

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: ПРЕДСТАВЛЕНИЕ И ОБРАБОТКА ЦЕЛЫХ ЧИСЕЛ.
ОРГАНИЗАЦИЯ ВЕТВЯЩИХСЯ ПРОЦЕССОВ.

Студентка гр. 0383

Преподаватель

Ханина М.И.

Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

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 18:

$$\begin{aligned} i1 = f3 &= \begin{cases} / 7 - 4*i, & \text{при } a > b \\ \backslash 8 - 6*i, & \text{при } a \leq b \end{cases} \\ i2 = f8 &= \begin{cases} / - (6*i+8), & \text{при } a > b \\ \backslash 9 - 3*(i-1), & \text{при } a \leq b \end{cases} \\ res = f6 &= \begin{cases} / |i1 - i2|, & \text{при } k < 0 \\ \backslash \max(7, |i2|), & \text{при } k \geq 0 \end{cases} \end{aligned}$$

Выполнение работы.

Числа для работы программы вводятся сразу в asm файл. Для реализации алгоритмов использовались команда сравнения `cmp` и различные условные переходы. Для функций `f3` и `f8` условия одинаковы, поэтому их вычисление проходит в одном блоке. Сначала командой `cmp` сверяются значения `a` и `b`. С помощью команды `jle` проверяется, что `a <= b`, и в зависимости от результата программа переходит к блоку, где рассчитываются соответствующие значения `f3` и `f8`. Для операций умножения использовался битовый сдвиг влево (команда `shl`) и сложение (команда `add`).

Результаты тестирования представлены в табл. 1.

Тексты исходных файлов программ см. в приложении А.

Тексты файлов диагностических сообщений см. в приложении Б.

Таблица 1. Проверка работы программы.

№	Входные данные	Значение i1	Значение i2	Значение res	Комментарий
1	a = 2, b = 3, i = 1, k = 4	2	9	9	Программа работает корректно
2	a = 1, b = 4, i = 2, k = 0	-4	6	7	Программа работает корректно
3	a = 6, b = 4, i = 3, k = -2	-5	-26	21	Программа работает корректно
4	a = 10, b = -3, i = 0, k = -10	3	-8	11	Программа работает корректно

Выводы.

В ходе выполнения данной лабораторной работы была изучена работа с целыми числами и условными переходами на языке Ассемблер.

ПРИЛОЖЕНИЕ А
ТЕКСТЫ ИСХОДНЫХ ФАЙЛОВ ПРОГРАММ

Название файла: **lr3.asm**

; Стек программы

AStack SEGMENT STACK

 DW 12 DUP(?)

AStack ENDS

;Данные программы

DATA SEGMENT

;Директивы описания данных

a DW 6

b DW 4

i DW 3

k DW -2

i1 DW 0

i2 DW 0

T DW 0

DATA ENDS

; Код программы

CODE SEGMENT

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

; Головная процедура

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
shl cx, 1 ; C = 4i
mov T, cx ; T = 4i
add T, ax
add T, ax ; T = 6i
mov bx, b
cmp a, bx
```

```
; a>b
```

```
jle f83less
```

```
mov ax, cx
mov cx, 7
sub cx, ax
mov i1, cx
add T, 8
neg T
mov cx, T
mov i2, cx
jmp final
```

```
; a<b
```

```
f83less:
```

```
mov ax, T
mov cx, 8
sub cx, ax
mov i1, cx
mov cx, i
add cx, -1
mov ax, cx
shl cx, 1
shl cx, 1
sub cx, ax
neg cx
add cx, 9
mov i2, cx
```

```
;расчет f6
```

```
final:
```

```
mov bx, k
cmp bx, 0
jl f6Second
mov bx, ax
cmp bx, 7
mov cx, i2
cmp cx, 0
jge skip2
neg cx
mov ax, cx
skip2:
jl max1
mov cx, bx ; |i2| >= 7
```

```

    jmp MainFinal
max1:
    mov cx, 7    ; |i2| < 7
    jmp MainFinal
f6Second:
    mov cx, i1
    sub cx, i2
    cmp cx, 0
    jge MainFinal
    neg cx
    jmp MainFinal
MainFinal:
    ret
Main      ENDP
CODE      ENDS
END Main

```


ПРИЛОЖЕНИЕ Б
ТЕКСТЫ ФАЙЛОВ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ

Название файла: **LR3.lst**

Microsoft (R) Macro Assembler Version 5.10

11/11/21 20:46:3

Page 1-1

```
                                ; Стек программы
0000      AStack SEGMENT STACK
0000 000C[                      DW 12 DUP(?)
                                ????
                                ]

0018      AStack ENDS

                                ;Данные программы
0000      DATA      SEGMENT

                                ;Директивы описания данны
                                x
0000 0006      a      DW  6
0002 0004      b      DW  4
0004 0003      i      DW  3
0006 FFFE      k      DW -2
0008 0000      i1     DW  0
000A 0000      i2     DW  0
000C 0000      T      DW  0

000E      DATA      ENDS

                                ; Код программы
```

```

0000                                CODE    SEGMENT

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

                                ; Головная процедура

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 D1 E1            shl  cx, 1 ; C = 4i
0016 89 0E 000C R     mov  T, cx ; T = 4i
001A 01 06 000C R     add  T, ax
001E 01 06 000C R     add  T, ax ; T = 6i
0022 8B 1E 0002 R     mov  bx, b
0026 39 1E 0000 R     cmp  a, bx

                                ; a>b

002A 7E 1F            jle  f83less
002C 8B C1            mov  ax, cx
002E B9 0007          mov  cx, 7
0031 2B C8            sub  cx, ax
0033 89 0E 0008 R     mov  i1, cx
0037 83 06 000C R 08  add  T, 8

```

003C	F7 1E 000C R	neg T
0040	8B 0E 000C R	mov cx, T
0044	89 0E 000A R	mov i2, cx

Microsoft (R) Macro Assembler Version 5.10

11/11/21 20:46:3

Page 1-2

0048	EB 25 90	jmp final
------	----------	-----------

; a<b

004B		f83less:
------	--	----------

004B	A1 000C R	mov ax, T
------	-----------	-----------

004E	B9 0008	mov cx, 8
------	---------	-----------

0051	2B C8	sub cx, ax
------	-------	------------

0053	89 0E 0008 R	mov i1, cx
------	--------------	------------

0057	8B 0E 0004 R	mov cx, i
------	--------------	-----------

005B	83 C1 FF	add cx, -1
------	----------	------------

005E	8B C1	mov ax, cx
------	-------	------------

0060	D1 E1	shl cx, 1
------	-------	-----------

0062	D1 E1	shl cx, 1
------	-------	-----------

0064	2B C8	sub cx, ax
------	-------	------------

0066	F7 D9	neg cx
------	-------	--------

0068	83 C1 09	add cx, 9
------	----------	-----------

006B	89 0E 000A R	mov i2, cx
------	--------------	------------

;расчет f6

006F		final:
------	--	--------

006F	8B 1E 0006 R	mov bx, k
------	--------------	-----------

0073	83 FB 00	cmp bx, 0
------	----------	-----------

```

0076 7C 1F                jl f6Second
0078 8B D8                mov bx, ax
007A 83 FB 07             cmp bx, 7
007D 8B 0E 000A R        mov cx, i2
0081 83 F9 00             cmp cx, 0
0084 7D 04                jge skip2
0086 F7 D9                neg cx
0088 8B C1                mov ax, cx
008A                    skip2:
008A 7C 05                jl max1
008C 8B CB                mov cx, bx        ; |i2| >= 7
008E EB 19 90            jmp MainFinal
0091                    max1:
0091 B9 0007              mov cx, 7        ; |i2| < 7
0094 EB 13 90            jmp MainFinal
0097                    f6Second:
0097 8B 0E 0008 R        mov cx, i1
009B 2B 0E 000A R        sub cx, i2
009F 83 F9 00             cmp cx, 0
00A2 7D 05                jge MainFinal
00A4 F7 D9                neg cx
00A6 EB 01 90            jmp MainFinal
00A9                    MainFinal:
00A9 CB                  ret
00AA                    Main      ENDP
00AA                    CODE      ENDS

```

END Main

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA		STACK
CODE	00AA	PARA		NONE
DATA	000E	PARA		NONE

Symbols:

N a m e	Type	Value	Attr	
A	L WORD	0000		DATA
B	L WORD	0002		DATA
F6SECOND	L NEAR	0097		CODE
F83LESS	L NEAR	004B		CODE
FINAL	L NEAR	006F		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 = 00AA

MAINFINAL L NEAR 00A9 CODE

MAX1 L NEAR 0091 CODE

SKIP2 L NEAR 008A CODE

T L WORD 000C DATA

@CPU TEXT 0101h

@FILENAME TEXT 1r3

@VERSION TEXT 510

99 Source Lines

99 Total Lines

22 Symbols

47988 + 461319 Bytes symbol space free

0 Warning Errors

0 Severe Errors