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 _____ Подопригора И.П.

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: а) 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.

Вариант 9:

/ - (4*i+3) , при a>b

i1 = f2 =

\ 6*i -10 , при a<=b

/ -(6*i - 4) , при a>b

i2 = f4 =

\ 3*(i+2) , при a<=b

/ |i1| + |i2|, при k<0

res = f7 =

\ max(6, |i1|), при k>=0

Выполнение работы.

Для считывания чисел из командной строки реализована процедура Read, в которой сначала считывается строка функцией 0Ah и прерыванием int 21h, для этой функции заранее подготовлен буфер string содержащий в первом байте максимальное число вводимых символов. После считывания строки по втором байте находится число считанных символов. Первый символ введенной строки проверяется на то, является ли он минусом, и в зависимости от этого в переменную sign заносится знак введенного числа (1 или -1). Далее в цикле последовательно обрабатываются цифры числа, формируя само число, которое будет храниться в регистре cx в конце цикла. Алгоритм формирования числа: изначально в cx хранится 0, при обработке очередной цифры значение в cx умножается на 10 и к нему прибавляется число, состоящее из обрабатываемой цифры. Цикл прекращается, когда индекс(di) обрабатываемой строки дойдет до её конца. В конце, если знак числа отрицательный, то значение в cx командой neg переводится в отрицательное.

Далее происходит расчет функций f2, f4, f7. При организации ветвящихся процессов использовалась функция cmp и условные переходы. Для операций умножения использовался битовый сдвиг влево(команда shl) и сложение (команда add).

Таблица 1. Проверка работы программы с помощью отладчика (все результаты заносились в регистр CX).

Введённые значения a, b, i, k	Полученное значение i1	Полученное значение i2	Полученное значение res	Примечание
a=-3, b=-1, i=4, k=-5	001Eh = 14	0012h = 18	0020h = 32	Верно, a<b => f2=6*i–10=14, f4=3*(i+2)=18, k<0=>f7=|i1|+|i2|=32
a=111, b=83, i = -4, k = 5	000Dh = 13	001Ch = 28	000Dh	Верно, a >b => f2=-(4*i+3)=13, f4=-(6*i-4)=28, k>0 => f7= max(6, |i1|)=13
a=-10, b=-10, i = -10, k = 17	FFBAh=-70	FFE8h=-24	0046h=70	Верно, a =b => f2=6*i–10=-70, f4=3*(i+2)=-24, k>0 => f7= max(6, |i1|)=70
a=-666, b=666, i = -123, k = -120	FD14h=-748	FE95h=-363	0457h=1111	Верно, a<b => f2=6*i–10=-748, f4=3*(i+2)=-363, k<0=>f7=|i1|+|i2|=1111

Тексты исходных файлов программ см. в приложении А.

Тексты файлов диагностических сообщений см. в приложении Б.

Выводы.

В ходе выполнения данной лабораторной работы была изучена работа с целыми числами и условными переходами на языке Ассемблер.

ПРИЛОЖЕНИЕ А

ТЕКСТЫ ИСХОДНЫХ ФАЙЛОВ ПРОГРАММ

Название файла: **lr3.asm**

```
; Стек  программы
AStack SEGMENT  STACK
   DW 12 DUP(?)
AStack  ENDS
;Данные программы
DATA     SEGMENT
;Директивы описания данных
string DB    15, 15 DUP('$')
sign   DB    1
a      DW    0
b      DW    0
i      DW    0
k      DW    0
i1     DW    0
i2     DW    0

DATA     ENDS

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

Read     PROC NEAR
    mov  dx, OFFSET string
         mov  ah, 0Ah
         int  21h
         mov  di, 2
         cmp  string[2], 45   ;проверка, введён ли минус
         jne  positive
           mov  al, -1
                mov  di, 3        ;индекс первой цифры в string, если введён минус
           jmp  EndOfIf
         positive:
           mov  al, 1
         EndOfIf:
         mov  sign, al       ; в sign храниться знак числа
         mov  cx, 0
         ; цикл формирования числа и сохранения его в cx
         L:
                mov  ax, cx        ;>
                SHL  cx, 1         ;>>
                SHL  cx, 1         ;>>> умножение на 10
                SHL  cx, 1         ;>>>
                add  cx, ax        ;>>
                add  cx, ax        ;>
                mov  al, string[di] ;
```

```
                sub  al, 30h        ;  занесение в ax числа, состоящего из очередной цифры введённого числа
                mov  ah, 0         ;
                add  cx, ax

                mov  al, string[1]
                mov  ah, 0
                add  ax, 1  ; в ax сохраняется индекс последнего символа в строке
                cmp  di, ax
                je   LExit   ; выход из цикла, если текущая обработанная цифра последняя
                  inc  di
                  jmp  L
            LExit:
            cmp sign, -1
        jne final
              neg cx        ; изменение знака числа
            final:
        ret
    Read ENDP


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

            ;считывание a, b, i, k
            Call  Read
            mov a, cx
            Call  Read
            mov b, cx
            Call  Read
            mov i, cx
            Call  Read
            mov k, cx

            ;вычисление f2
            mov cx, i
            mov ax, cx
            shl cx, 1
            shl cx, 1
            mov bx, b    ;
            cmp a, bx    ; сравнение a и b
            jle f2second
                  add cx, 3
                  neg cx
                  jmp f2final
            f2second:
```

```
        add cx, ax
        add cx, ax
        add cx, -10
f2final:
mov i1, cx

;вычисление f4
mov cx, i
cmp a, bx
jle f4second
  shl cx, 1
        mov ax, cx
        shl cx, 1
        add cx, ax
        add cx, -4
        neg cx
        jmp f4final
f4second:
  add cx, 2
        mov ax, cx
        shl cx, 1
        add cx, ax
f4final:
mov i2, cx

mov cx, i1
cmp cx, 0
jge skip1     ;модуль i1
  neg cx
        mov i1, cx
skip1:

mov cx, i2
cmp cx, 0
jge skip2     ;модуль i2
  neg cx
        mov i2, cx
skip2:

;рассчет f7
mov bx, k
cmp bx, 0
jl f7Second
  mov bx, i1
        cmp bx, 6
        jl max1
          mov cx, bx       ; |i1| >= 6
          jmp MainFinal
        max1:
          mov cx, 6        ; |i1| < 6
          jmp MainFinal
f7Second:
```

```
            mov cx, i1
                add cx, i2
          MainFinal:          ; в cx лежит значение функции f7
    ret
Main     ENDP
CODE     ENDS
END Main
```

ПРИЛОЖЕНИЕ Б
ТЕКСТЫ ФАЙЛОВ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ

Название файла: **lr3.lst**

```
Microsoft (R) Macro Assembler Version 5.10              10/5/21 16:25:13
                                                        Page     1-1


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

0018                    AStack  ENDS
                        ;Данные программы
0000                    DATA    SEGMENT
                        ;Директивы описания данны
                        x
0000  0F                string DB    15, 15 DUP('$')
    000F[
        24
                    ]

0010  01                sign   DB    1
0011  0000              a      DW    0
0013  0000              b      DW    0
0015  0000              i      DW    0
0017  0000              k      DW    0
0019  0000              i1     DW    0
001B  0000              i2     DW    0
```

```
001D                    DATA    ENDS

                        ; Код программы
0000                    CODE    SEGMENT
                            ASSUME CS:CODE, DS:DATA, SS:AStack

0000                    Read    PROC NEAR
0000  BA 0000 R             mov  dx, OFFSET string
0003  B4 0A                      mov  ah, 0Ah
0005  CD 21                      int  21h
0007  BF 0002                    mov  di, 2
000A  80 3E 0002 R 2D             cmp  string[2], 45   ;проверкЙ
                        °, введён ли минус
000F  75 08                      jne  positive
0011  B0 FF                       mov  al, -1
0013  BF 0003                         mov  di, 3        ;индекэ
                        □ первой цифры в string, если Й
                        ²ведён минус
0016  EB 03 90                    jmp  EndOfIf
0019                        positive:
0019  B0 01                       mov  al, 1
001B                        EndOfIf:
001B  A2 0010 R                 mov  sign, al       ; в sign хра
                        ниться знак числа
001E  B9 0000                     mov  cx, 0
                        ; цикл формирования э
                        □исла и сохранения его в cx
0021                        L:
0021  8B C1                           mov  ax, cx       ;>
```

```
0023  D1 E1                          SHL  cx, 1        ;>>
```


```
0025  D1 E1                          SHL  cx, 1        ;>>> умн
                    ожение на 10
0027  D1 E1                          SHL  cx, 1        ;>>>
0029  03 C8                          add  cx, ax       ;>>
002B  03 C8                          add  cx, ax       ;>
002D  8A 85 0000 R                   mov  al, string[di] ;
0031  2C 30                          sub  al, 30h      ;  зане
                    сение в ax числа, состоящег
                    о из очередной цифры введэ
                    □нного числа
0033  B4 00                          mov  ah, 0        ;
0035  03 C8                        add  cx, ax

0037  A0 0001 R                  mov  al, string[1]
003A  B4 00                          mov  ah, 0
003C  05 0001                        add  ax, 1  ; в ax сохраЙ
                    ½яется индекс последнего э
                    □имвола в строке
003F  3B F8                          cmp  di, ax
0041  74 03                       je   LExit   ; выход из
                    цикла, если текущая обрабЙ
                    ¾танная цифра последняя
0043  47                           inc  di
0044  EB DB                          jmp  L
0046                            LExit:
```

```
0046  80 3E 0010 R FF                cmp sign, -1
004B  75 02                       jne final
004D  F7 D9                         neg cx        ; изменение
                          знака числа
004F                              final:
004F  C3                        ret
0050                        Read ENDP


                          ; Головная процедура
0050                        Main      PROC  FAR
0050  1E                        push  DS
0051  2B C0                        sub   AX,AX
0053  50                        push  AX
0054  B8 ---- R                 mov   AX,DATA
0057  8E D8                        mov   DS,AX
0059  B9 0000                          mov   CX, 0

                                ;считывание a, b, i, k
005C  E8 0000 R                 Call  Read
005F  89 0E 0011 R                      mov a, cx
0063  E8 0000 R                 Call  Read
0066  89 0E 0013 R                      mov b, cx
006A  E8 0000 R                 Call  Read
006D  89 0E 0015 R                      mov i, cx
0071  E8 0000 R                 Call  Read
0074  89 0E 0017 R                      mov k, cx

                                ;вычисление f2
```

```
0078  8B 0E 0015 R                mov cx, i
007C  8B C1                      mov ax, cx
007E  D1 E1                      shl cx, 1
0080  D1 E1                      shl cx, 1
0082  8B 1E 0013 R               mov bx, b    ;
0086  39 1E 0011 R               cmp a, bx    ; сравнение a Ù
                          ¸ b
008A  7E 08                      jle f2second
008C  83 C1 03                        add cx, 3
008F  F7 D9                           neg cx
0091  EB 08 90                        jmp f2final
0094                        f2second:
0094  03 C8                           add cx, ax
0096  03 C8                           add cx, ax
0098  83 C1 F6                        add cx, -10
009B                        f2final:
009B  89 0E 0019 R                mov i1, cx

                            ;вычисление f4
009F  8B 0E 0015 R                mov cx, i
00A3  39 1E 0011 R                cmp a, bx
00A7  7E 10                       jle f4second
00A9  D1 E1                        shl cx, 1
00AB  8B C1                           mov ax, cx
00AD  D1 E1                           shl cx, 1
00AF  03 C8                           add cx, ax
00B1  83 C1 FC                        add cx, -4
```

```
00B4  F7 D9                     neg cx
00B6  EB 0A 90                  jmp f4final
00B9                    f4second:
00B9  83 C1 02                  add cx, 2
00BC  8B C1                     mov ax, cx
00BE  D1 E1                     shl cx, 1
00C0  03 C8                     add cx, ax
00C2                    f4final:
00C2  89 0E 001B R              mov i2, cx


00C6  8B 0E 0019 R              mov cx, i1
00CA  83 F9 00                  cmp cx, 0
00CD  7D 06                     jge skip1     ;модуль i1
00CF  F7 D9                     neg cx
00D1  89 0E 0019 R              mov i1, cx
00D5                    skip1:

00D5  8B 0E 001B R              mov cx, i2
00D9  83 F9 00                  cmp cx, 0
00DC  7D 06                     jge skip2     ;модуль i2
00DE  F7 D9                     neg cx
00E0  89 0E 001B R              mov i2, cx
00E4                    skip2:

                        ;рассчет f7
00E4  8B 1E 0017 R              mov bx, k
Microsoft (R) Macro Assembler Version 5.10          10/5/21 16:25:13
                                Page     1-4
```

```
00E8  83 FB 00                    cmp bx, 0
00EB  7C 14                       jl f7Second
00ED  8B 1E 0019 R                  mov bx, i1
00F1  83 FB 06                        cmp bx, 6
00F4  7C 05                           jl max1
00F6  8B CB                             mov cx, bx      ; i1 >= 6
00F8  EB 0F 90                          jmp MainFinal
00FB                              max1:
00FB  B9 0006                           mov cx, 6       ; i1 < 6
00FE  EB 09 90                          jmp MainFinal
0101                            f7Second:
0101  8B 0E 0019 R                    mov cx, i1
0105  03 0E 001B R                      add cx, i2
0109                            MainFinal:        ; в cx лежи
                        т значение функции f7
0109  CB                          ret
010A                            Main      ENDP
010A                            CODE      ENDS
                                END Main
Microsoft (R) Macro Assembler Version 5.10              10/5/21 16:25:13
                                                        Symbols-1


Segments and Groups:

                N a m e         Length   Align   Combine Class

ASTACK . . . . . . . . . . . .  0018  PARA      STACK
CODE . . . . . . . . . . . . .  010A  PARA      NONE
```

```
DATA . . . . . 001D PARA        NONE

Symbols:

                N a m e         Type    Value Attr

A . . . . . .       L WORD     0011  DATA

B . . . . . .       L WORD     0013  DATA

ENDOFIF . . . . . .         L NEAR     001B  CODE

F2FINAL . . . . . .         L NEAR     009B  CODE
F2SECOND . . . . . .        L NEAR     0094  CODE
F4FINAL . . . . . .         L NEAR     00C2  CODE
F4SECOND . . . . . .        L NEAR     00B9  CODE
F7SECOND . . . . . .        L NEAR     0101  CODE
FINAL . . . . . .           L NEAR     004F  CODE

I . . . . . .       L WORD     0015  DATA
I1 . . . . . .      L WORD     0019  DATA
I2 . . . . . .      L WORD     001B  DATA

K . . . . . .       L WORD     0017  DATA

L . . . . . .       L NEAR     0021  CODE
LEXIT . . . . . .           L NEAR     0046  CODE

MAIN . . . . . .            F PROC     0050  CODE        Length = 00BA
MAINFINAL . . . . . .       L NEAR     0109  CODE
```

MAX1 . . . . . L NEAR 00FB CODE

POSITIVE . . . . . L NEAR 0019 CODE

READ . . . . . N PROC 0000 CODE Length = 0050

SIGN . . . . . L BYTE 0010 DATA
SKIP1 . . . . . L NEAR 00D5 CODE
SKIP2 . . . . . L NEAR 00E4 CODE
STRING . . . . . L BYTE 0000 DATA

@CPU . . . . . TEXT 0101h
@FILENAME . . . . . TEXT lr3
@VERSION . . . . . TEXT 510

157 Source Lines
157 Total Lines
32 Symbols

47916 + 455247 Bytes symbol space free

0 Warning Errors

0 Severe Errors