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. 0383 _____ Куртова К. А.

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:

a) значения функций i1 = f1(a,b,i) и i2 = f2(a,b,i);

b) значения результирующей функции res = f3(i1,i2,k),

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.

Вариант 12:

$$f2 = \begin{cases} -(4i+3), a > b \\ 6i - 10, a \leq b \end{cases}$$

$$f7 = \begin{cases} -(4i-5), a > b \\ 10 - 3i, a \leq b \end{cases}$$

$$f4 = \begin{cases} \min(|i_1 - i_2|, 2), k < 0 \\ \max(-6, -i_2), k \geq 0 \end{cases}$$

Ход работы.

Для программы прописано три сегмента — сегмент стека, сегмент данных, в котором хранится информация о переменных a, b, I, k, i_1, i_2, res, для каждого из которых выделено слово (word) в памяти. Описан сегмент кода, в котором находится головная процедура Main, в которой и производятся основные вычисления.

В ходе выполнения работы были использованы условные (jmp) и безусловные (jle, lge, jl) переходы. Вычисление функций было выполнено без использования отдельных процедур, только с использованием переходов, которые переходят к указанным лейблам. Код программы представлен в приложении А, листинг программы представлен в приложении Б.

Тестирование программы.

В таблице 1 представлен результат тестирования программы.

Таблица 1 — Результат тестирования программы

Входные данные	Результирующие данные	Проверка
a = 5 b = 3 i = 2 k = -1	i1 = FFF5 = -11 i2 = FFFD = -3 res = 0002 = 2	Верно. i1 = -(8+3) = -11 i2 = -(8-5) = -3 |-11 + 3| > 2 => res = 2
a = 3 b = 5 i = 4 k = -1	i1 = 000E = 14 i2 = i2 = FFFE = -2 res = 0002 = 2	Верно. i1 = 24 – 10 = 14 i2 = 10 – 12 = -2 |14 + 2| > 2 => res = 2
a = 3 b = 5 i = 4 k = 1	i1 = 000E = 14 i2 = i2 = FFFE = -2 res = 0002 = 2	Верно. i1 = 24 – 10 = 14 i2 = 10 – 12 = -2 -6 < 2 => res = 2
a = 3 b = -5 i = -2 k = 1	i1 = 0005 = 5 i2 = 000D = 13 res = FFFA = -6	Верно. i1 = -(-8 + 3) = 5 i2 = -(-8 -5) = 13 -6 > -13 => res = -6

Выводы.

В ходе лабораторной работы были изучены представление и обработка целых чисел. Также были рассмотрены условные и безусловные переходы, и с их помощью была разработана программа, которая по заданным значениям переменных вычисляет значение нескольких функций.

ПРИЛОЖЕНИЕ А

Текст разработанной программы lab3.txt

```
EOL EQU '$

;----------Стек программы----------
AStack SEGMENT STACK
DW 32 DUP(?)
AStack ENDS

;----------Данные программы--------
DATA SEGMENT
buffer DB 128 DUP(?)
a DW 3
b DW -5
i DW -2
k DW 1
i1 DW ?
i2 DW ?
res DW ?
DATA ENDS

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

;--------Головная процедура--------
Main PROC FAR
  push ds
  sub ax, ax
  push ax
  mov ax, DATA
  mov ds, ax

f1:                    ; Вычисляем f1
```

```
  mov ax, i
  shl ax, 1       ; Умножаем i на 4
  shl ax, 1
  mov bx, a
  cmp bx, b     ; Сравниваем a и b
  jle f1_case2 ; Если a <= b, перейти к f2_case2

f1_case1:               ; a > b
  add ax, 3     ; 4i + 3
  neg ax
  mov cx, ax   ; cx: -4i - 3
  jmp f1_end

f1_case2:               ; a <= b
  mov bx, ax   ; bx = ax = 4i
  shr bx, 1      ; bx = 2i
  add ax, bx    ; ax = 4i + 2i = 6i
  mov cx, ax   ; cx = 6i
  sub cx, 10    ; cx = 6i - 10

f1_end:
  mov i1, cx   ; i1 = f1(a, b, i)



f2:                     ; Вычисляем f2
  mov ax, i
  mov bx, a
  cmp bx, b
  jle f2_case2 ; Если a <= b, перейти к f2_case2

f2_case1:
  shl ax, 1       ; Умножаем i на 4
  shl ax, 1
  sub ax, 5      ; ax = 4i - 5
```

```
  neg ax                ; ax = -4i + 5
  mov cx, ax
  jmp f2_end

f2_case2:
  mov bx, ax  ; bx = i
  shl ax, 1       ; ax = 2i
  add ax, bx    ; ax = 3i
  mov bx, ax  ; bx = 3i
  mov ax, 10  ; ax = 10
  sub ax, bx    ; ax = 10 - 3i
  mov cx, ax

f2_end:
  mov i2, cx

f3:                     ; Вычисляем f3
  mov bx, k
  cmp bx, 0
  jge f3_case2          ; Если k >= 0, перейти к f3_case2

f3_case1:
  mov ax, i1
  mov bx, i2
  sub ax, bx    ; i1-i2
  cmp ax, 0
  jl abs          ; Если i1 - i2 < 0, найдём модуль выражения
  jmp min               ; Больше 0, перейти к поиску минимального числа в паре
abs:
  neg ax

min:
  mov bx, 2
  cmp ax, 2
```

```
  jle absmin    ; Если |i1 - i2| <= 2, перейти к absmin
twomin:
  mov cx, 2
  jmp f3_end
absmin:
  mov cx, ax
  jmp f3_end

f3_case2:
  mov ax, i2
  neg ax
  cmp ax, -6
  jge i2max     ; Если -i2 >= -6, перейти к i2max
neg6max:
  mov cx, 6
  neg cx
  jmp f3_end
i2max:
  mov cx, ax
f3_end:
  mov res, cx
  ret

Main ENDP
CODE ENDS
END Main
```

ПРИЛОЖЕНИЕ Б

Файл листинга программы lab3.lst


```
= '$                    EOL EQU '$

                        ;----------PЎC,PµPє PïCЂPsPiCЂP°PjPjC‹---------
                        -
0000                    AStack SEGMENT STACK
0000  0020[             DW 32 DUP(?)
   ????
                ]

0040                    AStack ENDS

                        ;----------P”P°PSPSC‹Pµ PïCЂPsPiCЂP°PjPjC‹-----
                        ---
0000                    DATA SEGMENT
0000  0080[             buffer DB 128 DUP(?)
   ??
                ]

0080  0003              a DW 3
0082  FFFB              b DW -5
0084  FFFE              i DW -2
0086  0001              k DW 1
0088  0000              i1 DW ?
008A  0000              i2 DW ?
008C  0000              res DW ?
008E                    DATA ENDS

                        ;----------PљPsPґ PïCЂPsPiCЂP°PjPjC‹-----------
0000                    CODE SEGMENT
```

```
                        ASSUME CS:CODE, DS:DATA, SS:AStack

                        ;--------P“PsP»PsPIPSP°CЏ PїCЂPsC†PµPrCґCЂP°---
                        -----
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                    f1:                 ; P’C‹C‡PëCЃP»CЏPµPj f1
0009  A1 0084 R               mov ax, i
000C  D1 E0                  shl ax, 1      ; PJPjPSPsP¶P°PµPj i PS
                        P° 4
000E  D1 E0                  shl ax, 1
0010  8B 1E 0080 R                mov bx, a
0014  3B 1E 0082 R                cmp bx, b    ; PЎCЂP°PIPSPëPIP°PµPj
                        a Pë b
0018  7E 0A                  jle f1_case2 ; P•CЃP»Pë a <= b, PїPµ
                        CЂPµP№C,Pë Pє f2_case2

001A                    f1_case1:           ; a > b
001A  05 0003                      add ax, 3     ; 4i + 3
001D  F7 D8                  neg ax
Microsoft (R) Macro Assembler Version 5.10          11/2/21 13:16:09
                                    Page     1-2


001F  8B C8                  mov cx, ax   ; cx: -4i - 3
0021  EB 0C 90                     jmp f1_end

0024                    f1_case2:           ; a <= b
0024  8B D8                  mov bx, ax   ; bx = ax = 4i
```

```
0026  D1 EB                 shr bx, 1      ; bx = 2i
0028  03 C3                 add ax, bx     ; ax = 4i + 2i = 6i
002A  8B C8                 mov cx, ax     ; cx = 6i
002C  83 E9 0A                    sub cx, 10     ; cx = 6i - 10

002F                  f1_end:
002F  89 0E 0088 R                 mov i1, cx     ; i1 = f1(a, b, i)


0033                  f2:                  ; P'C‹C‡PëCЃP»CЏPµPj f2
0033  A1 0084 R             mov ax, i
0036  8B 1E 0080 R                mov bx, a
003A  3B 1E 0082 R                cmp bx, b
003E  7E 0E                  jle f2_case2 ; P•CЃP»Pë a <= b, PїPµ
                  CЂPµP№C,Pë Pє f2_case2

0040                  f2_case1:
0040  D1 E0                      shl ax, 1      ; PJPjPSPsP¶P°PµPj i PS
                  P° 4
0042  D1 E0                      shl ax, 1
0044  2D 0005                      sub ax, 5      ; ax = 4i - 5
0047  F7 D8                  neg ax               ; ax = -4i + 5
0049  8B C8                  mov cx, ax
004B  EB 10 90                     jmp f2_end

004E                  f2_case2:
004E  8B D8                      mov bx, ax   ; bx = i
0050  D1 E0                      shl ax, 1      ; ax = 2i
0052  03 C3                      add ax, bx     ; ax = 3i
0054  8B D8                      mov bx, ax   ; bx = 3i
0056  B8 000A                      mov ax, 10   ; ax = 10
0059  2B C3                      sub ax, bx     ; ax = 10 - 3i
005B  8B C8                      mov cx, ax
```

```
005D                    f2_end:
005D  89 0E 008A R              mov i2, cx

0061                    f3:             ; P'C‹C‡PëCЃP»CЏPµPj f3
0061  8B 1E 0086 R             mov bx, k
0065  83 FB 00                  cmp bx, 0
0068  7D 26                 jge f3_case2       ; P•CЃP»Pë k >= 0, PїPµ
                    CЂPµP№C,Pë Pє f3_case2

006A                    f3_case1:
006A  A1 0088 R             mov ax, i1
006D  8B 1E 008A R              mov bx, i2
0071  2B C3                 sub ax, bx    ; i1-i2
0073  3D 0000                   cmp ax, 0
0076  7C 03                 jl abs         ; P•CЃP»Pë i1 - i2 < 0,
Microsoft (R) Macro Assembler Version 5.10          11/2/21 13:16:09
                                Page     1-3

                     PSP°P№PrC'Pj PjPsPrCґP»CЊ PIC‹CЂP°P¶PµPSPëCЏ
0078  EB 03 90                jmp min          ; P'PsP»CЊC€Pµ 0, PїPµC
                    ЂPµP№C,Pë Pє PїPsPëCЃPєCґ PjPëPSPëPjP°P»CЊPSPsP
                    iPs C‡PëCЃP»P° PI PїP°CЂPµ
007B                    abs:
007B  F7 D8                 neg ax

007D                    min:
007D  BB 0002                   mov bx, 2
0080  3D 0002                   cmp ax, 2
0083  7E 06                 jle absmin    ; P•CЃP»Pë |i1 - i2| <=
                     2, PїPµCЂPµP№C,Pë Pє absmin
0085                    twomin:
0085  B9 0002                   mov cx, 2
0088  EB 1A 90                 jmp f3_end
008B                    absmin:
```

```
008B  8B C8                   mov cx, ax
008D  EB 15 90                jmp f3_end

0090                    f3_case2:
0090  A1 008A R               mov ax, i2
0093  F7 D8                   neg ax
0095  3D FFFA                 cmp ax, -6
0098  7D 08                   jge i2max     ; P•CЃP»Pë -i2 >= -6, P
                    ïPµCЂPµP№C,Pë Pє i2max
009A                    neg6max:
009A  B9 0006                 mov cx, 6
009D  F7 D9                   neg cx
009F  EB 03 90                jmp f3_end
00A2                    i2max:
00A2  8B C8                   mov cx, ax
00A4                    f3_end:
00A4  89 0E 008C R            mov res, cx
00A8  CB                      ret

00A9                    Main ENDP
00A9                    CODE ENDS
                    END Main
Microsoft (R) Macro Assembler Version 5.10          11/2/21 13:16:09
                                                    Symbols-1


Segments and Groups:

                N a m e         Length   AlignCombine Class

ASTACK . . . . . .              0040  PARA      STACK
CODE . . . . . .                00A9 PARA       NONE
DATA . . . . . .                008E  PARA      NONE
```

Symbols:

```
            N a m e         Type   Value        Attr

A . . . . . . . . . . . . . .   L WORD    0080  DATA
ABS . . . . . . . . . . . . .   L NEAR    007B  CODE
ABSMIN . . . . . . . . . . . .      L NEAR    008B  CODE

B . . . . . . . . . . . . . .   L WORD    0082  DATA
BUFFER . . . . . . . . . . . .      L BYTE    0000  DATA       Length = 0080

EOL . . . . . . . . . . . . .   TEXT  '$

F1 . . . . . . . . . . . . . .   L NEAR    0009  CODE
F1_CASE1 . . . . . . . . . . .      L NEAR    001A  CODE
F1_CASE2 . . . . . . . . . . .      L NEAR    0024  CODE
F1_END . . . . . . . . . . . .   L NEAR    002F  CODE
F2 . . . . . . . . . . . . . .   L NEAR    0033  CODE
F2_CASE1 . . . . . . . . . . .      L NEAR    0040  CODE
F2_CASE2 . . . . . . . . . . .      L NEAR    004E  CODE
F2_END . . . . . . . . . . . .   L NEAR    005D  CODE
F3 . . . . . . . . . . . . . .   L NEAR    0061  CODE
F3_CASE1 . . . . . . . . . . .      L NEAR    006A  CODE
F3_CASE2 . . . . . . . . . . .      L NEAR    0090  CODE
F3_END . . . . . . . . . . . .   L NEAR    00A4  CODE

I . . . . . . . . . . . . . .   L WORD    0084  DATA
I1 . . . . . . . . . . . . . .   L WORD    0088  DATA
I2 . . . . . . . . . . . . . .   L WORD    008A  DATA
I2MAX . . . . . . . . . . . .   L NEAR    00A2  CODE

K . . . . . . . . . . . . . .   L WORD    0086  DATA

MAIN . . . . . . . . . . . . .   F PROC    0000  CODE       Length = 00A9
```

MIN . . . . . L NEAR 007D CODE

NEG6MAX . . . . . L NEAR 009A CODE

RES . . . . . L WORD 008C DATA

TWOMIN . . . . . L NEAR 0085 CODE

@CPU . . . . . TEXT 0101h
@FILENAME . . . . . TEXT lab3
@VERSION . . . . . TEXT 510
Microsoft (R) Macro Assembler Version 5.10 11/2/21 13:16:09
Symbols-2

126 Source Lines
126 Total Lines
36 Symbols

47970 + 457240 Bytes symbol space free

0 Warning Errors
0 Severe Errors