.45484282422713**01** | 0.9682857168253**498** |
| Програма | 14.85007598801745 | 28.45484282422713 | 0.9682857168253**5** |
|  | Власні розрахунки | 1.525 | 0.96 | 1.49 |
| Програма | 1.525 | 0.96 | 1.49 |
|  | Власні розрахунки | -0.525 | 0.04 | -0.49 |
| Програма | -0.525 | 0.04 | -0.49 |
|  | Власні розрахунки | -28.2858**59** | 711.3711**02** | -1.976093**02** |
| Програма | -28.2858**6** | 711.3711 | -1.976093 |
| Коректність роботи програми |  | ˅ | ˅ | ˅ |

|  |  |  |  |
| --- | --- | --- | --- |
|  | | №4 | №5 |
| a=2.97  b=9.05  c=-3.26  d=1.99 | a=0.77  b=5  c=5.5  d=10.63 |
| 4\**b* | Власні розрахунки | 36.2 | 20 |
| Програма | 36.2 | 20 |
| *c*+4\**b* | Власні розрахунки | 32.94 | 25.5 |
| Програма | 32.94 | 25.5 |
|  | Власні розрахунки | 244.77 | 1307.49 |
| Програма | 244.77 | 1307.49 |
|  | Власні розрахунки | 15.6451270368763**69** | 36.15923118651723**3** |
| Програма | 15.64512703687637 | 36.15923118651723 |
|  | Власні розрахунки | 17.29487296312363**1** | -10.659231186517230**1** |
| Програма | 17.29487296312363 | -10.659231186517230 |
|  | Власні розрахунки | 1.395 | 0.385 |
| Програма | 1.395 | 0.385 |
|  | Власні розрахунки | -0.395 | 0.615 |
| Програма | -0.395 | 0.615 |
|  | Власні розрахунки | -43.78449**1** | -17.3320**78** |
| Програма | -43.78449 | -17.33208 |
| Коректність роботи програми |  | ˅ | ˅ |

**Висновки**

У ході виконання лабораторної роботи я вивчив прийоми модульного програмування, методи звернення до процедур і передачі в них параметрів.

Я створив програму на мові асемблер, в якій використовувати три процедури з різними способами передачі параметрів: через регістри, стек та за допомогою директив EXTRN та PUBLIC.

Чисельник дробу зі свого варіанту індивідуального завдання до лабораторної роботи 6 розділив на два доданка, з яких для першого застосувати передачу параметрів і результату через регістри, а для другого – через стек. Для знаменника використовав метод оголошення загальних змінних директивами public і extern. Виведення результату виконав в основній програмі. Розрахунки повторив в програмі для 5 значень змінних, причому всі вхідні значення задав дійсними числами у вигляді одновимірних масивів.

Перевірив правильність виконання розрахунків і результатів, що виводяться, заздалегідь виконав контрольні розрахунки. Виконав відладку програми шляхом порівняння розрахованих програмою результатів з контрольними прикладами. Проміжні і остаточні результати контрольних розрахунків привів в таблиці. Програма з заданою точністю змогла чітко розрахувати вираз і зобразити його на екрані з підстановкою значень і результату.

У протоколі по лабораторній роботі для першого і другого способів передачі параметрів поруч з відповідними командами у лістингу відобразив в графічному вигляді стани стека при зверненні до процедур, виконання у них команд та повернення з процедур до основної програми.