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.

Студент гр. 9383 _____ Ноздрин В.Я.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Изучить представление целых чисел, научиться их обрабатывать. Познакомиться с организацией ветвящихся процессов на Ассемблере.

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. из методички.

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 13:

$f1\ (a,\ b,\ i)\ =\ -(4 * i + 3),$ при $a > b$

$f1\ (a,\ b,\ i)\ =\ 6 * i - 10,$ при $a\ \leq b$

$f2\ (a,\ b,\ i)\ =\ -(6 * i + 8),$ при $a > b$

$f2\ (a,\ b,\ i)\ =\ 9 - 3 * (i - 1),$ при $a\ \leq b$

$f3\ (i1,\ i2,\ k)\ =\ |i1 + i2|,$ при $k = 0$

$f3\ (i1,\ i2,\ k)\ =\ min(i1,\ i2),$ при $k \neq 0$

Ход работы.

В ходе работы была написана программа на языке Ассемблер, которая по заданным целочисленным параметрам вычисляет значения некоторых функций. Процесс выполнения программы ветвящийся и использует следующие команды Ассемблера:

- cmp – сравнение аргументов и установка флага ZF в соответствующее результату сравнения значение. 0, если аргументы равны и 1, если аргументы не равны.
- jle – условный переход по заданной метке при условии, что в предыдущем сравнении с использованием cmp первый аргумент меньше или равен второму.

- shl – побитовый сдвиг влево. Для целых чисел применение сдвига на 1 эквивалентно умножению значения на 2.
- add – арифметическое действие сложения целых чисел
- neg – арифметическое действие взятия противоположного целого числа
- sub – арифметическое действие вычитания целых чисел
- jmp – безусловный переход по заданной метке. Передача управления.
- jne – условный переход по заданной метке при условии, что в предыдущем сравнении с использованием cmp первый аргумент не равен второму.
- jge – условный переход по заданной метке при условии, что в предыдущем сравнении с использованием cmp первый аргумент больше или равен второму.

Исходные данные заносятся в программу до выполнения, а результат работы отслеживается через отладчик.

Тестирование.

Входные данные (a, b, i, k)	Результат вычислений (i1, i2, res)
1 1 1 1	FFFC=-4 0000=0 FFFC=-4
1 1 1 0	FFFC=-4 0000=0 0004=4
2 1 1 1	FFF9=-7 FFF2=-14 FFF2=-14
2 1 1 0	FFF9=-7 FFF2=-14 0015=21

Выводы.

Изучено представление целых чисел и разработана программа, выполняющая некоторые арифметические действия над целыми числами. Программа содержит ветвящиеся процессы.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Текст файла lab3.asm

```
; Задание 13: (2, 8, 3)
; f1 (a, b)        = (a > b)?     -(4*i+3)    : 6*i-10
; f2 (a, b)        = (a > b)?     -(6*i+8)    : 9-3*(i-1)
; f3 (i1, i2, k)   = (k == 0)?    |i1+i2|     : min(i1, i2)

AStack  SEGMENT STACK
   DW  32  DUP(?)
AStack  ENDS

DATA    SEGMENT
   A   DW  1
   B   DW  1
   I   DW  1
   K   DW  1
   I1  DW  ?
   I2  DW  ?
   RES DW  ?
DATA    ENDS

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

Main    PROC FAR
   mov ax, DATA
   mov ds, ax
; f1 (a, b)        = (a > b)?     -(4*i+3)    : 6*i-10
f1  :
   mov ax, A
   cmp ax, B    ; if
   jle f1_     ; (a <= b): jmp f1_
   mov ax, I   ; ax = i
   shl ax, 1   ; ax *= 2       ax = 2*i
   shl ax, 1   ; ax *= 2       ax = 4*i
   add ax, 3   ; ax += 3       ax = 4*i+3
   neg ax      ; ax = -ax      ax = -(4*i+3)
   mov I1, ax  ; I1 = ax
   jmp f2
f1_ :          ; else
```

```
    mov ax, I   ; ax = i
    shl ax, 1   ; ax *= 2       ax = 2*i
    mov bx, ax  ; bx = ax       bx = 2*i
    shl ax, 1   ; ax *= 2       ax = 4*i
    add ax, bx  ; ax += bx      ax = 6*i
    sub ax, 10  ; ax -= 10      ax = 6*i-10
    mov I1, ax  ; I1 = ax
    jmp f2
; f2 (a, b)         = (a > b)?      -(6*i+8)    : 9-3*(i-1)
f2  :
    mov ax, A
    cmp ax, B    ; if
    jle f2_     ; (a <= b): jmp f1_
    mov ax, I   ; ax = i
    shl ax, 1   ; ax *= 2       ax = 2*i
    mov bx, ax  ; bx = ax       bx = 2*i
    shl ax, 1   ; ax *= 2       ax = 4*i
    add ax, bx  ; ax += bx      ax = 6*i
    add ax, 8   ; ax += 8       ax = 6*i+8
    neg ax      ; ax = -ax      ax = -(6*i+8)
    mov I2, ax  ; I2 = ax
    jmp f3
f2_ :           ; else
    mov ax, I   ; ax = i
    sub ax, 1   ; ax -= 1       ax = i-1
    mov bx, ax  ; bx = ax       bx = i-1
    shl ax, 1   ; ax *= 2       ax = 2*(i-1)
    add ax, bx  ; ax += bx      ax = 3*(i-1)
    mov I2, ax  ; I2 = ax
    jmp f3
; f3 (i1, i2, k)    = (k == 0)?     |i1+i2|     : min(i1, i2)
f3  :
    cmp K, 0
    jne min
    mov ax, I1  ; ax = I1
    add ax, I2  ; ax = I1 + I2
    cmp ax, 0   ; if (ax >= 0)
    jge fin     ; skip
    neg ax      ; else ax = -ax
    jmp fin
 min :
    mov ax, I1
    cmp ax, I2
    jle fin
    mov ax, I2
fin :
```

```
    mov RES, ax
    mov  ah, 4ch
    int  21h

Main    ENDP
CODE    ENDS
END Main
```

Текст файла lab3.lst

```
Microsoft (R) Macro Assembler Version 5.10          11/4/20 20:58:36
                                                    Page     1-1


                        ; Задание 13: (2, 8, 3)
                        ; f1 (a, b)       = (a > b)?     -(4*i+3)
                         : 6*i-10
                        ; f2 (a, b)       = (a > b)?     -(6*i+8)
                         : 9-3*(i-1)
                        ; f3 (i1, i2, k)   = (k == 0)?    |i1+i2|
                         : min(i1, i2)

0000                            AStack  SEGMENT STACK
0000  0020[                       DW  32  DUP(?)
  ????
                ]

0040                            AStack  ENDS

0000                            DATA    SEGMENT
0000  0001                        A   DW  1
0002  0001                        B   DW  1
0004  0001                        I   DW  1
0006  0001                        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  B8 ---- R                   mov ax, DATA
0003  8E D8                      mov ds, ax
                        ; f1 (a, b)       = (a > b)?      -(4*i+3)
                         : 6*i-10
0005                        f1  :
0005  A1 0000 R                   mov ax, A
0008  3B 06 0002 R                cmp ax, B    ; if
000C  7E 12                       jle f1_     ; (a <= b): jmp f1_
000E  A1 0004 R                   mov ax, I   ; ax = i
0011  D1 E0                       shl ax, 1   ; ax *= 2       ax = 2*i
0013  D1 E0                       shl ax, 1   ; ax *= 2       ax = 4*i
0015  05 0003                     add ax, 3   ; ax += 3       ax = 4*i+3
0018  F7 D8                       neg ax      ; ax = -ax      ax = -(4*i+3)
001A  A3 0008 R                   mov I1, ax  ; I1 = ax
001D  EB 15 90                          jmp f2
0020                        f1_ :          ; else
0020  A1 0004 R                   mov ax, I   ; ax = i
0023  D1 E0                       shl ax, 1   ; ax *= 2       ax = 2*i
0025  8B D8                       mov bx, ax  ; bx = ax       bx = 2*i
0027  D1 E0                       shl ax, 1   ; ax *= 2       ax = 4*i
0029  03 C3                       add ax, bx  ; ax += bx      ax = 6*i
002B  2D 000A                           sub ax, 10  ; ax -= 10      ax = 6*i-10
002E  A3 0008 R                   mov I1, ax  ; I1 = ax
0031  EB 01 90                          jmp f2
                        ; f2 (a, b)       = (a > b)?      -(6*i+8)
Microsoft (R) Macro Assembler Version 5.10              11/4/20 20:58:36
                                        Page     1-2


                         : 9-3*(i-1)
0034                        f2  :
0034  A1 0000 R                   mov ax, A
0037  3B 06 0002 R                cmp ax, B    ; if
003B  7E 16                       jle f2_     ; (a <= b): jmp f1_
003D  A1 0004 R                   mov ax, I   ; ax = i
0040  D1 E0                       shl ax, 1   ; ax *= 2       ax = 2*i
0042  8B D8                       mov bx, ax  ; bx = ax       bx = 2*i
0044  D1 E0                       shl ax, 1   ; ax *= 2       ax = 4*i
0046  03 C3                       add ax, bx  ; ax += bx      ax = 6*i
0048  05 0008                     add ax, 8   ; ax += 8       ax = 6*i+8
004B  F7 D8                       neg ax      ; ax = -ax      ax = -(6*i+8)
004D  A3 000A R                   mov I2, ax  ; I2 = ax
0050  EB 13 90                          jmp f3
0053                        f2_ :          ; else
0053  A1 0004 R                   mov ax, I   ; ax = i
```

```
0056  2D 0001                   sub ax, 1   ; ax -= 1       ax = i-1
0059  8B D8                    mov bx, ax  ; bx = ax       bx = i-1
005B  D1 E0                    shl ax, 1   ; ax *= 2       ax = 2*(i-1)
005D  03 C3                    add ax, bx  ; ax += bx      ax = 3*(i-1)
005F  A3 000A R                mov I2, ax  ; I2 = ax
0062  EB 01 90                 jmp f3
                        ; f3 (i1, i2, k)    = (k == 0)?     |i1+i2|
                         : min(i1, i2)
0065                    f3  :
0065  83 3E 0006 R 00          cmp K, 0
006A  75 11                    jne min
006C  A1 0008 R                mov ax, I1  ; ax = I1
006F  03 06 000A R             add ax, I2  ; ax = I1 + I2
0073  3D 0000                  cmp ax, 0   ; if (ax >= 0)
0076  7D 11                    jge fin     ; skip
0078  F7 D8                    neg ax      ; else ax = -ax
007A  EB 0D 90                 jmp fin
007D                    min :
007D  A1 0008 R                mov ax, I1
0080  3B 06 000A R             cmp ax, I2
0084  7E 03                    jle fin
0086  A1 000A R                mov ax, I2
0089                    fin :
0089  A3 000C R                mov RES, ax
008C  B4 4C                    mov  ah, 4ch
008E  CD 21                    int  21h

0090                    Main    ENDP
0090                    CODE    ENDS
                      END Main
Microsoft (R) Macro Assembler Version 5.10          11/4/20 20:58:36
                                                    Symbols-1


Segments and Groups:

                N a m e                 Length  Align   Combine Class

ASTACK . . . . . . . . . . . . .        0040    PARA    STACK
CODE . . . . . . . . . . . . . .        0090    PARA    NONE
DATA . . . . . . . . . . . . . .        000E    PARA    NONE

Symbols:

                N a m e                 Type     Value   Attr
```

```
A . . . . . L WORD      0000   DATA

B . . . . . L WORD      0002   DATA

F1 . . . . . L NEAR     0005   CODE
F1_ . . . . . L NEAR    0020   CODE
F2 . . . . . L NEAR     0034   CODE
F2_ . . . . . L NEAR    0053   CODE
F3 . . . . . L NEAR     0065   CODE
FIN . . . . . L NEAR    0089   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 = 0090
MIN . . . . . L NEAR    007D   CODE

RES . . . . . L WORD    000C   DATA

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


   91 Source  Lines
   91 Total   Lines
   23 Symbols

 48040 + 459217 Bytes symbol space free

    0 Warning Errors
    0 Severe  Errors
```