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»
Тема: Представление и обработка целых чисел. Организация ветвящихся процессов.

Студент гр. 0382 _____ Кривенцова Л.С.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Изучить представление и обработку целых чисел. Научиться организовывать ветвящиеся процессы на языке Ассемблера.

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),

f1 = -(4i+3) , при a>b;

6i-10, при a<=b

f2 = 20 - 4i , при a>b;

-(6i-6) , при a<=b

f3 = |i1 - i2|, при k<0;

max(7,i2), при k>=0

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.

Порядок выполнения работы.

Перед началом выполнения функций i записывается в bx и умножается на 4, путём битового сдвига. При помощи команды *jle* программы переходят к соответствующей метке. Затем вычисляется значения функций f1 и f2. После ветвления идёт запись результата в память. Для функции res проверяется условие k >= 0. Для проверки неравенства по модулю число проверяется дважды с разным знаком.

Вывод.

Были изучены представление и обработка целых чисел. Получены знания об организации ветвящихся процессов на языке Ассемблера.

ТЕСТИРОВАНИЕ

Таблица 1. Результат тестирования.

№ т.	Входные данные	Результат	Комментарий
1	a = 1 b = 1 i = 2 k = 1	i1 = 2 i2 = -6 res = 7	Верно
2	a = 2 b = 1 i = 2 k = 1	i1 = -11 i2 = 12 res = 12	Верно
3	a = 2 b = 1 i = 2 k = -1	i1 = -11 i2 = 12 res = 23	Верно
4	a = 1 b = 1 i = 2 k = -1	i1 = 2 i2 = -6 res = 8	Верно

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММ

Файл LAB3.ASM

```
AStack SEGMENT STACK
 DW 12 DUP(?)
AStack ENDS



DATA SEGMENT
a DW 0
b DW 0
i DW 0
k DW 0
i1 DW 0
i2 DW 0
result 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

 ;Entering data
 mov a,1
 mov b,2
 mov i,3
 mov k,4




 mov DX, i
 shl DX, 1 ; = 2i
 mov AX, a
```

```
 cmp AX, b
 jle second

first:
    mov AX, DX
    shl AX, 1 ; = 4i
    mov i1, -3
    sub i1, AX ; = -3 - 4i
    add AX, i ; = 5i
    add AX, i ; = 6i
    mov i2, 4
    sub i2, AX ; = 4 - 6i
    jmp cmp_k

second:
    mov AX, DX
    add AX, i ; = 3i
    mov i2, 6
    add i2, AX ; = 3i + 6
    shl AX, 1 ; = 6i
    mov i1, -10
    add i1, AX ; = -10 + 6i
    jmp cmp_k

cmp_k:
    cmp i1, 0
    jge next
    neg i1
next:
    mov AX, k
    cmp AX, 0
    jl f3_first

f3_second:
    mov AX, i1
    cmp AX, 6
    jge case
    mov result, 6
    ret
```

```
case:
    mov result, AX
    ret

f3_first:
    cmp i2, 0
    jge f3_next
    neg i2
f3_next:
    mov AX,i1
    add AX,i2 ; = |i1| + |i2|
    mov result, AX
    ret


Main ENDP
CODE ENDS
 END Main
```

ПРИЛОЖЕНИЕ В
ДИАГНОСТИЧЕСКОЕ СООБЩЕНИЕ

Файл LAB3.lst

```
Microsoft (R) Macro Assembler Version 5.10                  11/18/21
11:50:2
                                                             Page
1-1


0000                    AStack SEGMENT STACK
0000  000C[                    DW 12 DUP(?)
       ????
                ]

0018                    AStack ENDS


0000                    DATA SEGMENT
0000  0000                    a DW 0
0002  0000                    b DW 0
0004  0000                    i DW 0
0006  0000                    k DW 0
0008  0000                    i1 DW 0
000A  0000                    i2 DW 0
000C  0000                    result DW 0
000E                    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

                         ;Entering data
0009  C7 06 0000 R 0001        mov a,1
000F  C7 06 0002 R 0002        mov b,2
0015  C7 06 0004 R 0003        mov i,3
001B  C7 06 0006 R 0004        mov k,4



0021  8B 16 0004 R              mov DX, i
0025  D1 E2                     shl DX, 1 ; = 2i
0027  A1 0000 R           mov AX, a
002A  3B 06 0002 R              cmp AX, b
002E  7E 23                     jle second

0030                    first:
0030  8B C2                       mov AX, DX
0032  D1 E0                       shl AX, 1 ; = 4i
0034  C7 06 0008 R FFFD           mov i1, -3
003A  29 06 0008 R                sub i1, AX ; = -3 - 4i
```

```
003E  03 06 0004 R            add AX, i ; = 5i
0042  03 06 0004 R            add AX, i ; = 6i
0046  C7 06 000A R 0004       mov i2, 4
004C  29 06 000A R            sub i2, AX ; = 4 - 6i
0050  EB 20 90                jmp cmp_k

0053                  second:
```



```
0053  8B C2                   mov AX, DX
0055  03 06 0004 R            add AX, i ; = 3i
0059  C7 06 000A R 0006       mov i2, 6
005F  01 06 000A R            add i2, AX ; = 3i + 6
0063  D1 E0                   shl AX, 1 ; = 6i
0065  C7 06 0008 R FFF6       mov i1, -10
006B  01 06 0008 R            add i1, AX ; = -10 + 6i
006F  EB 01 90                jmp cmp_k

0072                  cmp_k:
0072  83 3E 0008 R 00              cmp i1, 0
0077  7D 04                   jge next
0079  F7 1E 0008 R            neg i1
007D                  next:
007D  A1 0006 R           mov AX, k
0080  3D 0000                 cmp AX, 0
0083  7C 13                   jl f3_first

0085                  f3_second:
0085  A1 0008 R           mov AX, i1
0088  3D 0006                 cmp AX, 6
008B  7D 07                   jge case
008D  C7 06 000C R 0006       mov result, 6
0093  CB                  ret

0094                  case:
0094  A3 000C R           mov result, AX
0097  CB                  ret

0098                  f3_first:
0098  83 3E 000A R 00              cmp i2, 0
009D  7D 04                   jge f3_next
009F  F7 1E 000A R            neg i2
00A3                  f3_next:
00A3  A1 0008 R           mov AX,i1
00A6  03 06 000A R            add AX,i2 ; = |i1| + |i2|
00AA  A3 000C R           mov result, AX
00AD  CB                  ret


00AE                  Main ENDP
00AE                  CODE ENDS
                       END Main
```

Segments and Groups:

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

Symbols:

N a m e	Type	Value	Attr
A	L WORD	0000	DATA
B	L WORD	0002	DATA
CASE	L NEAR	0094	CODE
CMP_K	L NEAR	0072	CODE
F3_FIRST	L NEAR	0098	CODE
F3_NEXT	L NEAR	00A3	CODE
F3_SECOND	L NEAR	0085	CODE
FIRST	L NEAR	0030	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 = 00AE
NEXT	L NEAR	007D	CODE
RESULT	L WORD	000C	DATA
SECOND	L NEAR	0053	CODE
@CPU	TEXT	0101h	
@FILENAME	TEXT	lb3	
@VERSION	TEXT	510	

94 Source Lines
94 Total Lines
24 Symbols

47996 + 461311 Bytes symbol space free

0 Warning Errors
0 Severe Errors