МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Организация ЭВМ и систем»

Тема: Представление и обработка целых чисел. Организация ветвящихся процессов

Студент гр. 9383 _____ Камзолов Н.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Научиться представлять и обрабатывать целые числа. Познакомиться с ветвящимися процессами.

Текст задания(Вариант 4).

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.

$$f1 = \begin{cases} 15-2*i, & \text{при } a>b \\ 3*i+4, & \text{при } a<=b \end{cases} \qquad f5 = \begin{cases} 20 - 4*i, & \text{при } a>b \\ -(6*I - 6), & \text{при } a<=b \end{cases}$$

$$f4 = \begin{cases} \min(|i1 - i2|, 2), & \text{при } k<0 \\ \max(-6, -i2), & \text{при } k>=0 \end{cases}$$

Ход работы.

Были использованы следующие команды для выполнения данного задания:

1. shl – логический сдвиг влево, равнозначный умножению на 2.
2. cmp – сравнение чисел, которое меняет флаги в программе.
3. jmp – безусловный переход по заданной метке.
4. jle – условный переход по заданной метке, в том случае если первый аргумент <= второму аргументу.
5. jge - условный переход по заданной метке, в том случае если первый аргумент >= второму аргументу.

6. jl – условный переход по заданной метке, в том случае если первый аргумент < второго аргумента.
7. jg – условный переход по заданной метке, в том случае если первый аргумент > второго аргумента.

Тестирование.

Входные данные	Ожидаемый результат	Результат работы программы
a = 1, b = 2, i = 4, k = 1	i1 = 16, i2= -18, res = 18	i1 = 16, i2= -18, res = 18
a = 2, b = 1, i = 3, k = 1	i1 = 9, i2= 8, res = -6	i1 = 9, i2= 8, res = -6
a = 2, b = 1, i = 3, k = -1	i1 = 9, i2= 8, res = 1	i1 = 9, i2= 8, res = 1
a = 1, b = 2, i = 4, k = -1	i1 = 16, i2= -18, res = 2	i1 = 16, i2= -18, res = 2

Выводы.

Приобретены знания о представлении и обработке целых чисел. Произошло знакомство с ветвящимися процессами.

ПРИЛОЖЕНИЕ А
КОД ПРОГРАММЫ

Название файла: lab3.asm

```
DATA SEGMENT
RES     DW      ?
A       DW      1
B       DW      2
I       DW      4
K       DW      -1
I1      DW      ?
I2      DW      ?
DATA ENDS

ASTACK SEGMENT STACK
        DW 16 DUP(?)
ASTACK ENDS

CODE SEGMENT
        ASSUME CS:CODE, SS:ASTACK, DS:DATA
MAIN PROC FAR
        MOV AX, DATA
        MOV DS, AX
F1:
        MOV AX, A
        CMP AX, B ;СРАВНЕНИЕ A И Б
        JLE F1_SECOND ;A <= B
        MOV AX, I ;
        SHL AX, 1 ;
        NEG AX ;
        ADD AX, 15 ; ПОМЕЩАЕМ В AX 15-2I
        MOV I1, AX ; ПОМЕЩАЕМ В I1 ЗНАЧЕНИЕ ИЗ AX

        JMP F2
F1_SECOND: ;JUMP СЮДА ЕСЛИ A <= B
```

```
        MOV AX, I ;
        SHL AX, 1 ;
        ADD AX, I ;
        ADD AX, 4 ;Помещаем в AX 3I+4
        MOV I1, AX ;Помещаем в I1 значение из AX

F2_SECOND:;если A <= B
        MOV AX, I;
        SHL AX, 1;
        ADD AX, I;
        SHL AX, 1;
        NEG AX;
        ADD AX, 6; Помещаем в AX -6I+6
        MOV I2, AX ;Помещаем в I2 значение из AX

        JMP FK_4
F2:; JUMP СЮДА ЕСЛИ A > B
        MOV AX, I;
        SHL AX, 1;
        SHL AX, 1;
        NEG AX;
        ADD AX, 20;Помещаем в AX 20-4I
        MOV I2, AX ;Помещаем в I2 значение из AX
FK_4:
        MOV AX, K
        CMP AX, 0
        JGE FK4_SECOND ;K >= 0
        MOV AX, I1;
        SUB AX, I2;Помещаем в AX I1-I2
        CMP AX, 0;Сравниваем I1-I2 и 0
        JL F_NEG ;I1-I2 < 0
        JMP FK_4_FIRST_FINAL

F_NEG:;если I1-I2 < 0 и нужен модуль
        NEG AX;Переводим в отрицательное(нужен модуль)
```

```
        JMP FK_4_FIRST_FINAL

FK4_SECOND:;ЕСЛИ K>=0
        MOV AX, I2;
        NEG AX;ПОМЕЩАЕМ В AX -I2
        CMP AX, -6;СРАВНИВАЕМ AX И -6
        JL FMAX ; -I2 < -6
        MOV RES, AX ;ПОМЕЩАЕМ AX В РЕЗУЛЬТАТ
        JMP F_END
FMAX:;ЕСЛИ -I2 < -6
        MOV RES, -6 ;ПОМЕЩАЕМ -6 В РЕЗУЛЬТАТ
        JMP F_END

FK_4_FIRST_FINAL:
        CMP AX, 2;СРАВНИВАЕМ AX И 2
        JG FMIN ;|I1-I2| > 2
        MOV RES, AX; ПОМЕЩАЕМ В RES AX
        JMP F_END
FMIN:; ЕСЛИ |I1-I2| > 2
        MOV RES, 2;ПОМЕЩАЕМ 2 В РЕЗУЛЬТАТ
        JMP F_END
F_END:
        MOV AH, 4CH
        INT 21H
MAIN  ENDP
CODE ENDS
END MAIN
```

LAB3.LST:

```
 MICROSOFT (R) MACRO ASSEMBLER VERSION 5.10                10/26/20
21:16:2
                                                          PAGE     1-1
```

```
0000                    DATA SEGMENT
0000  0000              RES     DW      ?
0002  0001              A       DW      1
0004  0002              B       DW      2
0006  0004              I       DW      4
0008  FFFF              K       DW      -1
000A  0000              I1      DW      ?
000C  0000              I2      DW      ?
000E                    DATA ENDS

0000                    ASTACK SEGMENT STACK
0000  0010[                    DW 16 DUP(?)
      ????
              ]

0020                    ASTACK ENDS

0000                    CODE SEGMENT
                          ASSUME CS:CODE, SS:ASTACK, DS:DATA
0000                    MAIN PROC FAR
0000  B8 ---- R                MOV AX, DATA
0003  8E D8                    MOV DS, AX
0005                    F1:
0005  A1 0002 R                MOV AX, A
0008  3B 06 0004 R             CMP AX, B ;СРАВНЕНИЕ А И Б
000C  7E 10                    JLE F1_SECOND ;A <= B
000E  A1 0006 R                MOV AX, I ;
0011  D1 E0                    SHL AX, 1 ;
0013  F7 D8                    NEG AX ;
0015  05 000F                  ADD AX, 15 ; ПОМЕЩАЕМ В АХ 15-
                  2I
0018  A3 000A R                MOV I1, AX ; ПОМЕЩАЕМ В I1 З
                  АЧЕНИЕ ИЗ АХ

001B  EB 26 90                 JMP F2
```

```
001E                    F1_SECOND: ;JUMP СЮДА ЕСЛИ A <= B
001E  A1 0006 R                 MOV AX, I ;
0021  D1 E0                     SHL AX, 1 ;
0023  03 06 0006 R              ADD AX, I ;
0027  05 0004                   ADD AX, 4 ;ПОМЕЩАЕМ В АХ 3I+4
002A  A3 000A R                 MOV I1, AX ;ПОМЕЩАЕМ В I1 ЗН
                    АЧЕНИЕ ИЗ АХ

002D                    F2_SECOND:;ЕСЛИ A <= B
002D  A1 0006 R                 MOV AX, I;
0030  D1 E0                     SHL AX, 1;
0032  03 06 0006 R              ADD AX, I;
0036  D1 E0                     SHL AX, 1;
0038  F7 D8                     NEG AX;
003A  05 0006                   ADD AX, 6; ПОМЕЩАЕМ В АХ -6I+6
003D  A3 000C R                 MOV I2, AX ;ПОМЕЩАЕМ В I2 ЗН
                    АЧЕНИЕ ИЗ АХ

0040  EB 10 90                  JMP FK_4
```

```
0043                    F2:; JUMP СЮДА ЕСЛИ A > B
0043  A1 0006 R                MOV AX, I;
0046  D1 E0                    SHL AX, 1;
0048  D1 E0                    SHL AX, 1;
004A  F7 D8                    NEG AX;
004C  05 0014                  ADD AX, 20;ПОМЕЩАЕМ В AX 20-4I
004F  A3 000C R                MOV I2, AX ;ПОМЕЩАЕМ В I2 ЗН
                    АЧЕНИЕ ИЗ AX
0052                    FK_4:
0052  A1 0008 R                MOV AX, K
0055  3D 0000                  CMP AX, 0
0058  7D 14                    JGE FK4_SECOND ;K >= 0
005A  A1 000A R                MOV AX, I1;
005D  2B 06 000C R             SUB AX, I2;ПОМЕЩАЕМ В AX I1-I2
0061  3D 0000                  CMP AX, 0;СРАВНИВАЕМ I1-I2 И
                     0
0064  7C 03                    JL F_NEG ;I1-I2 < 0
0066  EB 1F 90                 JMP FK_4_FIRST_FINAL

0069                    F_NEG:;ЕСЛИ I1-I2 < 0 И НУЖЕН МОД
                    ЛЬ
0069  F7 D8                    NEG AX;ПЕРЕВОДИМ В ОТРИЦ
                    АТЕЛЬНОЕ(НУЖЕН МОДУЛЬ)
006B  EB 1A 90                 JMP FK_4_FIRST_FINAL

006E                    FK4_SECOND:;ЕСЛИ K>=0
006E  A1 000C R                MOV AX, I2;
0071  F7 D8                    NEG AX;ПОМЕЩАЕМ В AX -I2
0073  3D FFFA                  CMP AX, -6;СРАВНИВАЕМ AX И -
                    6
0076  7C 06                    JL FMAX ; -I2 < -6
```

```
0078  A3 0000 R                 MOV RES, AX ;ПОМЕЩАЕМ AX В Р
                                ЗУЛЬТАТ
007B  EB 1E 90                  JMP F_END
007E                        FMAX:;ЕСЛИ -I2 < -6
007E  C7 06 0000 R FFFA         MOV RES, -6 ;ПОМЕЩАЕМ -6 В Р
                                ЗУЛЬТАТ
0084  EB 15 90                  JMP F_END

0087                        FK_4_FIRST_FINAL:
0087  3D 0002                      CMP AX, 2;СРАВНИВАЕМ AX И 2
008A  7F 06                     JG FMIN ;|I1-I2| > 2
008C  A3 0000 R                 MOV RES, AX; ПОМЕЩАЕМ В RES AX
008F  EB 0A 90                  JMP F_END
0092                        FMIN:; ЕСЛИ |I1-I2| > 2
0092  C7 06 0000 R 0002         MOV RES, 2;ПОМЕЩАЕМ 2 В РЕЗ
                            УЛЬТАТ
0098  EB 01 90                  JMP F_END
009B                        F_END:
009B  B4 4C                     MOV AH, 4CH
009D  CD 21                     INT 21H
009F                        MAIN  ENDP
009F                        CODE ENDS
                              END MAIN
```

SEGMENTS AND GROUPS:

N A M E	LENGTH	ALIGN	COMBINE	CLASS
ASTACK	0020	PARA	STACK	
CODE	009F	PARA	NONE	
DATA	000E	PARA	NONE	

SYMBOLS:

N A M E	TYPE	VALUE	ATTR
A	L WORD	0002	DATA
B	L WORD	0004	DATA
F1	L NEAR	0005	CODE
F1_SECOND	L NEAR	001E	CODE
F2	L NEAR	0043	CODE
F2_SECOND	L NEAR	002D	CODE
FK4_SECOND	L NEAR	006E	CODE
FK_4	L NEAR	0052	CODE
FK_4_FIRST_FINAL	L NEAR	0087	CODE
FMAX	L NEAR	007E	CODE
FMIN	L NEAR	0092	CODE
F_END	L NEAR	009B	CODE
F_NEG	L NEAR	0069	CODE
I	L WORD	0006	DATA
I1	L WORD	000A	DATA
I2	L WORD	000C	DATA

```
K  . . . . . .    L WORD    0008 DATA

MAIN . . . . . .    F PROC    0000 CODE LENGTH =
009F

RES  . . . . . .    L WORD    0000 DATA

@CPU . . . . . .    TEXT  0101H
@FILENAME  . . . . . .    TEXT  LAB3
@VERSION . . . . . .    TEXT  510


    93 SOURCE  LINES
    93 TOTAL   LINES
    27 SYMBOLS

  47974 + 457236 BYTES SYMBOL SPACE FREE

     0 WARNING ERRORS
     0 SEVERE  ERRORS
```