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»

Студентка гр. 1381 _____ Васильева О. М.

Преподаватель _____ Ефремов М. А.

Санкт-Петербург

2022

Задание (Шифр 1.4.3.)

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.

Выполнение работы.

1. Функции в соответствие с указанным шифром:

$$f1 = \begin{cases} 15-2*i, & \text{при } a>b \\ 3*i+4, & \text{при } a<=b \end{cases}$$

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

$$f3 = \begin{cases} |i1 + i2|, & \text{при } k=0 \\ \min(i1,i2), & \text{при } k/=0 \end{cases}$$

2. В начале программы описана модель памяти с помощью директивы .model. Модель памяти - .small.

3. Описание директив сегментации:

.stack - начало указания сегмента стека;

.code — начало указания сегмента кода;

.data — начало указания сегмента данных.

4. Также в начале мы инициализируем переменные i, k, a, b и выбираем размер стека.

5. С помощью команды cmp мы сравниваем значения a и b, если a>b, то переходим к метке f12_else, иначе — f12_then.

6. Затем вычисляются значения i1 и i2, которые записываются в регистры ax и cx соответственно.

7. Финальная функция по заданию должна сравнивать k с 0. При k=0 вычисляется модуль суммы i1 и i2, иначе эти два значения сравниваются. Из минимальное записывается в ax.

8. Результат сохраняется в переменной res.

Минимизация длины кода.

- a<=b

  i1 = 3i+4; i2 = 3*(i+2) = 3i+6

  Пусть g= 3i+4, тогда

  i2 = g+2; g=i1.

- a>b

  i1 = 15-2i; i2 = -(6i-4)

  Пусть g= 15-2i, тогда

  i2 = 3*g-11; g=i1.

Вывод.

В ходе лабораторной работы мы познакомились с организацией ветвления на языке Ассемблер и написали программу в соответствие с индивидуальным заданием.

ПРИЛОЖЕНИЕ А. Код программ.

Имя файла: lr3.asm

```
dosseg
.model small
.stack 100h
.data
i dw 1
a dw -5
b dw 8
k dw 4
res dw ?
.code
mov ax, @data
mov ds, ax
mov ax, a
cmp ax, b  ;сравниваем a и b.
jg f12_else  ;выполняет короткий переход, если a больше b

f12_that:    ;if(a<=b)
mov cx, i ;i
add cx, i ;2i
add cx, i ;3i
add cx, 4 ;3i+4

mov ax, cx ;3i+4
add ax, 2  ;3*(i+2) = 3i+6
jmp final

f12_else:    ;if(a>b)
mov cx, 15  ;15
sub cx, i  ;15-i
sub cx, i  ;15-2i

mov ax, cx ;15-2i
```

```
sub ax, i  ;15-3i
sal ax, 1  ;30-6i
sub ax, 26 ;-(6i-4) = -6i+4

final:
abs_that:
cmp k,0
jnz abs_else ;перейти если не равно
add ax, cx
neg ax
jmp results

abs_else:
cmp cx, ax ;сравниваем i1 и i2
jl min
jmp results

min: mov ax, cx

results:
mov res, ax
mov ah, 4ch
int 21h
end
```

ПРИЛОЖЕНИЕ В. *Листинг успешной трансляции программы.*

Текст сообщения (lr3.lst).

```
 #Microsoft (R) Macro Assembler Version 5.10
11/13/22 19:56:0

                                                                 Page
1-1


                                                        dosseg

                                                           .mod
el small

                                                           .sta
ck 100h

                                                           .dat
a
  0000  0001                                            i dw 1

  0002  FFFB                                            a dw -5

  0004  0008                                            b dw 8

  0006  0004                                            k dw 4

  0008  0000                                            res
dw ?


                                                           .cod
e
  0000  B8 ---- R                                          mov
ax, @data
  0003  8E D8                                           mov ds,
ax
  0005  A1 0002 R                                          mov
ax, a
  0008  3B 06 0004 R                                       cmp
ax, b  ;сравниваем a и b.
  000C  7F 17                                           jg
f12_else  ;выполняет коротки
```

```
                                                                  й
переход, если a больше b


  000E

f12_that:    ;if(a<=b)
  000E  8B 0E 0000 R                                                 mov
cx, i ;i
  0012  03 0E 0000 R                                                 add
cx, i ;2i
  0016  03 0E 0000 R                                                 add
cx, i ;3i
  001A  83 C1 04
                                                                     add cx,
4 ;3i+4



  001D  8B C1
                                                                     mov ax,
cx ;3i+4
  001F  05 0002
                                                                     add ax,
2  ;3*(i+2) = 3i+6
  0022  EB 17 90
                                                                     jmp
final



  0025

f12_else:    ;if(a>b)
  0025  B9 000F
                                                                     mov cx,
15  ;15
  0028  2B 0E 0000 R                                                 sub
cx, i  ;15-i
  002C  2B 0E 0000 R                                                 sub
cx, i  ;15-2i



  0030  8B C1
                                                                     mov ax,
cx ;15-2i
```

```
  0032  2B 06 0000 R                                   sub
ax, i  ;15-3i
  0036  D1 E0
                                                      sal ax,
1  ;30-6i
  0038  2D 001A
                                                      sub ax,
26 ;-(6i-4) = -6i+4



  003B
                                                      final:
  003B


abs_that:
  003B  83 3E 0006 R 00                               cmp
k,0
  0040  75 07
                                                      jnz
abs_else ;перейти если не ра

                                                         вно
  0042  03 C1
                                                      add ax,
cx
  0044  F7 D8
                                                      neg ax
  0046  EB 0A 90
                                                      jmp
results



  0049


abs_else:
  0049  3B C8
                                                      cmp cx,
ax ;сравниваем i1 и i2
  004B  7C 03
                                                      jl min
  004D  EB 03 90
                                                      jmp
results
```

```
  0050  8B C1                                                    min:
mov ax, cx



  0052


results:

 #Microsoft (R) Macro Assembler Version 5.10
11/13/22 19:56:0
                                                                 Page
1-2



  0052  A3 0008 R                                                  mov
res, ax
  0055  B4 4C
                                                                   mov ah,
4ch
  0057  CD 21
                                                                   int 21h


                                                                   end
 #Microsoft (R) Macro Assembler Version 5.10
11/13/22 19:56:0


Symbols-1

 Segments and Groups:

                N a m e                                           Length
 Align
                                                                  Combine
Class

 DGROUP . . . . . .                                               GROUP
   _DATA  . . . . . .                                             000A
WORD
                                                                  PUBLIC
                                                                  'DATA'
   STACK  . . . . . .                                             0100
PARA
                                                                  STACK
                                                                  'STACK'
 _TEXT  . . . . . .                                               0059
WORD
                                                                  PUBLIC
```

'CODE'

Symbols:

N a m e	Type Value Attr
A	L WORD 0002 _DATA
ABS_ELSE	L NEAR 0049 _TEXT
ABS_THAT	L NEAR 003B _TEXT
B	L WORD 0004 _DATA
F12_ELSE	L NEAR 0025 _TEXT
F12_THAT	L NEAR 000E _TEXT
FINAL	L NEAR 003B _TEXT
I	L WORD 0000 _DATA
K	L WORD 0006 _DATA
MIN	L NEAR 0050 _TEXT
RES	L WORD 0008 _DATA
RESULTS	L NEAR 0052 _TEXT
@CODE	TEXT _TEXT
@CODESIZE	TEXT 0
@CPU	TEXT 0101h

```
 @DATASIZE  . . . . . .                                  TEXT  0

 @FILENAME  . . . . . .                                  TEXT
lr3
 @VERSION . . . . . .                                    TEXT
510

      56 Source  Lines

      56 Total   Lines

      29 Symbols

   48004 + 459256 Bytes symbol space free

       0 Warning Errors

       0 Severe  Errors
```