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 _____ Крючков А.М.

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 2 DUP(?)
AStack ENDS

DATA SEGMENT
     a    DW 2
     b    DW 1
     i    DW 2
     k    DW -1
     i1   DW ?
     i2   DW ?
     res  DW ?
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
          ; f1: i1 = -(4i+3) if a>b else 6i-10
          ; f2: i2 = 20 - 4i if a>b else -(6i-6)
          mov bx, i ; bx = i
          shl bx, 1 ; bx = 2i
        shl bx, 1 ; bx = 4i
        mov ax, bx ; ax = 4i
          mov cx, a
          cmp cx, b ; cmp a b
          jle f_case2
          f_case1:
           ;f1
               add bx, 3 ; bx = 4i + 3
           neg bx ; bx = -(4i+3)
           ;f2
           neg ax ; ax = -4i
           add ax, 20 ; ax = -4i + 20
               jmp f_final
          f_case2:
               ;f1
               sub bx, i; bx = 3i
           shl bx, 1; bx = 6i
           mov ax, bx; ax = 6i !!!
           sub bx, 10 ; bx = 6i - 10
           ;f2 ax = 6i
           sub ax, 6 ; ax = 6i - 6
           neg ax; ax = -(6i - 6)
          f_final:
           mov i1, bx ; i1 = f1
           mov i2, ax ; i2 = f2
          ; f3: res = |i1 - i2| if k < 0 else max(7, |i2|)
          mov ax,i1
```

```
            mov bx,i2
            mov cx,k
            cmp k, 0
            jge case2
            case1:
                 sub ax, bx ; ax = i1 - i2
             mov res, ax
                 cmp ax, 0
                 jge final
                 neg res
                 jmp final
            case2:
             mov res, bx ;res = i2
                 cmp bx, 7 ; i2 > 7
                 jg final
             neg res
             cmp res, 7 ; 7 <= |i2|
                 jg final
             mov res, 7 ; res = 7
            final:
            ret
      Main ENDP
CODE ENDS
END Main
```

ПРИЛОЖЕНИЕ В
ДИАГНОСТИЧЕСКОЕ СООБЩЕНИЕ

Файл LAB3.lst

```
     #Microsoft      (R)      Macro      Assembler      Version      5.10
10/25/21 19:16:1
                                                                      Page
1-1


       0000                    AStack SEGMENT STACK
       0000  0002[                       DW 2 DUP(?)
              ????
                     ]

       0004                    AStack ENDS

       0000                    DATA SEGMENT
       0000  0002                        a     DW 2
       0002  0001                        b     DW 1
       0004  0002                        i     DW 2
       0006  FFFF                        k     DW -1
       0008  0000                        i1    DW ?
       000A  0000                        i2    DW ?
       000C  0000                        res   DW ?
       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
                                         ; f1: i1 = -(4i+3) if a>b else
                           6i-10
                                         ; f2: i2 = 20 - 4i if a>b else
                           -(6i-6)
       0009  8B 1E 0004 R                     mov bx, i ; bx = i
       000D  D1 E3                            shl bx, 1 ; bx = 2i
       000F  D1 E3                          shl bx, 1 ; bx = 4i
       0011  8B C3                          mov ax, bx ; ax = 4i
       0013  8B 0E 0000 R                     mov cx, a
       0017  3B 0E 0002 R                     cmp cx, b ; cmp a b
       001B  7E 0D                            jle f_case2
       001D                              f_case1:
                                          ;f1
       001D  83 C3 03                               add bx, 3 ; bx = 4i + 3
       0020  F7 DB                           neg bx ; bx = -(4i+3)
                                          ;f2
       0022  F7 D8                           neg ax ; ax = -4i
       0024  05 0014                         add ax, 20 ; ax = -4i + 20
       0027  EB 11 90                            jmp f_final
       002A                              f_case2:
                                              ;f1
       002A  2B 1E 0004 R                         sub bx, i; bx = 3i
```

```
002E  D1 E3                              shl bx, 1; bx = 6i
0030  8B C3                              mov ax, bx; ax = 6i !!!
0032  83 EB 0A                           sub bx, 10 ; bx = 6i - 10
                                       ;f2 ax = 6i
0035  2D 0006                            sub ax, 6 ; ax = 6i - 6
0038  F7 D8                              neg ax; ax = -(6i - 6)
003A                                   f_final:
#Microsoft      (R)      Macro      Assembler      Version      5.10
10/25/21 19:16:1
                                                                Page
1-2


003A  89 1E 0008 R                       mov i1, bx ; i1 = f1
003E  A3 000A R                        mov i2, ax ; i2 = f2
                                       ; f3: res = |i1 - i2| if k < 0
                             else max(7, |i2|)
0041  A1 0008 R                        mov ax,i1
0044  8B 1E 000A R                         mov bx,i2
0048  8B 0E 0006 R                         mov cx,k
004C  83 3E 0006 R 00                         cmp k, 0
0051  7D 11                                 jge case2
0053                                   case1:
0053  2B C3                                   sub ax, bx ; ax = i1 -
                             i2
0055  A3 000C R                         mov res, ax
0058  3D 0000                                 cmp ax, 0
005B  7D 21                                   jge final
005D  F7 1E 000C R                            neg res
0061  EB 1B 90                               jmp final
0064                                   case2:
0064  89 1E 000C R                           mov res, bx ;res = i2
0068  83 FB 07                               cmp bx, 7 ; i2 > 7
006B  7F 11                                  jg final
006D  F7 1E 000C R                           neg res
0071  83 3E 000C R 07                         cmp res, 7 ; 7 <= |i2|
0076  7F 06                                  jg final
0078  C7 06 000C R 0007                      mov res, 7 ; res = 7
007E                                   final:
007E  CB                               ret
007F                         Main ENDP
007F                       CODE ENDS
                           END Main
```

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0004	PARA	STACK	
CODE	007F	PARA	NONE	
DATA	000E	PARA	NONE	

Symbols:

N a m e	Type	Value	Attr
A	L WORD	0000	DATA
B	L WORD	0002	DATA
CASE1	L NEAR	0053	CODE
CASE2	L NEAR	0064	CODE
FINAL	L NEAR	007E	CODE
F_CASE1	L NEAR	001D	CODE
F_CASE2	L NEAR	002A	CODE
F_FINAL	L NEAR	003A	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 = 007F
RES	L WORD	000C	DATA
@CPU	TEXT	0101h	
@FILENAME	TEXT	LAB3	
@VERSION	TEXT	510	

77 Source Lines
77 Total Lines
22 Symbols

47998 + 459262 Bytes symbol space free

0 Warning Errors
0 Severe Errors