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(ка) гр. 9382 _____ Голубева В.П.
Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2020

Цель работы.

Научиться программировать ветвящиеся процессы.

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);

1.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.

Вариат 4

```
        / 15-2*i , при a>b
f1 = <
        \ 3*i+4 , при a<=b

        / 20 - 4*i , при  a>b
f5= <
        \ -(6*I - 6), при  a<=b

       / min (|i1 - i2|, 2), при  k<0
f4 = <
       \ max( -6, -i2), при  k>=0
```

Выполнение работы.

Переменные a, b, i, i1, i2, res объявлены в сегменте данных. Ветвление было организовано с помощь меток. При помощи команды cmp и jle(jump less equal) был осуществлён переход на нужную метку.

Тестирование.

№	Входные данные	Выходные данные	Правильный результат
1	a=1, b=1, i=1, k=0	i1=7, i2=0, res=0	i1=7, i2=0, res=0
2	a=1, b=1, i=1, k=-1	i1=7, i2=0, res=2	i1=7, i2=0, res=2
3	a=2, b=1, i=1, k=0	i1=13, i2=16, res=-6	i1=13, i2=16, res=-6
4	a=2, b=1, i=1, k=-1	i1=13, i2=16, res=2	i1=13, i2=16, res=2

Выводы.

Было изучено программирование ветвящихся процессов, написана программа, которая которая по заданным челоцисленным параметрам вычисляет значение функции.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.asm

```
STACKSG SEGMENT  PARA STACK 'Stack'
          DW       32 DUP(?)
STACKSG    ENDS

DATASG  SEGMENT  PARA 'Data'                    ;SEG DATA
     a         DW     1h
     b         DW     1h
     i         DW     1h
     k       DW  1h
     i1      DW  1h
     i2      DW  1h
     res     DW  1h
DATASG     ENDS                                    ;ENDS DATA

CODE      SEGMENT                                ;SEG CODE
ASSUME  DS:DATASG, CS:CODE

Main    PROC  FAR
      mov  ax, DATASG
      mov  ds, ax

f1:
     mov ax, a         ; переменная a в ax
     cmp ax, b          ;   сравниваем   переменные   a   и   b
соответственно
     jle f1_1           ; a <= b

                        ; если попали сюда, то a > b
     mov ax, i          ; переменная i в ax
     shl ax, 1          ; умножим i на 2
     mov bx, 15h         ; кладем в bx 15
     sub bx, ax         ; bx - ax = 15 - i * 2
     mov i1, bx         ; записываем в i1 результат 15 - i * 2
```

```
     jmp f2              ; переходим к f2

f1_1:                    ; a <= b

     mov ax, i          ; переменная i в ax
     mov bx, i          ; переменная i в bx

     shl ax, 1         ; умножим i на 2

     add ax, bx ;получаем 3*i

     mov bx, 4h ; кладем в bx 4
     add ax, bx ;ax+4

     mov i1, ax ; записываем в i1 результат 3*i+4

     jmp f2

f2:
     mov ax, a           ; переменная a в ax
     cmp ax, b           ; сравниваем переменные a и b
     jle f2_1            ; a <= b



     mov ax, i          ; переменная i в ax
     mov bx, 20h ; кладем в bx 20
     shl ax, 1          ; умножим i на 2
     shl ax, 1          ; умножим i на 2

     mov i2, bx ; кладем в i2 20
     sub i2, ax ;i2=20-4i

     jmp f3              ; переходим к f3

f2_1:
     mov ax, i  ; переменная i в ax
```

```
        mov bx, 1h ; кладем в bx 1
        sub ax, bx ;ax=i-1

        shl ax, 1       ;ax=(i-1)2
        mov bx, ax ;bx=(i-1)2
        shl ax, 1       ;ax=(i-1)4
        add ax, bx ;ax=(i-1)6
        neg ax          ;ax=-(i-1)6

        mov i2, ax ;i2=bx

        jmp f3

    f3:
        mov ax, k           ; кладем в ax переменную k
        cmp ax, 0           ; сравним k с 0
        jge f3_1            ; k >= 0

                            ; если оказались здесь, то k < 0
        mov ax, i1          ; кладем в ax переменную i1
        sub ax, i2          ; ax = i1 - i2

        cmp ax, 0           ; сравним i1 - i2 с нулем
        jl f3_ABS           ; если i1 - i2 < 0, то стоит взять модуль
        jmp f3_result   ; переход в f3_result

    f3_ABS:
        neg ax              ; взяли модуль i1 - i2

    f3_result:
        cmp ax, 2h          ; сравним |i1 - i2| с 2
          jge f3_jl_result; если |i1 - i2| >= 2, переместимся в
f3_jl_result
        mov res, ax         ; |i1 - i2| < 2 => res = |i1 - i2|
        jmp end_f           ; завершаем программу

    f3_jl_result:
```

```
     mov res, 2h         ; |i1 - i2| >= 2 => res = 2
     jmp end_f           ; завершаем программу

f3_1:                    ; k >= 0
     mov ax, i2          ; кладем в ax переменную i2
     neg ax              ; ax = -ax
     cmp ax, -6h         ; сравниваем ax, -6
     jle f3_jge     ; если -i2 <= -6, переместимся в f3_jge
     mov res, ax         ; -i2 > -6 => res = -i2
     jmp end_f           ; завершаем программу

f3_jge:
     mov res, -6h        ; -i2 <= -6 => res = -6

end_f:
     mov ah, 4ch         ; завершим программу
     int 21h


Main      ENDP
CODE      ENDS
END Main                   ;ENDS CODE
```

ПРИЛОЖЕНИЕ Б

ЛИСТИНГ ПРОГРАММЫ

```
#Microsoft (R) Macro Assembler Version 5.10              10/22/20 17:40:0
                                                              Page     1-1


0000                            STACKSG SEGMENT  PARA STACK 'Stack'
0000  0020[                             DW       32 DUP(?)
        ????
                                ]

0040                            STACKSG       ENDS

0000                            DATASG  SEGMENT  PARA 'Data'
                                                ;SEG DATA
0000  0001                        a         DW  1h
0002  0001                        b         DW  1h
0004  0001                        i         DW  1h
0006  0001                        k      DW  1h
0008  0001                        i1     DW  1h
000A  0001                        i2     DW  1h
000C  0001                        res    DW  1h
000E                            DATASG        ENDS
                                                        ;ENDS DATA

0000                            CODE      SEGMENT
                                                      ;SEG CODE
                                ASSUME  DS:DATASG, CS:CODE

0000                            Main    PROC  FAR
0000  B8 ---- R                 mov  ax, DATASG
0003  8E D8                             mov  ds, ax

0005                            f1:
0005  A1 0000 R                   mov ax, a             ; Ð¿ÐµÑ ÐµÐŒÐµÐœÐ°Ñ
                                 a Ð² ax
```

```
0008  3B 06 0002 R                cmp ax, b               ; Ñ) Ñ( Ð°Ð²ÐœÐžÐ²Ð°ÐµÐŒ
                                  Ð¿ÐµÑ(ÐµÐŒÐµÐœÐœÑ)Ðµ a Ðž b
Ñ)ÐŸÐŸÑ(Ð²ÐµÑ(Ñ)Ñ(
                                  Ð²ÐµÐœÐœÐŸ
000C  7E 11                     jle f1_1        ; a <= b

                                        ; ÐµÑ)Ð»Ðž Ð¿ÐŸÐ¿Ð°Ð»Ðž
                                  Ñ)Ñ ÐŽÐ°, Ñ(ÐŸ a > b
000E  A1 0004 R                  mov ax, i       ; Ð¿ÐµÑ(ÐµÐŒÐµÐœÐœÐ°Ñ
                                  i Ð² ax
0011  D1 E0                      shl ax, 1       ; Ñ)ÐŒÐœÐŸÐ¶ÐžÐŒ i ÐœÐ°
                                  2
0013  BB 0015                    mov bx, 15h     ; ÐºÐ»Ð°ÐŽÐµÐŒ Ð² bx 1
                                  5
0016  2B D8                      sub bx, ax      ; bx - ax = 15 - i * 2
0018  89 1E 0008 R                mov i1, bx      ; Ð·Ð°Ð¿ÐžÑ)Ñ)Ð²Ð°ÐµÐŒ
                                  Ð² i1 Ñ(ÐµÐ·Ñ)Ð»Ñ‡Ñ(Ð°Ñ( 15 - i * 2
001C  EB 17 90                        jmp f2          ; Ð¿ÐµÑ(ÐµÑ)ÐŸÐŽÐžÐŒ Ðº
                                  f2

001F                              f1_1:           ; a <= b

001F  A1 0004 R                   mov ax, i               ; Ð¿ÐµÑ(ÐµÐŒÐµÐœÐœÐ°Ñ
                                  i Ð² ax
0022  8B 1E 0004 R                mov bx, i               ; Ð¿ÐµÑ(ÐµÐŒÐµÐœÐœÐ°Ñ
#Microsoft (R) Macro Assembler Version 5.10                 10/22/20 17:40:0
                                        Page     1-2


                                  i Ð² bx

0026  D1 E0                       shl ax, 1               ; Ñ)ÐŒÐœÐŸÐ¶ÐžÐŒ i ÐœÐ°
                                  2

0028  03 C3                       add ax, bx    ;Ð¿ÐŸÐ»Ñ)Ñ›Ð°ÐµÐŒ 3*i

002A  BB 0004                          mov bx, 4h   ; ÐºÐ»Ð°ÐŽÐµÐŒ Ð² bx 4
```

```
002D  03 C3                   add ax, bx    ;ax+4

002F  A3 0008 R               mov i1, ax    ; Ð·Ð°Ð¿ÐžÑ)Ñ]Ð²Ð°ÐµÐŒ Ð² i1 Ñ(
                              ÐµÐ·Ñ)Ð»Ñ{Ñ(Ð°Ñ( 3*i+4

0032  EB 01 90                    jmp f2

0035                          f2:
0035  A1 0000 R               mov ax, a         ; Ð¿ÐµÑ(ÐµÐŒÐµÐœÐ°Ñ
                              a Ð² ax
0038  3B 06 0002 R            cmp ax, b         ; Ñ)Ñ(Ð°Ð²ÐœÐžÐ²Ð°ÐµÐŒ
                              Ð¿ÐµÑ(ÐµÐŒÐµÐœÐ]Ðµ a Ðž b
003C  7E 15                   jle f2_1          ; a <= b



003E  A1 0004 R               mov ax, i                 ; Ð¿ÐµÑ(ÐµÐŒÐµÐœÐ°Ñ
                               i Ð² ax
0041  BB 0020                 mov bx, 20h  ; ÐºÐ»Ð°ÐŽÐµÐŒ Ð² bx 2
                              0
0044  D1 E0                   shl ax, 1                 ; Ñ)ÐŒÐœÐŸÐ¶ÐžÐŒ i ÐœÐ
                              ° 2
0046  D1 E0                   shl ax, 1                 ; Ñ)ÐŒÐœÐŸÐ¶ÐžÐŒ i ÐœÐ
                              ° 2

0048  89 1E 000A R            mov i2, bx    ; ÐºÐ»Ð°ÐŽÐµÐŒ Ð² i2 20
004C  29 06 000A R            sub i2, ax    ;i2=20-4i

0050  EB 19 90                     jmp f3          ; Ð¿ÐµÑ(ÐµÑ)ÐŸÐŽÐžÐŒ Ðº
                               f3

0053                          f2_1:
0053  A1 0004 R               mov ax, i     ; Ð¿ÐµÑ(ÐµÐŒÐµÐœÐ°Ñ    i Ð² ax
0056  BB 0001                 mov bx, 1h    ; ÐºÐ»Ð°ÐŽÐµÐŒ Ð² bx 1
0059  2B C3                   sub ax, bx    ;ax=i-1

005B  D1 E0                   shl ax, 1                ;ax=(i-1)2
```

```
005D  8B D8                   mov bx, ax   ;bx=(i-1)2
005F  D1 E0                   shl ax, 1            ;ax=(i-1)4
0061  03 C3                   add ax, bx   ;ax=(i-1)6
0063  F7 D8                   neg ax               ;ax=-(i-1)6

0065  A3 000A R               mov i2, ax   ;i2=bx

0068  EB 01 90                     jmp f3

#Microsoft (R) Macro Assembler Version 5.10          10/22/20 17:40:0
                                     Page     1-3


006B                          f3:
006B  A1 0006 R                  mov ax, k         ; ĐºĐ»Đ°ĐŽĐµĐŒ Đ² ax Đ¿
                              ĐµÑ(ĐµĐŒĐµĐœĐœÑ)Ñ    k
006E  3D 0000                    cmp ax, 0         ; Ñ)Ñ(Đ°Đ²ĐœĐžĐŒ k Ñ) 0
0071  7D 25                      jge f3_1          ; k >= 0

                                       ; ĐµÑ)Đ»Đž ĐŸĐºĐ°Đ·Đ°Đ»
                              ĐžÑ)Ñ{ Đ·ĐŽĐµÑ)Ñ{, Ñ(ĐŸ k < 0
0073  A1 0008 R                  mov ax, i1        ; ĐºĐ»Đ°ĐŽĐµĐŒ Đ² ax Đ¿
                              ĐµÑ(ĐµĐŒĐµĐœĐœÑ)Ñ    i1
0076  2B 06 000A R               sub ax, i2        ; ax = i1 - i2

007A  3D 0000                    cmp ax, 0         ; Ñ)Ñ(Đ°Đ²ĐœĐžĐŒ i1 - i
                              2 Ñ) ĐœÑ)Đ»ĐµĐŒ
007D  7C 03                      jl f3_ABS         ; ĐµÑ)Đ»Đž i1 - i2 < 0,
                               Ñ(ĐŸ Ñ)Ñ(ĐŸĐžÑ( Đ²Đ·Ñ   Ñ(Ñ{ ĐŒĐŸĐŽÑ)Đ»Ñ{
007F  EB 03 90                         jmp f3_result   ; Đ¿ĐµÑ(ĐµÑ)ĐŸĐŽ Đ² f3_res
                              ult

0082                          f3_ABS:
0082  F7 D8                      neg ax            ; Đ²Đ·Ñ   Đ»Đž ĐŒĐŸĐŽÑ)Đ»
                              Ñ{ i1 - i2

0084                          f3_result:
```

```
0084  3D 0002                   cmp ax, 2h        ; Ñ)Ñ(Ð°Ð²ÐœÐžÐŒ |i1 -
                                i2| Ñ) 2
0087  7D 06                     jge f3_jl_result; ÐµÑ)Ð»Ðž |i1 - i2| >= 2,
                                Ð¿ÐµÑ(ÐµÐŒÐµÑ)Ñ(ÐžÐŒÑ)Ñ    Ð² f3_jl_result
0089  A3 000C R                 mov res, ax       ; |i1 - i2| < 2 => res
                                = |i1 - i2|
008C  EB 20 90                        jmp end_f         ; Ð·Ð°Ð²ÐµÑ(Ñ(Ð°ÐµÐŒ Ð¿
                                Ñ(ÐŸÐ³Ñ(Ð°ÐŒÐŒÑ)

008F                            f3_jl_result:
008F  C7 06 000C R 0002         mov res, 2h       ; |i1 - i2| >= 2 => res
                                 = 2
0095  EB 17 90                        jmp end_f         ; Ð·Ð°Ð²ÐµÑ(Ñ(Ð°ÐµÐŒ Ð¿
                                Ñ(ÐŸÐ³Ñ(Ð°ÐŒÐŒÑ)

0098                            f3_1:                   ; k >= 0
0098  A1 000A R                 mov ax, i2        ; ÐºÐ»Ð°ÐŽÐµÐŒ Ð² ax Ð¿
                                ÐµÑ(ÐµÐŒÐµÐœÐœÑ)Ñ    i2
009B  F7 D8                     neg ax            ; ax = -ax
009D  3D FFFA                         cmp ax, -6h       ; Ñ)Ñ(Ð°Ð²ÐœÐžÐ²Ð°ÐµÐŒ
                                ax, -6
00A0  7E 06                     jle f3_jge    ; ÐµÑ)Ð»Ðž -i2 <= -6, Ð¿ÐµÑ(
                                ÐµÐŒÐµÑ)Ñ(ÐžÐŒÑ)Ñ    Ð² f3_jge
00A2  A3 000C R                 mov res, ax       ; -i2 > -6 => res = -i2
00A5  EB 07 90                        jmp end_f         ; Ð·Ð°Ð²ÐµÑ(Ñ(Ð°ÐµÐŒ Ð¿
                                Ñ(ÐŸÐ³Ñ(Ð°ÐŒÐŒÑ)

00A8                            f3_jge:
00A8  C7 06 000C R FFFA         mov res, -6h      ; -i2 <= -6 => res = -6

#Microsoft (R) Macro Assembler Version 5.10              10/22/20 17:40:0
                                     Page     1-4


00AE                            end_f:
00AE  B4 4C                     mov ah, 4ch       ; Ð·Ð°Ð²ÐµÑ(Ñ(ÐžÐŒ Ð¿Ñ(
                                ÐŸÐ³Ñ(Ð°ÐŒÐŒÑ)
```

```
00B0  CD 21                    int 21h


00B2                           Main      ENDP
00B2                           CODE      ENDS
                               END Main                          ;ENDS C
                               ODE
#Microsoft (R) Macro Assembler Version 5.10              10/22/20 17:40:0
                                                          Symbols-1


Segments and Groups:

                N a m e         Length   Align   Combine Class

CODE . . . . . .               00B2     PARA   NONE
DATASG . . . . . .             000E     PARA   NONE  'DATA'
STACKSG . . . . . .            0040     PARA   STACK'STACK'

Symbols:

                N a m e         Type     Value    Attr

A . . . . . .                  L WORD         0000     DATASG

B . . . . . .                  L WORD         0002     DATASG

END_F . . . . . .              L NEAR         00AE     CODE

F1 . . . . . .                 L NEAR         0005     CODE
F1_1 . . . . . .               L NEAR         001F     CODE
F2 . . . . . .                 L NEAR         0035     CODE
F2_1 . . . . . .               L NEAR         0053     CODE
F3 . . . . . .                 L NEAR         006B     CODE
F3_1 . . . . . .               L NEAR         0098     CODE
F3_ABS . . . . . .             L NEAR         0082     CODE
F3_JGE . . . . . .             L NEAR         00A8     CODE
```

```
F3_JL_RESULT . . . . . .    L NEAR         008F    CODE
F3_RESULT  . . . . . .      L NEAR         0084    CODE

I  . . . . . .              L WORD         0004    DATASG
I1 . . . . . .              L WORD         0008    DATASG
I2 . . . . . .              L WORD         000A    DATASG

K  . . . . . .              L WORD         0006    DATASG

MAIN . . . . . .            F PROC         0000    CODE  Length = 00B2

RES  . . . . . .            L WORD         000C    DATASG

@CPU . . . . . .            TEXT  0101h
@FILENAME  . . . . . .      TEXT  LAB3
@VERSION . . . . . .        TEXT  510

#Microsoft (R) Macro Assembler Version 5.10                 10/22/20 17:40:0

                                                            Symbols-2



  127 Source  Lines
  127 Total   Lines
   29 Symbols

 47962 + 453153 Bytes symbol space free

    0 Warning Errors
    0 Severe  Errors
```