

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ
по лабораторной работе №3
по дисциплине «Организация ЭВМ и систем»
ТЕМА: ПРОГРАММИРОВАНИЕ ВЕТВЯЩИХСЯ ПРОЦЕССОВ.

Студент гр. 0383

Парфенов В.М.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Целью данной лабораторной работы заключается в создании программы с использованием ветвящихся процессов для вычисления значений переменных.

Задание.

Разработать на языке Ассемблера программу, которая по заданным целочисленным значениям параметров a , b , i , k вычисляет:

а) значения функций $i1 = f1(a,b,i)$ и $i2 = f2(a,b,i)$;

б) значения результирующей функции $res = f3(i1,i2,k)$, где вид функций $f1$ и $f2$ определяется из табл. 2, а функции $f3$ - из табл.3 по цифрам шифра индивидуального задания ($n1,n2,n3$), приведенным в табл.4. Значения a , b , i , k являются исходными данными, которые должны выбираться студентом самостоятельно и задаваться в процессе исполнения программы в режиме отладки. При этом следует рассмотреть всевозможные комбинации параметров a , b и k , позволяющие проверить различные маршруты выполнения программы, а также различные знаки параметров a и b .

Замечания:

1) при разработке программы нельзя использовать фрагменты, представленные на ЯВУ, в частности, для ввода-вывода данных. Исходные данные должны вводиться, а результаты контролироваться в режиме отладки;

2) при вычислении функций $f1$ и $f2$ вместо операции умножения следует использовать арифметический сдвиг и, возможно, сложение;

3) при вычислении функций $f1$ и $f2$ нельзя использовать процедуры;

4) при разработке программы следует минимизировать длину кода, для чего, если надо, следует преобразовать исходные выражения для вычисления функций.

Вариант 8

$/ 15-2*i$, при $a>b$

$f1 = <$

$\backslash 3*i+4$, при $a\leq b$

$\text{ / - (6*i+8), при } a > b$

$f2 = <$

$\text{ \ 9 -3*(i-1), при } a \leq b$

$\text{ / |i1| + |i2|, при } k < 0$

$f7 = <$

$\text{ \ max(6, |i1|), при } k \geq 0$

Выполнение работы.

Таблица 1 – Результаты тестирования

a	b	i	k	i1	i2	res	Комментарий
2	6	5	10	19	-3	19	$a \leq b; k > 0 \Rightarrow res = \max(6, 19) = 19$
3	4	4	-6	16	0	16	$a \leq b; k < 0 \Rightarrow res = 16 + 0 = 16$
8	5	6	9	3	-44	47	$a > b; k > 0 \Rightarrow res = \max(6, 3) = 6$
9	6	-4	-1	23	16	39	$a > b; k < 0 \Rightarrow res = 23 + 16 = 39$
-9	-11	-4	-1	23	16	39	$a > b; k < 0 \Rightarrow res = 23 + 16 = 39$

Выводы.

В ходе данной лабораторной работы была достигнута цель по созданию программы с помощью ветвящихся процессов, а также была изучена работа с условными переходами, командами сравнения и целыми числами.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb3.asm

EOFLine EQU '\$'

ASSUME CS:CODE, SS:AStack

AStack SEGMENT STACK

DW 12 DUP(?)

AStack ENDS

DATA SEGMENT

a DW 9

b DW 6

i DW -4

k DW -1

i1 DW 0

i2 DW 0

res DW 0

buff DW 0

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA, SS:AStack

Main PROC FAR

push DS

sub AX,AX

push AX

mov AX,DATA

mov DS,AX

mov CX, 0

mov cx, i

mov ax, cx

shl cx, 1

mov buff, cx

mov bx, b

cmp a, bx

; Система f1, f1s - при $a \leq b$

jle f12

; вычисление i1 при $a > b$

neg cx

add cx, 15

mov i1, cx

; вычисление i2 при $a > b$

mov cx, buff

shl cx, 1

add cx, ax

add cx, ax

neg cx

sub cx, 8

mov i2, cx

jmp takeModule

f12: ; a <= b

; вычисление i1 при a <= b

add cx, ax

add cx, 4

mov i1, cx

; вычисление i2 при a <= b

mov cx, buff

add cx, ax

neg cx

add cx, 12

mov i2, cx

takeModule:

; взятие модуля от i1

mov cx, i1

cmp cx, 0

jl module_i1

jmp skip1

module_i1:

neg cx

mov i1, cx

; взятие модуля от i2

skip1:

mov cx, i2

cmp cx, 0

jl module_i2

jmp skip2

module_i2:

neg cx

mov i2, cx

skip2:

cmp k, 0

```
; Вычисление res = f3, f3s при k < 0  
jl f3s
```

```
; mov cx, i1
```

```
cmp i1, 6  
jg first_bigger
```

```
mov res, 6  
jmp Mainend
```

```
first_bigger:  
    mov cx, i1  
    mov res, cx  
    jmp Mainend
```

```
f3s: ; k < 0
```

```
    mov cx, i1  
    add cx, i2  
    mov res, cx
```

```
Mainend:  
ret
```

```
Main    ENDP  
CODE    ENDS
```


END Main

Название файла: lb3.lst

Microsoft (R) Macro Assembler Version 5.10

11/9/21 12:18:19

Page 1-1

= 0024

EOFLine EQU '\$'

ASSUME CS:CODE, SS:AStack

0000

AStack SEGMENT STACK

0000 000C[

DW 12 DUP(?)

????

]

0018

AStack ENDS

0000

DATA SEGMENT

0000 0009

a DW 9

0002 0006

b DW 6

0004 FFFC

i DW -4

0006 FFFF

k DW -1

0008 0000

i1 DW 0

000A 0000	i2 DW 0
000C 0000	res DW 0
000E 0000	buff DW 0
0010	DATA ENDS
0000	CODE SEGMENT
	ASSUME CS:CODE, DS:DATA, SS:AStack
0000	Main PROC FAR
0000 1E	push DS
0001 2B C0	sub AX,AX
0003 50	push AX
0004 B8 ---- R	mov AX,DATA
0007 8E D8	mov DS,AX
0009 B9 0000	mov CX, 0
000C 8B 0E 0004 R	mov cx, i
0010 8B C1	mov ax, cx
0012 D1 E1	shl cx, 1
0014 89 0E 000E R	mov buff, cx
0018 8B 1E 0002 R	mov bx, b
001C 39 1E 0000 R	cmp a, bx
	; $\text{f1, f1s} - \text{a} \leq \text{b}$
0020 7E 1F	jle f12

;        if   a > b

```
0022 F7 D9          neg cx
0024 83 C1 0F       add cx, 15
0027 89 0E 0008 R   mov i1, cx
```

```
; 00000000 00000000 i2 00000000 a > b
```

```
002B 8B 0E 000E R   mov cx, buff
002F D1 E1         shl cx, 1
0031 03 C8         add cx, ax
0033 03 C8         add cx, ax
0035 F7 D9        neg cx
0037 83 E9 08      sub cx, 8
```

```
003A 89 0E 000A R   mov i2, cx
```










```
003E EB 19 90      jmp takeModule
```

```
0041              f12: ; a <= b
```











```
; 00000000 00000000 i1 00000000 a <= b
```

```
0041 03 C8         add cx, ax
0043 83 C1 04      add cx, 4
0046 89 0E 0008 R   mov i1, cx
```

```

;        i2   a <= b

004A 8B 0E 000E R      mov cx, buff
004E 03 C8             add cx, ax
0050 F7 D9             neg cx
0052 83 C1 0C          add cx, 12
0055 89 0E 000A R      mov i2, cx

0059                  takeModule:
;           i1










0059 8B 0E 0008 R      mov cx, i1

005D 83 F9 00          cmp cx, 0

0060 7C 03             jl module_i1

0062 EB 07 90          jmp skip1

0065                  module_i1:
0065 F7 D9             neg cx
0067 89 0E 0008 R      mov i1, cx

;          i2

006B                  skip1:
006B 8B 0E 000A R      mov cx, i2

006F 83 F9 00          cmp cx, 0

```










```
0072 7C 03                jl module_i2

0074 EB 07 90            jmp skip2

0077                    module_i2:
0077 F7 D9                neg cx
0079 89 0E 000A R        mov i2, cx

007D                    skip2:

007D 83 3E 0006 R 00     cmp k, 0

;        res = f3, f3s   k < 0

0082 7C 1B                jl f3s

; mov cx, i1

0084 83 3E 0008 R 06     cmp i1, 6
0089 7F 09                jg first_bigger

008B C7 06 000C R 0006   mov res, 6
0091 EB 18 90            jmp Mainend

0094                    first_bigger:
0094 8B 0E 0008 R        mov cx, i1
```

0098	89 0E 000C R	mov res, cx
009C	EB 0D 90	jmp Mainend

009F	f3s: ; k < 0	
------	--------------	--

009F	8B 0E 0008 R	mov cx, i1
00A3	03 0E 000A R	add cx, i2
00A7	89 0E 000C R	mov res, cx

00AB	Mainend:
00AB CB	ret

00AC	Main	ENDP
00AC	CODE	ENDS
	END Main	

Symbols-1

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA		STACK
CODE	00AC	PARA		NONE
DATA	0010	PARA		NONE

Symbols:

N a m e	Type	Value	Attr
A	L WORD	0000	DATA
B	L WORD	0002	DATA
BUFF	L WORD	000E	DATA
EOFLINE	NUMBER	0024	
F12	L NEAR	0041	CODE
F3S	L NEAR	009F	CODE
FIRST_BIGGER	L NEAR	0094	CODE
I	L WORD	0004	DATA
I1	L WORD	0008	DATA
I2	L WORD	000A	DATA

K L WORD 0006 DATA

MAIN F PROC 0000 CODE Length = 00AC

MAINEND L NEAR 00AB CODE

MODULE_I1 L NEAR 0065 CODE

MODULE_I2 L NEAR 0077 CODE

RES L WORD 000C DATA

SKIP1 L NEAR 006B CODE

SKIP2 L NEAR 007D CODE

TAKEMODULE L NEAR 0059 CODE

@CPU TEXT 0101h

@FILENAME TEXT 1b3

@VERSION TEXT 510

Symbols-2

148 Source Lines

148 Total Lines

27 Symbols

48030 + 457180 Bytes symbol space free

0 Warning Errors

0 Severe Errors