

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

**ОТЧЕТ**  
**по лабораторной работе №3**  
**по дисциплине «Организация ЭВМ и систем»**  
**Тема: Представление и обработка целых чисел. Организация**  
**ветвящихся процессов**

Студент гр. 9382

\_\_\_\_\_

Субботин М.О.

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2020

### **Цель работы.**

Научиться представлять и обрабатывать целые числа, а также организовывать ветвящиеся процессы.

### **Основные теоретические положения.**

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  $b$ .

Вариант 20: 4.6.8

$$f4 = \begin{cases} -(6*i-4), & \text{при } a > b \\ \end{cases}$$

$$\backslash 3*(i+2), \text{ при } a \leq b$$

$$f6 = \begin{cases} /2*(i+1)-4, & \text{при } a > b \\ \end{cases}$$

$$\backslash 5 - 3*(i+1), \text{ при } a \leq b$$

$$f8 = \begin{cases} /|i1|-|i2|, & \text{при } k < 0 \\ \end{cases}$$

$$\backslash \max(4, |i2|-3), \text{ при } k \geq 0$$

### **Ход выполнения:**

В сегменте данных заданы метки для переменных a, b, k, l, i1, i2, res. Функции и их ветвления были реализованы с помощью меток в фрагменте кода. Эти “функции” кладут возвращаемые значения на стек. Для реализации ветвления функций и самих функций использовалась операция CMP, которая сравнивала два числа. В зависимости от сравнения двух чисел с помощью переходов мы переходили на метки, соответствующие конкретному ветвлению в функции. К примеру, если мы сравниваем два числа a и b и хотим, чтобы в случае  $a > b$  выполнялась функция по метке f, то следует выполнить:

```
cmp a,b
jg f1
```

Если же у нас существует ветвление и при  $a \leq b$ , то код, выполняемый в таком случае можно просто поместить ниже. Собственно с помощью таких ветвлений и переходов по коду и были реализованы функции с ветвлениями.

### **Исходный код программы:**

```
STACKSG SEGMENT PARA STACK 'Stack'
```

```
    DW    32 DUP(?)
```

```
STACKSG ENDS
```

```
DATASG SEGMENT PARA 'Data' ;SEG DATA
```

```
VARA    DW  1h
```

```
VARB    DW  1h
```

```
VARIDW  1h
```

```
VARK    DW  1h
```

```
VARI1   DW  1h
```

VARI2     DW   1h

VARRES   DW   1h

DATASG   ENDS

;ENDS DATA

CODE     SEGMENT

;SEG CODE

ASSUME DS:DataSG, CS:Code

Main     PROC FAR

      mov ax, DATASG

      mov ds, ax

      jmp f1

f1\_end:

      mov ax, VARI1

      pop VARI1

      jmp f2

f2\_end:

      pop VARI2

      jmp f3

f3\_end:

      pop VARRES

```

mov ah, 4ch          ;завершаем программу
int 21h

```

f1:

```

mov ax,VARA          ;переменная a в ax
mov si,VARB          ;переменная b в si
mov bx,VARI          ;переменная i в bx
shl bx,1             ;умножаем переменную i в bx на 2

cmp ax,si            ;сравниваем переменные a и b соответственно

jg  f1_1             ;если a>b переходим к метке f1_1

```

f1\_2: ;если a<=b ,то считаем  $3*(i+2)$

```

mov ax,VARI          ;ax = i
add ax,bx            ;ax = ax + bx = i + 2*i = 3i
add ax,6             ;ax +=6, ax = 3i+6
push ax              ;ret ax
jmp 1_end

```

f1\_1: ; если a>b, то считаем  $-(6*i-4) = -6*i + 4$

```

shl bx, 1            ;bx *= 2 , bx = 4*i
shl bx, 1            ;bx *= 2, bx = 8*i

```

```

mov ax,VARI      ;ax = i
shl ax,1         ;ax = 2*i
sub ax,bx        ;ax = ax - bx = 2*i - 8*i = -6*i
add ax,4         ;ax = -6*i + 4
push ax
                ;ret ax
jmp fl_end

```

f2:

```

mov ax,VARA      ;переменная а в ax
cmp ax,VARB      ;сравниваем переменные а и b соответственно
jg  f2_1         ;если a>b переходим к метке f2_1

```

f2\_2: ; если  $a \leq b$ , то считаем  $5 - 3*(i+1) = 2 - 3*i$

```

mov ax,VARI      ;ax = i
shl ax, 1        ;ax *= 2 = 2*i
shl ax, 1        ;ax *= 2 = 4*i
mov bx,VARI      ;bx = i
sub bx,ax        ;bx = bx - ax = i - 4*i = -3*i
add bx,2         ;bx+=2
push bx
                ;ret bx
jmp f2_end

```

```

f2_1:                                ; если a>b, то считаем  $2*(i+1)-4 = 2*i - 2$ 

    mov bx,VARI                      ;bx = i

    shl bx,1                        ;bx *= 2

    sub bx,2                        ;bx = bx - 2 = 2*i - 2

    push bx                          ;ret bx

    jmp f2_end

```

```

f3:

    mov     ax,VARI2                ;кладем в ax i2

    cmp ax,0                        ; сравниваем i2 с 0

    jl  f3_I2_NEG                   ;если i2 < 0

```

```

f3_K_POS:

    mov bx,VARCK                    ;кладем в bx k

    cmp bx, 0                      ;сравниваем k с 0

    jl f3_K_NEG

    sub ax,3                        ;ax = ax-3 = |i2|-3

    cmp 4,ax                       ;сравниваем 4 с |i2|-3

    jl f3_ax                        ;если 4 < |i2|-3

    push 4

    jmp f3_end

```

```

f3_ax:

    push ax

```

```
jmp f3_end
```

```
f3_K_NEG:
```

```
mov bx, VARI1    ;кладем в bx i1
```

```
cmp bx, 0        ;сравниваем i1 с 0
```

```
jl f3_I1_NEG     ;если i1<0
```

```
f3_K_NEG_COUNT:
```

```
sub bx, ax
```

```
push bx
```

```
jmp f3_end
```

```
f3_I1_NEG:
```

```
neg bx
```

```
jmp f3_K_NEG_COUNT
```

```
f3_I2_NEG:
```

```
neg ax
```

```
jmp f3_K_POS
```

```
Main    ENDP
```

```
CODE    ENDS
```



END Main ;ENDS CODE

Main ENDP

CODE ENDS

END Main ;ENDS CODE

### Листинг программы:

□Microsoft (R) Macro Assembler Version 5.10

10/9/20 19:22:55

Page 1-1

0000 STACKSG SEGMENT PARA STACK 'Stack'

0000 0020[ DW 32 DUP(?)

????

]

0040 STACKSG ENDS

0000 DATASG SEGMENT PARA 'Data'

;SEG DATA

0000 0001 VARA DW 1h

0002 0001 VARB DW 1h

0004 0001 VARIDW 1h

0006 0001 VARK DW 1h

0008 0001 VARI1 DW 1h

000A 0001 VARI2 DW 1h

000C 0001 VARRES DW 1h

```

000E          DATASG  ENDS

                                ;ENDS DATA

```

```

0000          CODE   SEGMENT

                                ;SEG CODE

ASSUME DS:DataSG, CS:Code

```

```

0000          Main   PROC FAR

0000 B8 ---- R          mov ax, DATASG

0003 8E D8              mov ds, ax

0005 EB 1A 90           jmp f1

0008          f1_end:

0008 A1 0008 R          mov ax, VARI1

000B 8F 06 0008 R       pop VARI1

000F EB 41 90           jmp f2

0012          f2_end:

0012 8F 06 000A R       pop VARI2

0016 EB 66 90           jmp f3

0019          f3_end:

0019 8F 06 000C R       pop VARRES

```

[illegible]














































001F CD 21 int 21h

```
0021      fl:;dw fl(VARA,VARB,VARI)
```

**0021 A1 0000 R**

a      mov ax,VARA                  ;⌋ ⌋⌋ =A ⌋⌋ ⌋⌋ ⌋⌋ ⌋⌋ ⌋⌋⌋⌋⌋⌋=Π

$$\perp\!\!\!\perp \text{ax}$$

0024 8B 36 0002 R                    mov si,VARB  
;    A                                            



Chemical structure showing a silicon atom (Si) bonded to a hydrogen atom (H) and a methyl group (CH<sub>3</sub>).

```
0028 8B 1E 0004 R      mov bx,VARI
```

$$; \vdash_{\Gamma} \vdash_{\Gamma} =_A \vdash_{\Gamma} \vdash_{\Gamma} \vdash_{\Gamma}$$
$$\prod_{i=1}^n \frac{1}{\Gamma(\alpha_i)} = \prod_{i=1}^n \frac{1}{\Gamma(\beta_i)} \quad \text{for } \alpha_i = \beta_i$$

```
002C D1 E3          shl bx,1          ; shl bx,1
```

[illegible]

□Microsoft (R) Macro Assembler Version 5.10

10/9/20 19:22:55

Page 1-2

002E 3B C6

cmpax,si

$$; \vdash B \vdash A \quad \text{---} \quad \text{---} \quad \text{---} \quad \text{---} \quad \text{---} \quad \text{---}$$

$\neg B \dashv\vdash B$     $\neg B \dashv\vdash \neg\neg B$     $\neg\neg A \dashv\vdash A$     $\neg\neg B \dashv\vdash B$     $\neg\neg\neg A \dashv\vdash \neg A$    a   b

$$\overline{\tau}B\overline{\tau}B\overline{\tau}B \begin{array}{|c|} \hline \text{---} \\ \hline \end{array} \begin{array}{|c|} \hline \text{---} \\ \hline \end{array} \begin{array}{|c|} \hline \text{---} \\ \hline \end{array}$$

```
0030 7F 0C          jg    f1_1          ;  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
          1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
          1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
          1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
```

0032 f1\_2: ; $\vdash \vdash B \vdash \vdash$   
 $\vdash \vdash a \leq b, \vdash B \vdash \vdash \vdash B \vdash \vdash 3 \vdash \vdash \vdash \vdash \vdash \vdash 3 * (i + 2)$

```
0032 A1 0004 R      mov ax,VARI      ;ax = i
```

```
0035 03 C3          add ax,bx          ;ax = a
```

$$\mathbf{x} + \mathbf{b}\mathbf{x} = \mathbf{i} + 2*\mathbf{i} = 3\mathbf{i}$$

```
0037 05 0006          add ax,6          ;ax +=6
                        , ax = 3i+6
```

```
003A 50          push ax                      ;ret ax
```

```
003B EB 13 90      jmp fl_c
```

003E fl\_1: ;  $\mathbb{L} \dashv \vdash \mathbb{B} \mathbb{L}$   
 $\neg \mathbb{L} \dashv \vdash a > b, \neg \mathbb{B} \mathbb{L} \dashv \vdash \neg \mathbb{B} \neg 3 \mathbb{L} \dashv \vdash \neg \mathbb{B} \mathbb{L} \mathbb{L} \dashv \vdash \mathbb{L} \mathbb{L} \dashv \vdash -(6 * i - 4) = -6 * i +$   
 4

```
003E D1 E3          shl bx, 1          ;bx *=
                2 , bx = 4*i
```

```
0040 D1 E3          shl bx, 1          ;bx *= 2, bx = 8*i
```

```
0042 A1 0004 R      mov ax,VARI      ;ax = i
```



```
005B A1 0004 R      mov ax,VARI      ;ax = i
```

```
005E D1 E0          shl ax, 1          ;ax *=
```

$$2 = 2^* \mathbf{i}$$

```
0060 D1 E0          shl ax, 1          ;ax *= 2 = 4*i
```

```
0062 8B 1E 0004 R      mov bx,VARI      ;bx = i
```

```
0066 2B D8          sub bx,ax      ;bx = bx - ax = i -
```

□Microsoft (R) Macro Assembler Version 5.10 10/9/20 19:22:55

Page 1-3

$$4\mathbf{i} = -3\mathbf{i}$$

```
0068 83 C3 02          add bx,2          ;bx+=2
```

```
006B 53          push bx          ;ret bx
```

```
006C EB 0E 90      jmp f2_c
```

006F                      f2\_1:                      ;  $\frac{1}{T} = \frac{1}{T_0} + \frac{1}{T_1}$

[illegible]

```
006F 8B 1E 0004 R      mov bx,VARI      ;bx = i
```

```
0073 D1 E3          shl bx,1          ;bx *=
```

2

```
0075 83 EB 02          sub bx,2          ;bx = b
```

$$x - 2 = 2 * i - 2$$

0078	53	push bx	;ret bx
0079	EB 01 90	jmp f2_c	
007C		f2_c:	
007C	EB 94	jmp f2_end	
007E		f3: ;dw f3(VARI1,VARI2,VARK)	
007E	A1 0006 R	mov ax,VARK	;ax = *si
		(k)	
0081	3D 0000	cmp ax,0	;cmp k,0
0084	7C 21	jl f3_1_1	;if k < 0
0086		f3_2_1:	;if k >= 0
0086	A1 000A R	mov ax,VARI2	;ax = i2
0089	3D 0000	cmp ax,0	;if i2 < 0
008C	7C 0E	jl f3_2_c1	;if i2 < 0
008E		f3_2_2:	
008E	2D 0003	sub ax,3	;ax =  i2 -3
0091	3D 0004	cmp ax,4	







00B6 7C 09 j1 f3\_1\_c2 ;lll 7Blll ll 7x lll 7B7A  
lll 7Jlll 7Blll ll 7Mlll ll

```
00BA EB 09 90      jmp f3_1_c
```

[illegible]


00BD F7 D8 neg ax ;  
 11 11 11 11 11 11  
 11 11 11 11 11 11

```
00BF EB EE      jmp f3_1_2
```

00C1 f3\_1\_c2: ;  
 bxi2) B A J B M

00C1 F7 DB neg bx ;  
;  $\frac{1}{\Gamma} \Pi$   
 $\frac{1}{\Gamma} i_2$

```
00C3 EB F3                jmp f3_1
```

00C5 50                    push ax                    ; 

[illegible]

```
00C6 EB 01 90      jmp f3_c
```

00C9 f3\_c:

```
00C9 E9 0019 R      jmp f3_end
```

00CC	Main	ENDP
------	------	------

□Microsoft (R) Macro Assembler Version 5.10 10/9/20 19:22:55

Page 1-5

```
00CC                                CODE    ENDS
```

END Main ;ENDS C

ODE

□Microsoft (R) Macro Assembler Version 5.10 10/9/20 19:22:55

## Symbols-1

### Segments and Groups:

N a m e	Length	Align	Combine	Class
---------	--------	-------	---------	-------

CODE ..... 00CC PARA NONE

DATASG .....	000E	PARA	NONE	'DATA'
STACKSG .....	0040	PARA	STACK	'STACK'

Symbols:

N a m e	Type	Value	Attr
F1 .....	L NEAR	0021	CODE
F1_1 .....	L NEAR	003E	CODE
F1_2 .....	L NEAR	0032	CODE
F1_C .....	L NEAR	0050	CODE
F1_END .....	L NEAR	0008	CODE
F2 .....	L NEAR	0052	CODE
F2_1 .....	L NEAR	006F	CODE
F2_2 .....	L NEAR	005B	CODE
F2_C .....	L NEAR	007C	CODE
F2_END .....	L NEAR	0012	CODE
F3 .....	L NEAR	007E	CODE
F3_1 .....	L NEAR	00B8	CODE
F3_1_1 .....	L NEAR	00A7	CODE
F3_1_2 .....	L NEAR	00AF	CODE
F3_1_C .....	L NEAR	00C5	CODE
F3_1_C1 .....	L NEAR	00BD	CODE
F3_1_C2 .....	L NEAR	00C1	CODE

```

F3_2_1 ..... L NEAR   0086 CODE
F3_2_2 ..... L NEAR   008E CODE
F3_2_C .....      L NEAR   00A0 CODE
F3_2_C1 .....      L NEAR   009C CODE
F3_C .....      L NEAR   00C9 CODE
F3_END .....      L NEAR   0019 CODE

MAIN .....      F PROC   0000 CODE      Length = 00CC

VARA .....      L WORD   0000 DATASG
VARB .....      L WORD   0002 DATASG
VARI .....      L WORD   0004 DATASG
VARI1 .....      L WORD   0008 DATASG
VARI2 .....      L WORD   000A DATASG
VARK .....      L WORD   0006 DATASG
VARRES .....      L WORD   000C DATASG

@CPU .....      TEXT 0101h
@FILENAME .....  TEXT lab3
@VERSION .....  TEXT 510

```

159 Source Lines

159 Total Lines

41 Symbols

48040 + 453075 Bytes symbol space free

0 Warning Errors

0 Severe Errors

### **Тестирование.**

<b>№</b>	<b>Входные данные</b>	<b>Выходные данные</b>	<b>Правильный результат</b>
<b>1</b>	<b>a=2 b=1 i=5 k=-26</b>	<b>i1=-26 i2=8 res=16</b>	<b>i1=-26 i2=8 res=16</b>
<b>2</b>	<b>a=2 b=1 i=3 k=2</b>	<b>i1=-14 i2=4 res=4</b>	<b>i1=-14 i2=4 res=4</b>
<b>3</b>	<b>a=2 b=1 i=5 k=2</b>	<b>i1=-26 i2=8 res=5</b>	<b>i1=-26 i2=8 res=5</b>
<b>4</b>	<b>a=1 b=2 i=5 k=2</b>	<b>i1=21 i2=-13 res=10</b>	<b>i1=21 i2=-13 res=10</b>
<b>5</b>	<b>a=-1 b=-2 i=5 k=2</b>	<b>i1=-26 i2=8 res=5</b>	<b>i1=-26 i2=8 res=5</b>
<b>6</b>	<b>a=-2 b=-1 i=5 k=2</b>	<b>i1=21 i2=-13 res=10</b>	<b>i1=21 i2=-13 res=10</b>

### **Обработка результатов тестирования.**

Были рассмотрены различные варианты входных данных и проверены все возможные пути работы алгоритма, на всех тестах программа отработала корректно и выдала правильные результаты.

### **Выводы.**

Я научился представлять и обрабатывать целые числа, а также организовывать ветвящиеся процессы.

### **Ответы на вопросы.**