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. 9382 _____ Кузьмин Д. И.

Преподаватель _____ Ефремов М. А.

Санкт-Петербург
2020

Цель работы.

Изучить режимы принцип представления и обработки целлых чисел в языке ассемблер, а также ветвлящиеся процессы.

Основные теоретические положения.

Выполнение работы производилось при помощи эмулятора операционной системы MS-DOS, DosBox. Для отладки программы использовался инструмент afdpro.

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.

Вариант 12 (2.7.4)

f1 = -(4*i + 3), при при a>b и (6*i - 10) при a <= b

f2 = -(4*i -5) , при a>b и 10 - 3*i , при a<=b

f3 = min (|i1 - i2|, 2), при k<0 и max(-6, -i2), при k>=0

Выполнение работы.

1) Первым шагом было объявление сегмента стека и данных.
2) В сегменте данных были объявленые переменные a, b, k, I, f1, f2, f3
3) Далее при помощи оператора cmp было реализовано ветвление
4) В зависимости от значений переменных далее выполнянились различные операции. Реализовывалось ветвление при помощи команд условного перехода, а также команды jmp.

Исходный код файлов main.asm и main.lst представлен в приложении А.

Выводы.

Были изучены принципы организации ветвящихся процессов, а также представление и обработки целых чисел.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД

Файл main.asm

```
AStack SEGMENT STACK
 DW 32 DUP(?)
AStack ENDS


DATA SEGMENT

a db ?
b db ?
i db ?
k db ?
f1 dw ?
f2 dw ?
f3 dw ?

DATA ENDS

CODE SEGMENT

MAIN PROC FAR

ASSUME SS:AStack, DS:Data, CS:Code

        mov ax,data
       mov ds,ax
       mov  ax, a
       mov  bx, b
        cmp ax, bx
        jle firstf1f2

     secondf1f2:
          mov cx, i
          shl cx, 1
          shl cx, 1
          add cx, i
          add cx, i
          sub cx, 10
          mov f1, cx
          mov cx, i
          shl cx,1
          add cx, i
          neg cx
          add cx, 10
          mov f2, cx
          jmp firstf3

     firstf1f2:
          mov cx, i
          shl cx, 1
          shl cx, 1
```

```
            add cx, 3
            neg cx
            mov f1, cx
            mov cx, i
            shl cx, 1
            shl cx, 1
            sub cx, 5
            neg cx
            mov f2, cx

      firstf3:
            mov cx, k
            cmp cx, 0
            jl secondf3
            mov cx, f1
            sub cx, f2
            cmp cx, 0
            jle getreswithneg
            cmp cx, 2
            jle get1
            mov f3, 2
            jmp exit
            get1:
            mov f3, cx
            jmp exit
      getreswithneg:
            neg cx
            cmp cx, 2
            jle get1
            mov f3, cx
            jmp exit

      secondf3:
            mov cx, 6
            neg cx
            mov dx, f2
            neg dx
            cmp dx, cx
            jge getres2
            mov f3, cx
            jmp exit
            getres2:
            mov f3, dx
            jmp exit
      exit:
            mov ah, 4ch
            int 21h

MAIN ENDP

CODE ENDS

END MAIN
```

Файл main.lst

```
#Microsoft (R) Macro Assembler Version 5.10                    11/5/20
12:30:43
                                                               Page     1-1


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

 0040                   AStack ENDS


 0000                   DATA SEGMENT

 0000  00               a db ?
 0001  00               b db ?
 0002  00               i db ?
 0003  00               k db ?
 0004  0000                  f1 dw ?
 0006  0000                  f2 dw ?
 0008  0000                  f3 dw ?

 000A                   DATA ENDS

 0000                   CODE SEGMENT

 0000                   MAIN PROC FAR

                        ASSUME SS:AStack, DS:Data, CS:Code

 0000  B8 ---- R              mov ax,data
 0003  8E D8                      mov ds,ax
 0005  A1 0000 R             mov  ax, a
test.asm(26): warning A4031: Operand types must match
 0008  8B 1E 0001 R               mov  bx, b
test.asm(27): warning A4031: Operand types must match
 000C  3B C3                       cmp ax, bx
 000E  7E 2D                       jle firstf1f2

 0010                     secondf1f2:
 0010  8B 0E 0002 R                  mov cx, i
test.asm(32): warning A4031: Operand types must match
 0014  D1 E1                         shl cx, 1
 0016  D1 E1                         shl cx, 1
 0018  03 0E 0002 R                  add cx, i
test.asm(35): warning A4031: Operand types must match
 001C  03 0E 0002 R                  add cx, i
test.asm(36): warning A4031: Operand types must match
 0020  83 E9 0A                      sub cx, 10
 0023  89 0E 0004 R                  mov f1, cx
 0027  8B 0E 0002 R                  mov cx, i
test.asm(39): warning A4031: Operand types must match
 002B  D1 E1                         shl cx,1
 002D  03 0E 0002 R                  add cx, i
```

```
test.asm(41): warning A4031: Operand types must match
 0031  F7 D9                          neg cx
 0033  83 C1 0A                       add cx, 10
 0036  89 0E 0006 R                   mov f2, cx
 003A  EB 23 90                       jmp firstf3

 003D                        firstf1f2:
 003D  8B 0E 0002 R                   mov cx, i
test.asm(48): warning A4031: Operand types must match
 0041  D1 E1                          shl cx, 1
 0043  D1 E1                          shl cx, 1
 0045  83 C1 03                       add cx, 3
#Microsoft (R) Macro Assembler Version 5.10                  11/5/20
12:30:43
                                                             Page     1-2


 0048  F7 D9                          neg cx
 004A  89 0E 0004 R                   mov f1, cx
 004E  8B 0E 0002 R                   mov cx, i
test.asm(54): warning A4031: Operand types must match
 0052  D1 E1                          shl cx, 1
 0054  D1 E1                          shl cx, 1
 0056  83 E9 05                       sub cx, 5
 0059  F7 D9                          neg cx
 005B  89 0E 0006 R                   mov f2, cx

 005F                        firstf3:
 005F  8B 0E 0003 R                   mov cx, k
test.asm(62): warning A4031: Operand types must match
 0063  83 F9 00                       cmp cx, 0
 0066  7C 30                          jl secondf3
 0068  8B 0E 0004 R                   mov cx, f1
 006C  2B 0E 0006 R                   sub cx, f2
 0070  83 F9 00                       cmp cx, 0
 0073  7E 15                          jle getreswithneg
 0075  83 F9 02                       cmp cx, 2
 0078  7E 09                          jle get1
 007A  C7 06 0008 R 0002              mov f3, 2
 0080  EB 33 90                       jmp exit
 0083                           get1:
 0083  89 0E 0008 R                        mov f3, cx
 0087  EB 2C 90                            jmp exit
 008A                        getreswithneg:
 008A  F7 D9                               neg cx
 008C  83 F9 02                            cmp cx, 2
 008F  7E F2                               jle get1
 0091  89 0E 0008 R                        mov f3, cx
 0095  EB 1E 90                            jmp exit

 0098                        secondf3:
 0098  B9 0006                             mov cx, 6
 009B  F7 D9                               neg cx
 009D  8B 16 0006 R                        mov dx, f2
 00A1  F7 DA                               neg dx
 00A3  3B D1                               cmp dx, cx
 00A5  7D 07                               jge getres2
 00A7  89 0E 0008 R                        mov f3, cx
```

```
 00AB  EB 08 90                         jmp exit
 00AE                             getres2:
 00AE  89 16 0008 R                     mov f3, dx
 00B2  EB 01 90                         jmp exit
 00B5                         exit:
 00B5  B4 4C                            mov ah, 4ch
 00B7  CD 21                            int 21h

 00B9                 MAIN ENDP

 00B9                 CODE ENDS

                      END MAIN
#Microsoft (R) Macro Assembler Version 5.10                 11/5/20
12:30:43
                                                            Symbols-1


Segments and Groups:

                N a m e         Length     Align    Combine Class

ASTACK . . . . . .              0040  PARA  STACK
CODE . . . . . .                00B9  PARA  NONE
DATA . . . . . .                000A  PARA  NONE

Symbols:

                N a m e         Type  Value     Attr

A  . . . . . .                  L BYTE    0000  DATA

B  . . . . . .                  L BYTE    0001  DATA

EXIT . . . . . .                L NEAR    00B5  CODE

F1 . . . . . .                  L WORD    0004  DATA
F2 . . . . . .                  L WORD    0006  DATA
F3 . . . . . .                  L WORD    0008  DATA
FIRSTF1F2  . . . . . .          L NEAR    003D  CODE
FIRSTF3  . . . . . .            L NEAR    005F  CODE

GET1 . . . . . .                L NEAR    0083  CODE
GETRES2  . . . . . .            L NEAR    00AE  CODE
GETRESWITHNEG  . . . . . .      L NEAR    008A  CODE

I  . . . . . .                  L BYTE    0002  DATA

K  . . . . . .                  L BYTE    0003  DATA

MAIN . . . . . .                F PROC    0000  CODE  Length = 00B9

SECONDF1F2 . . . . . .          L NEAR    0010  CODE
SECONDF3 . . . . . .            L NEAR    0098  CODE

@CPU . . . . . .                TEXT  0101h
@FILENAME  . . . . . .          TEXT  test
@VERSION . . . . . .            TEXT  510
```

```
  103 Source  Lines
  103 Total   Lines
   24 Symbols

48058 + 461249 Bytes symbol space free

   10 Warning Errors
    0 Severe  Errors
```