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. 0383 _____ Позолотин К.С.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

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.

Замечания: 1) при разработке программы нельзя использовать фрагменты, представленные на ЯВУ, в частности, для ввода-вывода данных. Исходные данные должны вводиться, а результаты контролироваться в режиме отладки; 2) при вычислении функций f1 и f2 вместо операции умножения следует использовать арифметический сдвиг и, возможно, сложение; 3) при вычислении функций f1 и f2 нельзя использовать процедуры; 4) при разработке программы следует минимизировать длину кода, для чего, если надо, следует преобразовать исходные выражения для вычисления функций.

Ход работы.

Вариант 19 (4.5.7):

$$f4 = \begin{cases} -(6*i - 4), & \text{при } a>b \\ 3*(i+2), & \text{при } a<=b \end{cases}$$

$$f5 = \begin{cases} 20 - 4*i, & \text{при } a>b \\ -(6*I - 6), & \text{при } a<=b \end{cases}$$

$$f7 = \begin{cases} / |i1| + |i2|, & \text{при } k<0 \\ \backslash \max(6, |i1|), & \text{при } k>=0 \end{cases}$$

Для сравнения использовалась команды cmd (выполняющий сравнение и в результате изменяя флаги) и команды jle, jge, работающие с положительными и отрицательными числами, проверяющие флаги, в которые внесла изменения команда cmd.

Команды jle, jge выполняли (или не выполняли) короткий переход на метки, в результате чего выполнялись именно те команды, которые нужны для вычисления значения функции при данных условиях.

Табл.1: Тестирование работы lab3.asm

Значения a, b, i, k	Результат работы f4 (в i1)	Результат работы f5 (в i2)	Результат работы f7 (в CX)	Комментарий
a = 3 b = 5 i = 8 k = 13	FFBE = -42	FFBE = 30	002A = 42	Верно
a = 3 b = 5 i = 8 k = -1	FFBE = -42	FFBE = 30	0006 = 6	Верно
a = 5 b = 3 i = 8 k = 2	FFF4 = -12	FFD4 = -44	002C = 44	Верно

a = 5 b = 3 i = 8 k = -2	FFF4 = -12	FFD4 = -44	0006 = 6	Верно

Компоненты программы см. в приложении А

Выводы.

В ходе выполнения работы были изучены способы работы с целыми положительными и отрицательными числами, изучены условные переходы и реализованы сравнения на языке Ассемблер.

ПРИЛОЖЕНИЕ А

Тексты компонентов программы lab3.exe

Lab3.asm:

```
AStack SEGMENT  STACK
   DW 12 DUP(?)
AStack  ENDS
;Данные программы
DATA      SEGMENT
;Директивы описания данных
a      DW    0
b      DW    0
i      DW    0
k      DW    13
i1     DW    0
i2     DW    0

DATA      ENDS

; Код программы
CODE      SEGMENT
     ASSUME CS:CODE, DS:DATA, SS:AStack

; Головная процедура
Main      PROC  FAR
     push  DS
     sub   AX,AX
     push  AX
     mov   AX,DATA
     mov   DS,AX
          mov   CX, 0

          mov a, 3
          mov b, 5
          mov i, 1
          mov k, 2

          ;f1&f2
          mov cx, i
          shl cx, 1 ; cx = 2*i
          mov ax, cx ; ax = 2*i
          mov bx, b

          cmp a, bx ; сравнение a и b
          jle f1f2
                neg cx ; a > b   -2i
                add cx, cx ; -2i + -2i
                add cx, 20 ; -4i + 20
                mov i1, cx ; i1 = -4i + 20
```

```
        mov i2, cx ; -4i + 20
        sub i2, ax ; -4i + 20 - 2i
        sub i2, 16 ; i2 = -6i + 4
        jmp f1f2End

f1f2: ; a <= b
        add cx, i ; cx = 3i
        mov ax, cx
        add cx, cx ; cx = 6i
        neg cx ; -6i
        mov i1, cx ;
        add i1, 6 ; i1 = -6i + 6

        add ax, 6
        mov i2, ax ; i2 = 3i + 6
f1f2End:

;f4
mov cx, i1 ; cx = i1


cmp cx, 0
jge module ; cx < 0
        neg cx
module: ; cx = |i1|

mov ax, i2

cmp ax, 0
jge module1 ; ax < 0
        neg ax
module1: ; ax = |i2|


mov bx, k


cmp bx, 0 ; сравнение k и 0 id 83
jge f4 ;k < 0
        add cx, ax
        jmp MainEnd
f4:       ; k >= 0
        mov bx, 6 ;
```

```
                cmp cx, bx
                jge max ; cx < bx
                  mov cx, bx
                max:



        MainEnd: ; в cx лежит значение функции f4
    ret
Main    ENDP
CODE    ENDS
END Main
```

Lab3.lst

```
  Microsoft (R) Macro Assembler Version 5.10                11/18/21 02:35:4
                                                            Page     1-1


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

 0018                                                                AStack
ENDS

                                                                    ;Данные
программы
 0000                                                                DATA
SEGMENT

                                                                ;Директивы описания данны
                                                                    x
 0000  0000                                                     a     DW    0
 0002  0000                                                     b     DW    0
 0004  0000                                                     i     DW    0
 0006  000D                                                     k     DW    13
 0008  0000                                                     i1    DW    0
```

```
 000A  0000                                    i2     DW    0

 000C                                          DATA
ENDS

                                              ; Код
программы
 0000                                          CODE
SEGMENT

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

                                              ; Головная
процедура
 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
 0009  B9 0000                                       mov   CX,
0

 000C  C7 06 0000 R 0003                     mov a, 3
 0012  C7 06 0002 R 0005                     mov b, 5
 0018  C7 06 0004 R 0001                     mov i, 1
 001E  C7 06 0006 R 0002                     mov k, 2


;f1&f2
 0024  8B 0E 0004 R                            mov cx, i
 0028  D1 E1                                         shl cx, 1 ;
cx = 2*i
 002A  8B C1                                         mov ax, cx
; ax = 2*i
 002C  8B 1E 0002 R                            mov bx, b
```

```
 0030  39 1E 0000 R                                    cmp a, bx ;
сравнение a и b
 0034  7E 1B                                              jle f1f2
 0036  F7 D9                                                  neg
cx ; a > b   -2i
 0038  03 C9                                                  add
cx, cx ; -2i + -2i
 003A  83 C1 14                                               add
cx, 20 ; -4i + 20
 003D  89 0E 0008 R                                      mov i1, cx ;
i1 = -4i + 20


 0041  89 0E 000A R                                      mov i2, cx ;
-4i + 20
 0045  29 06 000A R                                      sub i2, ax ;
-4i + 20 - 2i
 0049  83 2E 000A R 10                                   sub i2, 16 ;
i2 = -6i + 4
   Microsoft (R) Macro Assembler Version 5.10          11/18/21 02:35:4
                                                            Page     1-2


 004E  EB 1A 90                                               jmp
f1f2End


 0051
f1f2: ; a <= b
 0051  03 0E 0004 R                                      add cx, i ;
cx = 3i
 0055  8B C1                                                  mov
ax, cx
 0057  03 C9                                                  add
cx, cx ; cx = 6i
 0059  F7 D9                                                  neg
cx ; -6i
 005B  89 0E 0008 R                                      mov i1, cx ;
 005F  83 06 0008 R 06                                   add i1, 6 ;
i1 = -6i + 6
```

```
 0064  05 0006                                    add
ax, 6
 0067  A3 000A R                                 mov i2, ax ;
i2 = 3i + 6
 006A
f1f2End:


                                                 ;f4
 006A  8B 0E 0008 R                              mov cx, i1 ; cx =
i1



 006E  83 F9 00                                  cmp cx, 0
 0071  7D 02                                     jge
module ; cx < 0
 0073  F7 D9                                     neg
cx
 0075
module: ; cx = |i1|

 0075  A1 000A R                                 mov ax, i2

 0078  3D 0000                                   cmp ax, 0
 007B  7D 02                                     jge
module1 ; ax < 0
 007D  F7 D8                                     neg
ax
 007F
module1: ; ax = |i2|


 007F  8B 1E 0006 R                              mov bx, k


 0083  83 FB 00                                  cmp bx, 0
; сравнение k и 0
                                                 id 83
```

```
 0086  7D 05                                                    jge f4 ;k <
0
 0088  03 C8                                                       add
cx, ax
 008A  EB 0A 90                                                    jmp
MainEnd
 008D                                                               f4:
; k >= 0
 008D  BB 0006                                                     mov
bx, 6 ;
 0090  3B CB
                                                        cmp cx, bx
 0092  7D 02
                                                        jge max ; cx < bx
 0094  8B CB
                                                         mov cx, bx
 0096
                                                              max:



 0096
MainEnd: ; в cx лежит значЙ
                                                                 µние
функции f4
 0096  CB                                                      ret
 0097                                                           Main
ENDP
 0097                                                           CODE
ENDS
   Microsoft (R) Macro Assembler Version 5.10          11/18/21 02:35:4
                                                        Page     1-3


                                                                 END Main
   Microsoft (R) Macro Assembler Version 5.10          11/18/21 02:35:4
                                                        Symbols-1
```

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA	STACK	
CODE	0097	PARA	NONE	
DATA	000C	PARA	NONE	

Symbols:

N a m e	Type	Value	Attr
A	L WORD	0000	DATA
B	L WORD	0002	DATA
F1F2	L NEAR	0051	CODE
F1F2END	L NEAR	006A	CODE
F4	L NEAR	008D	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 = 0097
MAINEND	L NEAR	0096	CODE
MAX	L NEAR	0096	CODE
MODULE	L NEAR	0075	CODE
MODULE1	L NEAR	007F	CODE

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


  103 Source  Lines
  103 Total   Lines
   22 Symbols

 48056 + 461251 Bytes symbol space free

    0 Warning Errors
    0 Severe  Errors
```