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

ОТЧЕТ

по лабораторной работе №3
по дисциплине «ОргЭВМиС»
Тема: Представление и обработка целых чисел. Организация ветвящихся процессов.

Студент гр.0382 _____ Шангичев В. А.

Преподаватель _____ Ефремов М. А.

Санкт-Петербург
2021

Цель работы.

Изучить и применить на практике навыки обработки целых чисел и организации ветвящихся процессов.

Задание.

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

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

b) 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.

№ студенческого билета = 20

f1 = -(6*I - 4), при a > b,

f1 = 3 * (i + 2), при a <= b

f2 = 2 * (i + 1) – 4, при a > b

f2 = 5 – 3 * (i + 1), при a <= b

f3 = |i1| - |i2|, при k < 0

max(4, |i2|-3), при k >= 0

Выполнение работы.

После определения сегмента стека и сегмента данных и инициализации сегментных регистров в главной функции программы реализуется операция загрузки PSP в стек и инициализация сегментного регистра данных. После

этого происходит сравнение переменных a и b. Если a > b, то последующими строками кода осуществляется запись значений в переменные i1 и i2 в соответствии с заданием, после чего осуществляется переход на метку с вычислением модуля первой функции. В противном случае выполняется код с меткой secondcase, где переменным i1 и i2 задаются значения, соответствующие второму случаю.

Блоки кода с метками absi1 и absi2 записывают в переменные i1 и i2 значение их модулей, используя, если нужно, команду neg, меняющую знак числа.

Вычисление результата происходит в блоке кода с меткой f3. Логика ветвления схожа с реализацией функций f1 и f2: значение переменной k сравнивается с нулем, и если k < 0, то ответ вычисляется в блоке negative. В противном случае код выполняется по порядку с переходом в блок exit перед блоком negative.

Тестирование.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1	a = -2 b = 4 i = 6 k = -4 answer = 8	answer = 8	Программа работает корректно
2	a = 10 b = -20 i = 7 k = -1 answer = 26	answer = 26	Программа работает корректно
3	a = -7 b = -16 i = -3 k = 1 answer = 5	answer = 5	Программа работает корректно
4	a = 7	answer = 55	Программа работает

	b = 12 i = 20 k = 2 answer = 55		корректно

Выводы.

Были изучены и применены на практике навыки обработки целых чисел и организации ветвящихся процессов.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.asm

```
; Program stack
AStack SEGMENT STACK
    DW 12 DUP(?)
AStack ENDS

; Program data
DATA SEGMENT
a DW 7
b DW 12
i DW 20
k DW 2
i1 DW 0
i2 DW 0
answer DW 0

DATA ENDS


; Program code
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, a
    cmp cx, b
    jle secondcase ; a <= b

    ; calculate i1
    mov ax, i
```

```
mov bx, i

mov cl, 03
shl ax, cl ; ax = 8i

shl bx, 1
sub ax, bx ; ax = 6i

mov i1, 4
sub i1, ax


; calculate i2
mov ax, i
sub ax, 1
shl ax, 1
mov i2, ax

jmp absi1

secondcase: ; a <= b
    mov bx, i
    mov ax, bx
    mov cl, 02
    shl bx, cl
    sub bx, ax ; bx = 3i

    ; calculate i1
 mov i1, 6
    add i1, bx

    ; calculate i2
    mov i2, 2
    sub i2, bx

absi1:
    cmp i1, 0
    jl changesigni1
    jmp absi2
```

```
    changesigni1:
        neg i1

    absi2:
        cmp i2, 0
        jl changesigni2
        jmp f3

    changesigni2:
        neg i2

    f3:
        mov ax, k
        cmp ax, 0
        jl negative ; k < 0

    kge0:
        mov ax, i2
        sub ax, 3
        cmp ax, 4
        jle write4
        mov answer, ax
        jmp exit

    write4:
        mov answer, 4
        jmp exit

    negative:
        mov ax, i1
        sub ax, i2
        mov answer, ax

    exit:
        ret
Main ENDP
CODE ENDS
 END Main
```

Файл main.lst

```
                        1
                        2                   ; Program stack
 0000                   3                       AStack SEGMENT STACK
 0000  000C[            4                           DW 12 DUP(?)
       ????             5
                        6                   ]
                        7
 0018                   8                       AStack ENDS
                        9
                       10                   ; Program data
 0000                  11                       DATA SEGMENT
 0000  0007            12                       a DW 7
 0002  000C            13                       b DW 12
 0004  0014            14                       i DW 20
 0006  0002            15                       k DW 2
 0008  0000            16                       i1 DW 0
 000A  0000            17                       i2 DW 0
 000C  0000            18                       answer DW 0
                       19
 000E                  20                       DATA ENDS
                       21
                       22
                       23                   ; Program code
 0000                  24                       CODE SEGMENT
                       25                    ASSUME CS:CODE, DS:DATA, SS:AStack
                       26
 0000                  27                       Main PROC FAR
 0000  1E              28                           push DS
 0001  2B C0           29                           sub AX,AX
 0003  50              30                           push AX
 0004  B8 ---- R       31                           mov AX,DATA
 0007  8E D8           32                           mov DS,AX
                       33
 0009  8B 0E 0000 R    34                               mov cx, a
```

```
35 000D  3B 0E 0002 R              cmp cx, b
36 0011  7E 27                jle secondcase ; a <= b
37
38                        ; calculate i1
39 0013  A1 0004 R            mov ax, i
40 0016  8B 1E 0004 R              mov bx, i
41
42 001A  B1 03                mov cl, 03
43 001C  D3 E0                shl ax, cl ; ax = 8i
44
45 001E  D1 E3                shl bx, 1
46 0020  2B C3                sub ax, bx ; ax = 6i
47
48 0022  C7 06 0008 R 0004    mov i1, 4
49 0028  29 06 0008 R              sub i1, ax
50
51
52                        ; calculate i2
53 002C  A1 0004 R            mov ax, i
54 002F  2D 0001                   sub ax, 1
```



```
55 0032  D1 E0                shl ax, 1
56 0034  A3 000A R            mov i2, ax
57
58 0037  EB 21 90                  jmp absi1
59
60 003A                       secondcase: ; a <= b
61 003A  8B 1E 0004 R                   mov bx, i
62 003E  8B C3                     mov ax, bx
63 0040  B1 02                     mov cl, 02
64 0042  D3 E3                     shl bx, cl
65 0044  2B D8                     sub bx, ax ; bx = 3i
66
67                           ; calculate i1
68 0046  C7 06 0008 R 0006      mov i1, 6
```

```
 69 004C  01 1E 0008 R                  add i1, bx
 70
 71                             ; calculate i2
 72 0050  C7 06 000A R 0002         mov i2, 2
 73 0056  29 1E 000A R                  sub i2, bx
 74
 75 005A                        absi1:
 76 005A  83 3E 0008 R 00               cmp i1, 0
 77 005F  7C 03                     jl changesigni1
 78 0061  EB 05 90                      jmp absi2
 79
 80 0064                        changesigni1:
 81 0064  F7 1E 0008 R                  neg i1
 82
 83 0068                        absi2:
 84 0068  83 3E 000A R 00               cmp i2, 0
 85 006D  7C 03                     jl changesigni2
 86 006F  EB 05 90                      jmp f3
 87
 88 0072                        changesigni2:
 89 0072  F7 1E 000A R                  neg i2
 90
 91 0076                        f3:
 92 0076  A1 0006 R                 mov ax, k
 93 0079  3D 0000                       cmp ax, 0
 94 007C  7C 1A                     jl negative ; k < 0
 95
 96 007E                        kge0:
 97 007E  A1 000A R                 mov ax, i2
 98 0081  2D 0003                       sub ax, 3
 99 0084  3D 0004                       cmp ax, 4
100 0087  7E 06                     jle write4
101 0089  A3 000C R                 mov answer, ax
102 008C  EB 14 90                     jmp exit
103
104 008F                        write4:
105 008F  C7 06 000C R 0004         mov answer, 4
106 0095  EB 0B 90                     jmp exit
107
108 0098                        negative:
```

```
          109 0098  A1 0008 R               mov ax, i1
          110 009B  2B 06 000A R                  sub ax, i2
          111 009F  A3 000C R               mov answer, ax
          112
          113 00A2                       exit:
          114 00A2  CB                        ret
          115 00A3                    Main ENDP
          116 00A3                    CODE ENDS
          117                      END Main
```


Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA	STACK	
CODE	00A3	PARA	NONE	
DATA	000E	PARA	NONE	

Symbols:

N a m e	Type	Value	Attr
A	L WORD	0000	DATA
ABSI1	L NEAR	005A	CODE
ABSI2	L NEAR	0068	CODE
ANSWER	L WORD	000C	DATA
B	L WORD	0002	DATA

```
     CHANGESIGNI1 . . . . . .      L NEAR     0064  CODE
     CHANGESIGNI2 . . . . . .      L NEAR     0072  CODE

     EXIT . . . . . .              L NEAR     00A2  CODE

     F3 . . . . . .                L NEAR     0076  CODE

     I  . . . . . .                L WORD     0004  DATA
     I1 . . . . . .                L WORD     0008  DATA
     I2 . . . . . .                L WORD     000A  DATA

     K  . . . . . .                L WORD     0006  DATA
     KGE0 . . . . . .              L NEAR     007E  CODE

     MAIN . . . . . .              F PROC     0000  CODE  Length  =
00A3

     NEGATIVE . . . . . .          L NEAR     0098  CODE

     SECONDCASE . . . . . .        L NEAR     003A  CODE

     WRITE4 . . . . . .            L NEAR     008F  CODE

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

     Microsoft      (R)      Macro      Assembler      Version      5.10
10/18/21 20:17:1

                                                                  Symbol
s-2


         114 Source  Lines
         114 Total   Lines
```

```
    26 Symbols

 47512 + 461795 Bytes symbol space free

     0 Warning Errors
0 Severe  Errors
```