ЗВІТ

Виконав:

Заночкин Є. Д.

КІТ-119а, Варіант 7

23 вересня 2020 р.

Лабораторна робота №1

Теми: Виконання арифметичних операцій. Процедури з параметрами.

Завдання

1. Знайти результат виразу 2a/7 - c/7e.

2 – 1. Знайти результат виразу a/d + c/b + efg.

2 – 2. Задані масиви A и B с кількістю елементів більше 7. Написати програму формування масиву C згідно з правилом: якщо Ai – Bi ≤ 0, то Cj = Ai.

Тексти програм

1)

```
include \masm64\include64\masm64rt.inc
.data      ;  секция данных
a1 dq 30   ;  объявление операнда a1
c1 dq 20   ;  объявление операнда c1
e1 dq 4    ;  объявление операнда e1
temp1 dq 2 ;  объявление операнда temp1
temp2 dq 7 ;  объявление операнда temp2
title1 db "ЛР1. Решение уравнения.",0 ; название программы
txt1 db "Уравнение 2a/7 – c/7e",10,  ; индивидуальное задание
"Результат: %d",10,"Адрес переменной в памяти: %ph",10,10, ; вывод результата
"Автор: Заночкин Е.Д., КИТ-119а, Вариант 7",0 ; автор
buf1 dq 3 dup(0),0 ; очистка буффера
.code           ; секция кода
entry_point proc ; точка старта программы
mov rax,a1       ; пересылка операнда a1 в rax
mul temp1        ; умножение регистра rax на значение операнда temp1
                 ; инициализация rdx произошла при предыдущем умножении
div temp2        ; результат в rax — целая часть, в rdx — остаток
mov rsi,rax      ; сохраняем результат в регистре rsi
mov rax,e1       ; пересылка в rax операнда e1
mul temp2        ; умножение регистра rax на значение операнда temp2
mov rbx,rax      ; пересылка в rbx значение регистра rax
```

```
mov rax,c1       ; пересылка в rax операнда c1
div rbx          ; деление регистра rax на содержание rbx
sub rsi,rax      ; отнимание значений в регистре rax от значений в регистре rsi
invoke wsprintf,ADDR buf1,ADDR txt1,rsi ; функция преобразования
invoke MessageBox,0,ADDR buf1,ADDR title1,MB_ICONINFORMATION
invoke ExitProcess,0     ; завершение процесса  и  освобождение ресурсов
entry_point endp         ; завершение процедуры с именем
end                      ; завершение программы
```

2)

```
include \masm64\include64\masm64rt.inc ; библиотеки
count PROTO arg_a:QWORD,arg_b:QWORD,arg_c:QWORD,arg_d:QWORD, arg_e:QWORD,arg_f:QWORD,arg_g:QWORD
.data
_a1 dq 5    ; аргумент a
_b1 dq 2    ; аргумент b
_c1 dq 4    ; аргумент c
_d1 dq 1    ; аргумент d
_e1 dq 2    ; аргумент e
_f1 dq 3    ; аргумент f
_g1 dq 4    ; аргумент g
_res1 dq 0  ; переменная для результата
_title db "ЛР1-2. Процедуры.",0
_text db "Уравнение a/d + c/b + efg",10,"Результат: %d",10,"Адрес переменной в памяти: %p",10,10,
"Автор: Заночкин Е.Д., КИТ-119а, Вариант 7",0
buf1 dq 3 dup(0),0 ; буфер для вывода
.code
count proc arg_a:QWORD, arg_b:QWORD, arg_c:QWORD, arg_d:QWORD, arg_e:QWORD, arg_f:QWORD, arg_g:QWORD
mov rbx,rdx     ; заносим в rbx аргумент b
mov rax,rcx     ; заносим в rax аргумент a
xor rdx,rdx     ; обнуление регистра rdx
div r9          ; деление аргумента a на аргумент d
mov r10,rax     ; заносим результат a/d в регистр r10
mov rax,r8      ; заносим в rax аргумент c
mov r11, rbx    ; заносим в r11 аргумент b
xor rdx,rdx     ; обнуление регистра rdx
div r11         ; деление аргумента c на аргумент b
mov r12,rax     ; заносим результат c/b в регистр r12
add r10,r12     ; суммируем a/d и c/b
mov rax,arg_e   ; заносим в rax аргумент e
mov r13,arg_f   ; заносим в r13 аргумент f
```

```
mul r13         ; умножаем e на f
mov r14,arg_g   ; заносим в r14 аргумент g
mul r14         ; умножаем rax на аргумент f
add r10,rax     ; суммируем a/d и c/b и efg
mov _res1, r10  ; заносим результат в _res1
ret
count endp
entry_point proc
invoke count,_a1,_b1,_c1,_d1,_e1,_f1,_g1
invoke wsprintf,ADDR buf1,ADDR _text,_res1,ADDR _res1
invoke MessageBox,0,addr buf1, addr _title, MB_ICONINFORMATION
invoke ExitProcess,0
entry_point endp
end
```

3)

```
include \masm64\include64\masm64rt.inc          ; подключение библиотеки
.data                                   ; секция данных
arrA dq -43, -12, 8, -30, 9, 23, 56, 72         ; массив A
arrB dq 52, -32, 12, 12, 40, -32, 7, 2          ; массив B
arrC dq 8 dup(?)                            ; массив C
len1 dq 8                               ; длинна массива A
len2 dq 8                               ; длинна массива B
count1 dq 0                               ; кол-во циклов
res1 dq 0                               ; переменная результата
res2 dq 0                               ; переменная результата
res3 dq 0                               ; переменная результата
res4 dq 0                               ; переменная результата
title1 db "Лабораторная работа 1-2-2. Процедуры с параметрами. Массивы",0       ; заголовок окна вывода
txt1 db "Заданы массивы A и B с числом элементов больше 7. Написать программу формирования массива C по такому правилу: если Ai – Bi <= 0, то Cj = Ai.",10,10,
"Результат: ",10,
"arrC[0]: %d",10,
"arrC[1]: %d",10,
"arrC[2]: %d",10,
"arrC[3]: %d",10,10,
"Автор: Заночкин Е.Д., КИТ-119а",0
buf1 dq 3 dup(0),0
.code           ; директива сегмента кода
entry_point proc
xor rax,rax     ; очистка регистра RAX
xor rsi,rsi     ; очистка регистра RSI
```

```
xor rdi,rdi     ; очистка регистра RDI
xor rbp,rbp     ; очистка регистра RBP
xor rcx,rcx     ; очистка регистра RCX
xor rbx,rbx     ; очистка регистра RBX
xor r10,r10     ; очистка регистра R10
mov rcx,count1          ; указание кол-ва циклов
lea rsi,byte ptr arrA   ; установка указателя в начало массива A
lea rdi,byte ptr arrB   ; установка указателя в начало массива B
lea rbp,byte ptr arrC   ; установка указателя в начало массива C
@1:
mov rax,[rsi]           ; запись элемента массива A в RAX
mov rbx,[rdi]           ; запись элемента массива B в RBX
inc r10                 ; инкремент регистра R10
sub rax,rbx             ; отнимание элементов массивов
cmp rax,0               ; сравнение разницы элементов с нулём
jle BelowOrEquelZero    ; если сумма элементов <= 0, переход в ф-ию BelowOrEquelZero
jmp CheckArr            ; переход на проверку на выход за грани массива
BelowOrEquelZero:
mov rcx,[rsi]           ; запись элемента Ai в регистр RCX
mov [rbp],rcx           ; запись элемента в массив C
add rbp,type arrC       ; перемещение на следующий элемент массива C
jmp CheckArr            ; переход на проверку на выход за грани массива
CheckArr:
add rsi,type arrA       ; перемещение на следующий элемент массива A
add rdi,type arrB       ; перемещение на следующий элемент массива B
cmp r10,len1            ; проверка на выход за грани массива A
je _end                 ; если массив A пройден, то перейти в конец программы
cmp r10,len2            ; проверка на выход за грани массива B
je _end                 ; если массив B пройден, то перейти в конец программы
jmp @1                  ; переход в начало цикла
_end:                   ; конец программы
xor rax,rax             ; очистка регистра RAX
xor rbp,rbp             ; очистка регистра RBP
lea rbp,byte ptr arrC   ; установка указателя в начало массива C
mov rax,[rbp]           ; запись из массива C в регистр RAX
mov res1,rax            ; запись из RAX в переменную res1
xor rax,rax             ; очистка регистра RAX
add rbp,type arrC       ; переместиться на следующий элемент массива
mov rax,[rbp]           ; запись из массива C в регистр RAX
mov res2,rax            ; запись из RAX в переменную res2
xor rax,rax             ; очистка регистра RAX
add rbp,type arrC       ; переместиться на следующий элемент массива
```

```
mov rax,[rbp]          ; запись из массива C в регистр RAX
mov res3,rax           ; запись из RAX в переменную res3
xor rax,rax            ; очистка регистра RAX
add rbp,type arrC      ; переместиться на следующий элемент массива
mov rax,[rbp]          ; запись из массива C в регистр RAX
mov res4,rax           ; запись из RAX в переменную res4
invoke wsprintf,ADDR buf1,ADDR txt1, res1, res2, res3, res4
invoke MessageBox,0,ADDR buf1,ADDR title1,MB_ICONINFORMATION
invoke ExitProcess,0
entry_point endp       ; точка выхода
end
```

Результати виконання програм

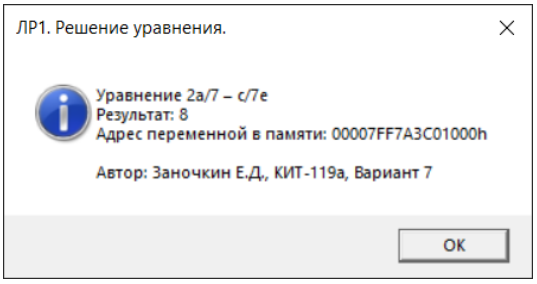

Рисунок 1.1а – Результат роботи 1 в MessageBox

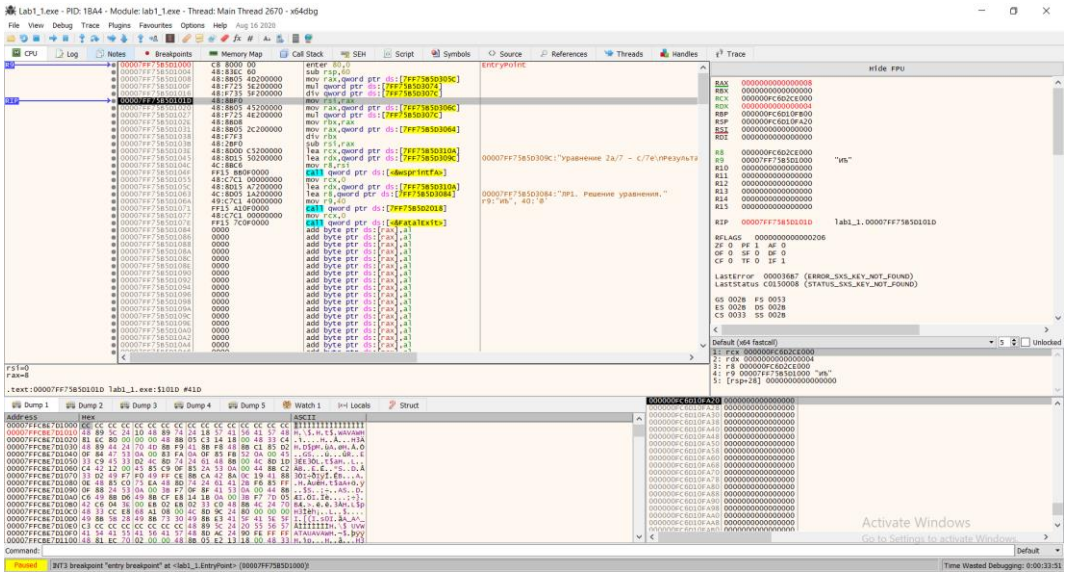

Рисунок 1.1б – Результат роботи 1 в x64dbg

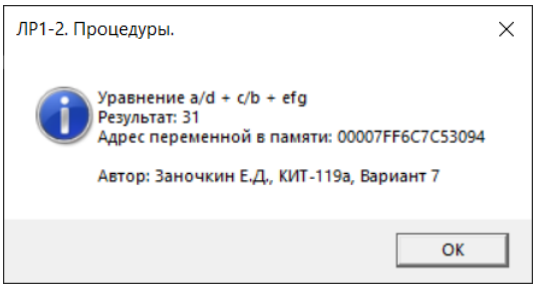

Рисунок 1.2а – Результат роботи 1-2-1 в MessageBox

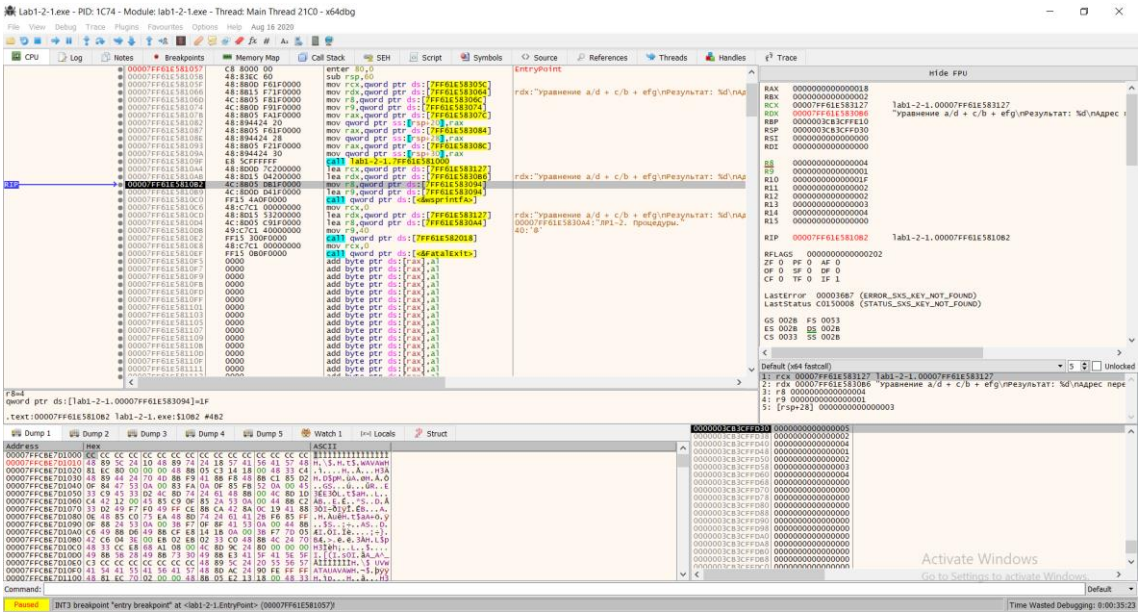

Рисунок 1.2б – Результат роботи 1-2-1 в x64dbg

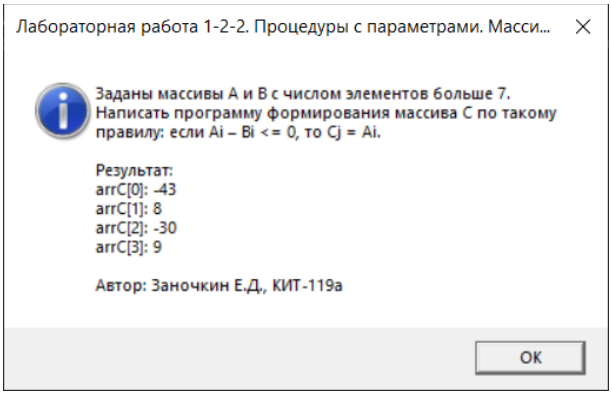

Рисунок 1.2в – Результат роботи 1-2-2 в MessageBox

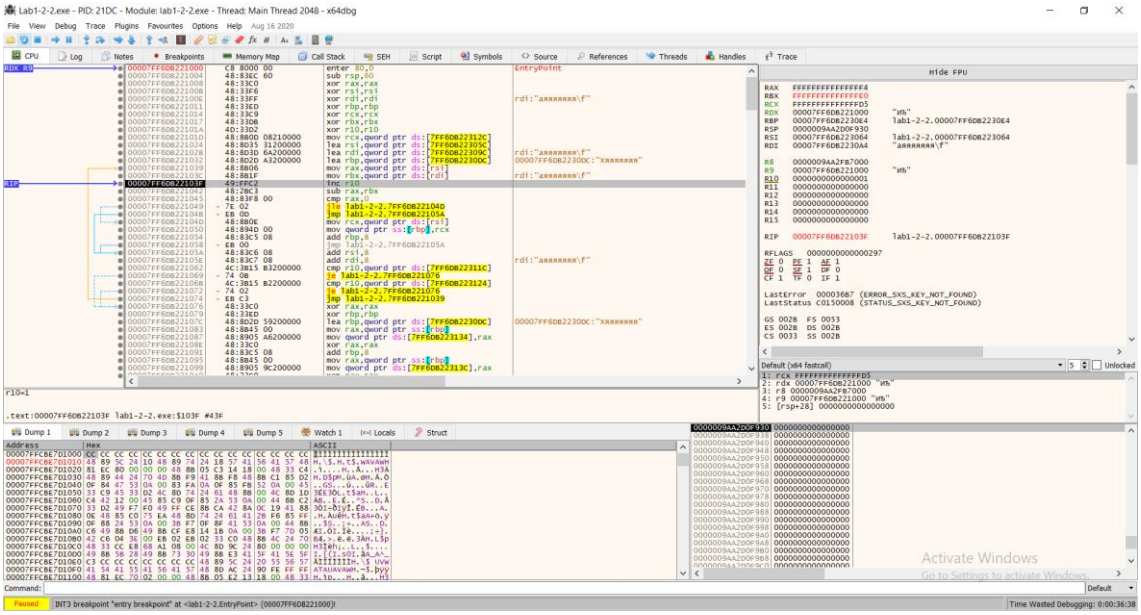

Рисунок 1.2г – Результат роботи 1-2-2 в x64dbg

Алгоритми виконання

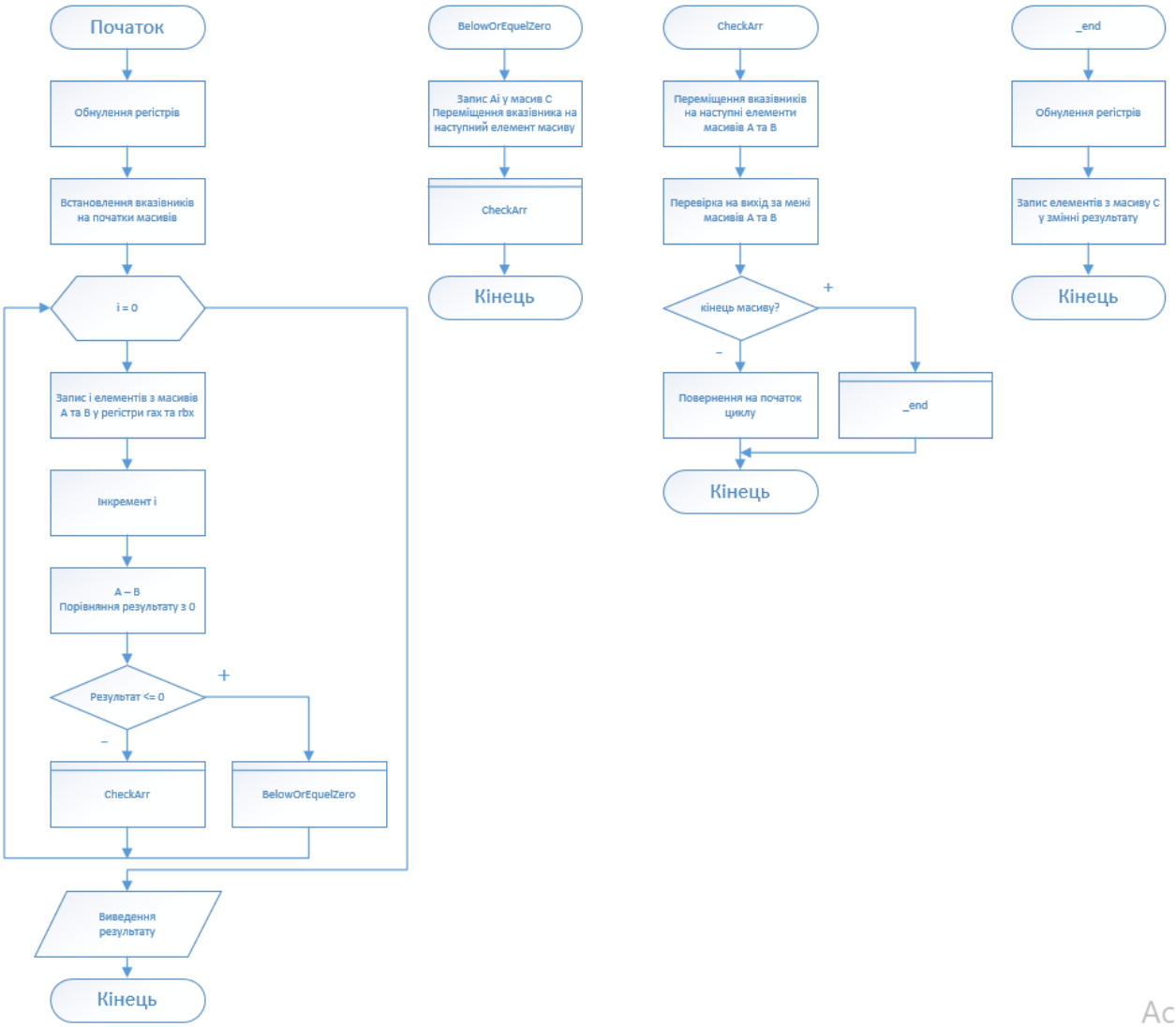

Рисунок 1.3 – Алгоритм виконання програми 1-2-2

Висновок

Під час лабораторної роботи було створено 3 програми, які виконуються згідно з індивідуальним завданням, було набуто навички проектування програм, в тому числі програм з процедурами та масивами. Програми протестовані, працюють без помилок.