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.

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

   firstf1:
          cmp   ax, bx
          jle secondf1
          mov cx, i
          shl cx, 1
          shl cx, 1
          add cx, 3
          neg cx
          mov f1, cx
          jmp firstf2

     secondf1:
          mov cx, i
          shl cx, 1
          shl cx, 1
          add cx, i
          add cx, i
          sub cx, 10
          mov f1, cx
          jmp firstf2
```

```
firstf2:
     cmp   ax, bx
     jg secondf2
     mov cx, i
     mov cx, 2
     sal cx, 1
     sub cx, 5
     mov f2, cx
     jmp firstf3

secondf2:

     mov cx, i
     sal cx,1
     add cx, i
     add cx, 10
     mov f2, cx
     jmp firstf3

firstf3:

     mov cx, k
     cmp cx, 0
     jl secondf3
     mov cx, f1
     sub cx, f2
     cmp cx, 0
     cmp cx, 2
     jle getres1
     getres1:
     mov ax, cx
     jmp exit
     mov ax, 2
     jl abs_
     abs_:
     neg cx

secondf3:
     mov cx, 6
     neg cx
     mov dx, f2
     neg dx
     cmp dx, cx
     jge getres2
     getres2:
     mov ax, cx
     jmp exit
     mov ax, cx
     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
09:57:23
                                                            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                     firstf1:
 000C  3B C3                              cmp   ax, bx
 000E  7E 14                              jle secondf1
 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  83 C1 03                           add cx, 3
 001B  F7 D9                              neg cx
 001D  89 0E 0004 R                       mov f1, cx
 0021  EB 1B 90                           jmp firstf2

 0024                       secondf1:
 0024  8B 0E 0002 R                       mov cx, i
test.asm(41): warning A4031: Operand types must match
 0028  D1 E1                              shl cx, 1
 002A  D1 E1                              shl cx, 1
```

```
 002C  03 0E 0002 R                    add cx, i
test.asm(44): warning A4031: Operand types must match
 0030  03 0E 0002 R                    add cx, i
test.asm(45): warning A4031: Operand types must match
 0034  83 E9 0A                        sub cx, 10
 0037  89 0E 0004 R                    mov f1, cx
 003B  EB 01 90                        jmp firstf2


 003E                       firstf2:
#Microsoft (R) Macro Assembler Version 5.10                 11/5/20
09:57:23
                                                            Page     1-2


 003E  3B C3                           cmp   ax, bx
 0040  7F 13                           jg secondf2
 0042  8B 0E 0002 R                    mov cx, i
test.asm(54): warning A4031: Operand types must match
 0046  B9 0002                         mov cx, 2
 0049  D1 E1                           sal cx, 1
 004B  83 E9 05                        sub cx, 5
 004E  89 0E 0006 R                    mov f2, cx
 0052  EB 15 90                        jmp firstf3

 0055                       secondf2:

 0055  8B 0E 0002 R                    mov cx, i
test.asm(63): warning A4031: Operand types must match
 0059  D1 E1                           sal cx,1
 005B  03 0E 0002 R                    add cx, i
test.asm(65): warning A4031: Operand types must match
 005F  83 C1 0A                        add cx, 10
 0062  89 0E 0006 R                    mov f2, cx
 0066  EB 01 90                        jmp firstf3

 0069                       firstf3:

 0069  8B 0E 0003 R                    mov cx, k
test.asm(72): warning A4031: Operand types must match
 006D  83 F9 00                        cmp cx, 0
 0070  7C 1C                           jl secondf3
 0072  8A 0E 0004 R                    mov cl, f1
test.asm(75): warning A4031: Operand types must match
 0076  2A 0E 0006 R                    sub cl, f2
test.asm(76): warning A4031: Operand types must match
 007A  80 F9 00                        cmp cl, 0
 007D  80 F9 02                        cmp cl, 2
 0080  7E 00                           jle getres1
 0082                             getres1:
 0082  8B C1                           mov ax, cx
 0084  EB 21 90                        jmp exit
 0087  B8 0002                         mov ax, 2
 008A  7C 00                           jl abs_
 008C                             abs_:
 008C  F7 D9                           neg cx

 008E                       secondf3:
```

```
 008E  B9 0006                          mov cx, 6
 0091  F7 D9                            neg cx
 0093  8B 16 0006 R                     mov dx, f2
 0097  F7 DA                            neg dx
 0099  3B D1                            cmp dx, cx
 009B  7D 00                            jge getres2
 009D                              getres2:
 009D  8B C1                            mov ax, cx
 009F  EB 06 90                         jmp exit
 00A2  8B C1                            mov ax, cx
 00A4  EB 01 90                         jmp exit

 00A7                         exit:
 00A7  B4 4C                            mov ah, 4ch
 00A9  CD 21                            int 21h

 00AB                   MAIN ENDP
#Microsoft (R) Macro Assembler Version 5.10                 11/5/20
09:57:23
                                                            Page     1-3


 00AB                   CODE ENDS

                        END MAIN
#Microsoft (R) Macro Assembler Version 5.10                 11/5/20
09:57:23
                                                            Symbols-1


Segments and Groups:

                N a m e         Length     Align     Combine Class

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

Symbols:

                N a m e         Type  Value     Attr

A  . . . . . .                  L BYTE     0000  DATA
ABS_ . . . . . .                L NEAR     008C  CODE

B  . . . . . .                  L BYTE     0001  DATA

EXIT . . . . . .                L NEAR     00A7  CODE

F1 . . . . . .                  L WORD     0004  DATA
F2 . . . . . .                  L WORD     0006  DATA
F3 . . . . . .                  L WORD     0008  DATA
FIRSTF1  . . . . . .            L NEAR     000C  CODE
FIRSTF2  . . . . . .            L NEAR     003E  CODE
FIRSTF3  . . . . . .            L NEAR     0069  CODE

GETRES1  . . . . . .            L NEAR     0082  CODE
```

```
GETRES2  . . . . . .     L NEAR     009D  CODE

I  . . . . . .           L BYTE     0002  DATA

K  . . . . . .           L BYTE     0003  DATA

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

SECONDF1 . . . . . .     L NEAR     0024  CODE
SECONDF2 . . . . . .     L NEAR     0055  CODE
SECONDF3 . . . . . .     L NEAR     008E  CODE

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

#Microsoft (R) Macro Assembler Version 5.10                  11/5/20
09:57:23

                                                             Symbols-2




    109 Source  Lines
    109 Total   Lines
     26 Symbols

  48058 + 461249 Bytes symbol space free

     12 Warning Errors
      0 Severe  Errors
```