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. 0382 _____ Тихонов С.В.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Разобрать и научится использовать механизм ветвления в программах на языке Ассемблер. Разработать программу на основе полученных знаний.

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.

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.

Выполнение работы.

Функции выбранные в соответствии с вариантом:

Рисунок 1: функция f1

$$f3 = \begin{cases} 7 - 4*i, & \text{при } a>b \\ 8 - 6*i, & \text{при } a<=b \end{cases}$$

Рисунок 2: функция f2

$$f8 = \begin{cases} -(6*i+8), & \text{при } a>b \\ 9 - 3*(i-1), & \text{при } a<=b \end{cases}$$

Рисунок 3: функция f3

$$f6 = \begin{cases} |i1 - i2|, & \text{при } k<0 \\ \max(7, |i2|), & \text{при } k>=0 \end{cases}$$

В процессе выполнения задания была разработана программ, которая состоит из несколько частей:

1. Описание сегментов программы. В их число входят сегмент стека , сегмент данных в котором была выделена память для переменных ***a, b, i, k, i1, i2, res***.

2. Потом идет сегмент кода, в котором прописана сама программа.

3. Прописываются необходимые вещи для нормальной работы любой программы , такие как сохранение адреса начала PSP в стеке, загрузка сегментного регистра данных и т.д.

4. Затем анализируются значения *a* и *b*. Если *a>b* то выполняется блок программы ответственный за функции *f1* и *f2* для *a>b*. Иначе выполняется блок для функций *f1* и *f2* при *a<=b*.

5. В блоке с меткой ***f3*** анализируется значение ***k*** и выполняется функция ***f3*** в соответствии со значением ***k***.

6. Выполняет выход из программы

Тестирование.

Результаты тестирования представлены в табл. 1. Тестирование проводилось в отладчике **AFDPRO**.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии

1.	a = 5; b = 2; i = 2; k = 0	i1 = FFFF (-1);
		i2 = FFEC (-20);
		res = FFF5(14)
			Программа работает корректно

2.	a = 2; b = 5; i = 2; k = 0	i1 = FFFC (-4);
		i2 = 6;
		res =7
			Программа работает корректно

3.	a = 5; b = 2; i = 2;
	k = -1
		i1 = FFFE (-1);
		i2 = FFEC (-20);
		res = 13(19)
			Программа работает корректно

Выводы.

Был изучен механизм ветвления в программах на языке Ассемблер.

Разработана программа, выполняющая поставленную задачу , а именно 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),

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb3.asm

```
EOL EQU '$'
ind EQU 2
n1 EQU 500
n2 EQU -50
AStack SEGMENT STACK
      DW 12 DUP(?)
AStack ENDS
DATA SEGMENT
      a DW 2
      b DW 1
      i DW 2
      k DW 1
      i1 DW 0
      i2 DW 0
      res DW 0
DATA ENDS
CODE SEGMENT
      ASSUME CS:CODE, DS:DATA, SS:AStack
Main PROC FAR
      push DS
      sub AX,AX
      push AX
      mov AX,DATA
      mov DS,AX
      mov ax,i ;ax=i
      shl ax,1 ;ax=2i
      mov bx,a
      cmp bx,b
      jg f38_1 ;jmp if a>b
f38_2:
      add ax,i ;ax=3i
      mov bx,12
      sub bx,ax
      mov i2,bx ;i2=12-3i=9-3*(i-1)
      shl ax,1 ;ax=6i
      neg ax;
      add ax,8
      mov i1,ax ;i1=8-6i
      jmp f6
f38_1:
      shl ax,1 ;ax=4i
      mov bx,ax ;bx=4i
      neg ax ;ax=-4i
      add ax,7 ;ax=7-4i
      mov i1,ax ;i1=7-4i
      mov ax,i ;ax=i
      shl ax,1 ;ax=2i
      add ax,bx ;ax=6i
      add ax,8
      neg ax
```

```
        mov i2,ax ;i2=-(6i+8)
    f6:
        cmp k,0
        jnl case_2 ;k>=0
        mov ax,i1 ;ax=i1
        sub ax,i2 ;ax=i1-i2
        cmp ax,0
        jl neg_sub
        jmp set_res
    case_2:
        mov ax,i2
        cmp ax,0
        jnl case_2_2 ;i2>=0
        neg ax
    case_2_2:
        cmp ax,7 ;
        jnl set_res ;if ax>=7
        mov ax,7
        jmp set_res
    neg_sub:
        neg ax
    set_res:
        mov res,ax
        ret

    Main ENDP
    CODE ENDS
        END Main
```

Название файла: lb3.lst

```
#Microsoft (R) Macro Assembler Version 5.10                    11/23/21
20:59:1
                                                                 Page
1-1
 = 0024                     EOL EQU '$'
 = 0002                     ind EQU 2
 = 01F4                     n1 EQU 500
 =-0032                     n2 EQU -50
 0000                  AStack SEGMENT STACK
 0000  000C[                       DW 12 DUP(?)
        ????
                   ]

 0018                  AStack ENDS
 0000                  DATA SEGMENT
 0000  0002                       a DW 2
```

```
0002  0001                    b DW 1
0004  0002                    i DW 2
0006  0001                    k DW 1
0008  0000                    i1 DW 0
000A  0000                    i2 DW 0
000C  0000                    res DW 0
000E                DATA ENDS
0000                CODE SEGMENT
                         ASSUME CS:CODE, DS:DATA, SS:AStack
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  A1 0004 R           mov ax,i ;ax=i
000C  D1 E0                    shl ax,1 ;ax=2i
000E  8B 1E 0000 R             mov bx,a
0012  3B 1E 0002 R             cmp bx,b
0016  7F 1A                    jg f38_1 ;jmp if a>b
0018                f38_2:
0018  03 06 0004 R             add ax,i ;ax=3i
001C  BB 000C                  mov bx,12
001F  2B D8                    sub bx,ax
0021  89 1E 000A R             mov i2,bx ;i2=12-3i=9-3*(i-1)
0025  D1 E0                    shl ax,1 ;ax=6i
0027  F7 D8                    neg ax;
0029  05 0008                  add ax,8
002C  A3 0008 R           mov i1,ax ;i1=8-6i
002F  EB 1C 90                 jmp f6
0032                f38_1:
0032  D1 E0                    shl ax,1 ;ax=4i
0034  8B D8                    mov bx,ax ;bx=4i
0036  F7 D8                    neg ax ;ax=-4i
0038  05 0007                  add ax,7 ;ax=7-4i
003B  A3 0008 R           mov i1,ax ;i1=7-4i
003E  A1 0004 R           mov ax,i ;ax=i
```

```
 0041  D1 E0                    shl ax,1 ;ax=2i
 0043  03 C3                    add ax,bx ;ax=6i
 0045  05 0008                  add ax,8
 0048  F7 D8                    neg ax
 004A  A3 000A R           mov i2,ax ;i2=-(6i+8)
#Microsoft (R) Macro Assembler Version 5.10                    11/23/21
20:59:1
                                                                     Page
1-2


 004D                f6:
 004D  83 3E 0006 R 00               cmp k,0
 0052  7D 0F                    jnl case_2 ;k>=0
 0054  A1 0008 R           mov ax,i1 ;ax=i1
 0057  2B 06 000A R             sub ax,i2 ;ax=i1-i2
 005B  3D 0000                  cmp ax,0
 005E  7C 18                    jl neg_sub
 0060  EB 18 90                 jmp set_res
 0063                case_2:
 0063  A1 000A R           mov ax,i2
 0066  3D 0000                  cmp ax,0
 0069  7D 02                    jnl case_2_2 ;i2>=0
 006B  F7 D8                    neg ax
 006D                case_2_2:
 006D  3D 0007                  cmp ax,7 ;
 0070  7D 08                    jnl set_res ;if ax>=7
 0072  B8 0007                  mov ax,7
 0075  EB 03 90                 jmp set_res
 0078                neg_sub:
 0078  F7 D8                    neg ax
 007A                set_res:
 007A  A3 000C R           mov res,ax
 007D  CB                  ret

 007E                Main ENDP
 007E                CODE ENDS
```

```
                            END Main
#Microsoft (R) Macro Assembler Version 5.10                 11/23/21
20:59:1
                                                            Symbols-1


Segments and Groups:

                N a m e         Length     Align     Combine Class

ASTACK . . . . . .              0018  PARA  STACK
CODE . . . . . .                007E  PARA  NONE
DATA . . . . . .                000E  PARA  NONE

Symbols:

                N a m e         Type   Value      Attr

A  . . . . . .                  L WORD     0000  DATA

B  . . . . . .                  L WORD     0002  DATA

CASE_2 . . . . . .              L NEAR     0063  CODE
CASE_2_2 . . . . . .            L NEAR     006D  CODE

EOL  . . . . . .                NUMBER     0024

F38_1  . . . . . .              L NEAR     0032  CODE
F38_2  . . . . . .              L NEAR     0018  CODE
F6 . . . . . .                  L NEAR     004D  CODE

I  . . . . . .                  L WORD     0004  DATA
I1 . . . . . .                  L WORD     0008  DATA
I2 . . . . . .                  L WORD     000A  DATA
IND  . . . . . .                NUMBER     0002

K  . . . . . .                  L WORD     0006  DATA
```

```
MAIN . . . . . .      F PROC     0000  CODE  Length     =
007E

N1 . . . . . .        NUMBER     01F4
N2 . . . . . .        NUMBER     -0032
NEG_SUB  . . . . . .  L NEAR     0078  CODE

RES  . . . . . .      L WORD     000C  DATA

SET_RES  . . . . . .  L NEAR     007A  CODE

@CPU . . . . . .      TEXT  0101h
@FILENAME  . . . . .  TEXT  LB3
@VERSION . . . . . .  TEXT  510

#Microsoft (R) Macro Assembler Version 5.10                  11/23/21
20:59:1
                                                          Symbols-2



     78 Source  Lines
     78 Total   Lines
     27 Symbols

  48034 + 461273 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```