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 _____ Павлов Р.В.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Освоить методы обработки целых чисел, вычислить значение выражения, зависящего от них.

Задание (вариант 14, 3.4.2).

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.

Ход работы.

1) Реализована процедура считывания строки из консоли и конвертирования её в целое число. Строка считывается с помощью функции прерывания (0ah), затем посимвольно анализируется и заносится в регистр AX с домножением старших разрядов на 10 (может быть указано другое значение системы счисления).

2) Реализована процедура вычисления итогового результата из значений двух первоначально вычисленных функций. Проверяется условие (cmp) положительности/отрицательности последнего параметра, затем в зависимости от условия вычисляется необходимое значение и заносится в AX (jmp, jg, jl и т.п.).

3) В главной процедуре (main) считываются параметры функций, после чего для двух первых функций (всё так же с проверкой условия) вычисляются значения. Эти значения подаются в качестве параметров в результирующую функцию и после вычисления её значения и занесения в AX программа завершается.

Тестирование

Результаты тестирования представлены в таблице 1

Таблица 1

Входные данные	Результаты вычислений
a:1 b:3 i:13 k:6	i1:-70 i2:45 res:115
a:6 b:2 i:5 k:0	i1:-13 i2:-26 res:13
a:4 b:5 i:9 k:-5	i1:-46 i2:33 res:-23
a:110 b:54 i:10 k:-14	i1:-33 i2:-56 res:66

Выводы.

В результате выполнения лабораторной работы были изучены способы обработки целых чисел и взаимодействия с ними.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД

- имя файла : lab3.asm

```
stack segment stack
      dw 4 dup(?)
stack ends

data segment
      i1 dw 0
      i2 dw 0
      buff    db 4,?,4 Dup(?)

data ends

code segment

      assume cs:code ds:data ss:stack

      input proc near

            mov cx, 0
            mov ah,0ah
            sub di,di
            mov dx,offset buff            ; Считывание строки и запись её в
буфер, перевод на новую строку
            int 21h
            mov dl,0ah
            mov ah,02
            int 21h

            mov si,offset buff+2
            cmp byte ptr [si], '-'  ;Запоминаем знак для флага
            jnz pre
            mov di,1
            inc si
```

```
            pre:
            sub ax,ax                       ; Готовим регистры для записи: ax =
0 , bx = 10 - основание СС
            mov bx,10

            transform:
            mov cl,[si]                     ; Проверка на последний символ
            cmp cl,0dh
            jz sign


            cmp cl,'0'                      ; Проверка на соответствие цифре
            jb err
            cmp cl,'9'
            ja err

            sub cl,'0'                      ; Перевод из кода символа в цифру,
домножение на 10, прибавление в конец
            mul bx
            add ax,cx
            inc si
            jmp transform

            err:
            mov dx, offset error    ; Ошибка (если не цифра), выход
            mov ah,09
            int 21h
            int 20h

            sign:
            cmp di,1                        ; Установка знака
            jnz fin
            neg ax

            fin:
            ret

      error db "incorrect number$"
      input endp
```

```
result proc near
        cmp bx, 0
        jl less
        sub cx, dx                  ; Если k >= 0, находим разность
        cmp cx, 0                   ; Если разность < 0, меняем знак,
находя модуль
        jge skip
        neg cx

        skip:
        mov ax, cx                  ; Запись в ax
        jmp exit

        less:
        sub dx, 10                  ; Если k < 0, вычитаем из i2 10
        neg dx                          ; Меняем знак
        cmp cx, dx
        jg greater
        mov ax, dx                  ; Находим наибольшее, записываем
его в ax
        jmp exit

        greater:
        mov ax, cx

        exit:
        ret
result endp

main proc far
        push ds
        sub ax,ax                   ; Подготовка сегментов
        push ax

        mov ax, seg data
        mov ds, ax                  ; Привязка к сегменту данных
```

```
sub ax, ax

call input
mov ds:i1, ax                ;ввод a
call input
mov ds:i2, ax                ;ввод b
call input
mov bx, ax                   ;ввод i

mov cx, ds:i1                ; Заносим в регистры
mov dx, ds:i2

cmp cx, dx
jg greater1

shl bx, 1
shl bx, 1
add bx, ax
add bx, ax                   ; F1 если a <= b
sub bx, 8
neg bx
mov ds:i1, bx
jmp next1

greater1:
shl bx, 1
shl bx, 1                    ; F1 если a > b
sub bx, 7
neg bx
mov ds:i1, bx

next1:
mov bx, ax

cmp cx, dx
jg greater2

shl bx, 1
```

```
            add bx, ax
            add bx, 6                       ; F2 если a <= b
            mov ds:i2, bx
            jmp next2

            greater2:
            shl bx, 1
            shl bx, 1
            add bx, ax                      ; F2 если a > b
            add bx, ax
            sub bx, 4
            neg bx
            mov ds:i2, bx

            next2:

            call input
            mov bx, ax        ;k
            mov cx, ds:i1                   ; Заносим в регистры значения, вы-
численные функциями
            mov dx, ds:i2

            call result                     ; Вызов процедуры нахождения значе-
ния F3 (RES)

            ret
      main endp
code ends
end main
```

ПРИЛОЖЕНИЕ Б. ТЕКСТ ЛИСТИНГОВ

- имя файла: lab3.lst

```
#Microsoft (R) Macro Assembler Version 5.10                 10/29/20 01:32:4
                                                            Page     1-1


 0000                   stack segment stack
 0000  0004[                           dw 4 dup(?)
         ????
                    ]

 0008                   stack ends

 0000                   data segment
 0000  0000                   i1 dw 0
 0002  0000                   i2 dw 0
 0004  04 00                           buff    db 4,?,4 Dup(?)
       0004[
         ??
                    ]


 000A                   data ends

 0000                   code segment

                               assume cs:code ds:data ss:stack
lab3.asm(14): warning A4001: Extra characters on line

 0000                          input proc near

 0000  B9 0000                                 mov cx, 0
 0003  B4 0A                                   mov ah,0ah
 0005  2B FF                                   sub di,di
 0007  BA 0004 R                       mov dx,offset buff
                              ; Считывание строки и
                       запись её в буфер, перевод
                        на новую строку
 000A  CD 21                                   int 21h
 000C  B2 0A                                   mov dl,0ah
 000E  B4 02                                   mov ah,02
 0010  CD 21                                   int 21h

 0012  BE 0006 R                       mov si,offset buff+2
 0015  80 3C 2D                                cmp byte ptr [si], '-'  ;Зап
                       оминаем знак для флага
 0018  75 04                                   jnz pre
 001A  BF 0001                                 mov di,1
 001D  46                              inc si

 001E                                  pre:
 001E  2B C0                                   sub ax,ax
                              ; Готовим регистр
                       ы для записи: ax = 0 , bx = 10 - ос
                       нование СС
 0020  BB 000A                                 mov bx,10

 0023                                  transform:
```

```
0023  8A 0C                             mov cl,[si]
                                ;         Проверка          на          посЙ
                          »едний символ
#Microsoft (R) Macro Assembler Version 5.10                  10/29/20 01:32:4
                                                             Page     1-2


0025  80 F9 0D                          cmp cl,0dh
0028  74 1D                             jz sign


002A  80 F9 30                          cmp cl,'0'
                                ;         Проверка          на          сооэ
                          ⓩветствие цифре
002D  72 0F                             jb err
002F  80 F9 39                          cmp cl,'9'
0032  77 0A                             ja err

0034  80 E9 30                          sub cl,'0'
                                ; Перевод из кода
                          символа           в           цифру,           домноженЙ
                          ¸е на 10, прибавление в коне
                          ц
0037  F7 E3                             mul bx
0039  03 C1                             add ax,cx
003B  46                          inc si
003C  EB E5                             jmp transform

003E                              err:
003E  BA 004F R                   mov dx, offset error    ;              ОшЙ
                          ¸бка (если не цифра), выход
0041  B4 09                             mov ah,09
0043  CD 21                             int 21h
0045  CD 20                             int 20h

0047                              sign:
0047  83 FF 01                         cmp di,1
                                ; Установка знака
004A  75 02                             jnz fin
004C  F7 D8                             neg ax

004E                              fin:
004E  C3                          ret

004F  69 6E 63 6F 72 72           error db "incorrect number$"
      65 63 74 20 6E 75
      6D 62 65 72 24
0060                        input endp



0060                        result proc near
0060  83 FB 00                         cmp bx, 0
0063  7C 0E                            jl less
0065  2B CA                            sub cx, dx
                                ; Если k >= 0, находи
                          м разность
0067  83 F9 00                         cmp cx, 0
                                ; Если разность < 0
                          ,       меняем       знак,       находя       модуЙ
                          »ь
```

```
006A  7D 02                             jge skip
006C  F7 D9                             neg cx

006E                                skip:
006E  8B C1                             mov ax, cx
                                    ; Запись в ax
0070  EB 11 90                          jmp exit

0073                                less:
0073  83 EA 0A                          sub dx, 10
                                    ;     Если     k     <     0,     вычитаЙ
                        µм из i2 10
0076  F7 DA                             neg dx
                                    ; Меняем знак
0078  3B CA                             cmp cx, dx
007A  7F 05                             jg greater
007C  8B C2                             mov ax, dx
                                    ; Находим наиболь
                        шее, записываем его в ax
007E  EB 03 90                          jmp exit

0081                                greater:
0081  8B C1                             mov ax, cx

0083                                exit:
0083  C3                            ret
0084                          result endp

0084                          main proc far
0084  1E                             push ds
0085  2B C0                             sub ax,ax
                                    ; Подготовка сегм
                        ентов
0087  50                            push ax

0088  B8 ---- R                     mov ax, seg data
008B  8E D8                             mov ds, ax
                                    ;         Привязка         к         сегмЙ
                        µнту данных
008D  2B C0                             sub ax, ax

008F  E8 0000 R                     call input
0092  A3 0000 R                     mov ds:i1, ax
                                ;ввод a
0095  E8 0000 R                        call input
0098  A3 0002 R                        mov ds:i2, ax
                                ;ввод b
009B  E8 0000 R                        call input
009E  8B D8                             mov bx, ax
                                    ;ввод i

00A0  8B 0E 0000 R                      mov cx, ds:i1
                                ; Заносим в регистры
00A4  8B 16 0002 R                      mov dx, ds:i2
```

```
00A8  3B CA                             cmp cx, dx
00AA  7F 14                             jg greater1

00AC  D1 E3                             shl bx, 1
00AE  D1 E3                             shl bx, 1
00B0  03 D8                             add bx, ax
00B2  03 D8                             add bx, ax
                                    ; F1 если a <= b
00B4  83 EB 08                          sub bx, 8
00B7  F7 DB                             neg bx
00B9  89 1E 0000 R                      mov ds:i1, bx
00BD  EB 0E 90                          jmp next1

00C0                                greater1:
00C0  D1 E3                             shl bx, 1
00C2  D1 E3                             shl bx, 1
                                    ; F1 если a > b
00C4  83 EB 07                          sub bx, 7
00C7  F7 DB                             neg bx
00C9  89 1E 0000 R                      mov ds:i1, bx

00CD                                next1:
00CD  8B D8                             mov bx, ax

00CF  3B CA                             cmp cx, dx
00D1  7F 0E                             jg greater2

00D3  D1 E3                             shl bx, 1
00D5  03 D8                             add bx, ax
00D7  83 C3 06                          add bx, 6
                                    ; F2 если a <= b
00DA  89 1E 0002 R                      mov ds:i2, bx
00DE  EB 12 90                          jmp next2

00E1                                greater2:
00E1  D1 E3                             shl bx, 1
00E3  D1 E3                             shl bx, 1
00E5  03 D8                             add bx, ax
                                    ; F2 если a > b
00E7  03 D8                             add bx, ax
00E9  83 EB 04                          sub bx, 4
00EC  F7 DB                             neg bx
00EE  89 1E 0002 R                      mov ds:i2, bx

00F2                                next2:

00F2  E8 0000 R                     call input
00F5  8B D8                             mov bx, ax      ;k
00F7  8B 0E 0000 R                      mov cx, ds:i1
                          ; Заносим в регистры з
                        начения, вычисленные функ
                        циями
00FB  8B 16 0002 R                      mov dx, ds:i2
```

```
#Microsoft (R) Macro Assembler Version 5.10                 10/29/20 01:32:4
                                                            Page     1-5


 00FF  E8 0060 R                    call result
                                    ; Вызов процедуры
                          нахождения значения F3 (RES)

 0102  CB                           ret
 0103                          main endp
 0103                    code ends
                         end main
#Microsoft (R) Macro Assembler Version 5.10                 10/29/20 01:32:4
                                                            Symbols-1


Segments and Groups:

                N a m e             Length       Align      Combine Class

CODE . . . . . .                    0103  PARA  NONE
DATA . . . . . .                    000A  PARA  NONE
STACK  . . . . . .                  0008  PARA  STACK

Symbols:

                N a m e             Type   Value       Attr

BUFF . . . . . .                    L BYTE      0004  DATA

ERR  . . . . . .                    L NEAR      003E  CODE
ERROR  . . . . . .                  L BYTE      004F  CODE
EXIT . . . . . .                    L NEAR      0083  CODE

FIN  . . . . . .                    L NEAR      004E  CODE

GREATER  . . . . . .                L NEAR      0081  CODE
GREATER1 . . . . . .                L NEAR      00C0  CODE
GREATER2 . . . . . .                L NEAR      00E1  CODE

I1 . . . . . .                      L WORD      0000  DATA
I2 . . . . . .                      L WORD      0002  DATA
INPUT  . . . . . .                  N PROC      0000  CODE  Length = 0060

LESS . . . . . .                    L NEAR      0073  CODE

MAIN . . . . . .                    F PROC      0084  CODE  Length = 007F

NEXT1  . . . . . .                  L NEAR      00CD  CODE
NEXT2  . . . . . .                  L NEAR      00F2  CODE

PRE  . . . . . .                    L NEAR      001E  CODE

RESULT . . . . . .                  N PROC      0060  CODE  Length = 0024

SIGN . . . . . .                    L NEAR      0047  CODE
SKIP . . . . . .                    L NEAR      006E  CODE

TRANSFORM  . . . . . .              L NEAR      0023  CODE
```

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

#Microsoft (R) Macro Assembler Version 5.10                 10/29/20 01:32:4
                                                            Symbols-2


    171 Source  Lines
    171 Total   Lines
     28 Symbols

  47940 + 455223 Bytes symbol space free

      1 Warning Errors
      0 Severe  Errors
```