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. 1303 _____ Куклина Ю.Н.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Разработать на языке Ассемблера программу, которая по заданным целочисленным значениям параметров вычисляет значения функций.

Задание.

Разработать на языке Ассемблера программу, которая по заданным целочисленным значениям параметров a, b, i, k вычисляет: a) значения функцийi1 = 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 маршрутывыполнения программы, а также различные знаки параметров a и b.

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

$$f6 = \begin{cases} 2*(i+1) - 4, & \text{при } a>b \\ 5 - 3*(i+1), & \text{при } a<=b \end{cases}$$

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

Выполнение работы:

1. Из таблицы получен вариант набора функций, который необходимо реализовать.
2. Программа протранслирована с различными значениями переменных.
3. Для выполнения данного задания были использованы следующие команды общего назначения:

Mov – присваивание

Cmp – сравнение

Sub – вычитание

Add – сложение

Neg – смена знака

Sal – арифметический сдвиг влево

Sar – арифметический сдвиг вправо

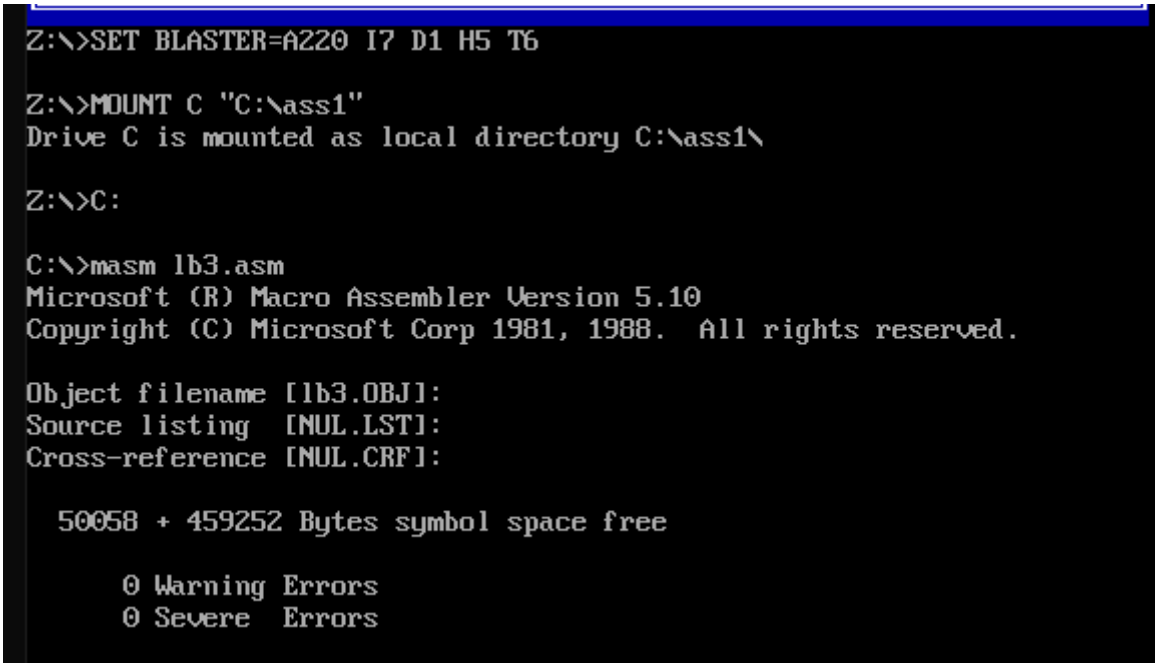
Jmp – безусловный переход

Jg – выполняет короткий переход, если первый операнд больше второго при выполнении операции сравнения с помощью команды cmp.

JGe – выполняет короткий переход, если первый операнд больше или равен второму при выполнении операции сравнения с помощью команды cmp.

4. Также были использованы метки (например PART1) для перехода между некоторыми командами.

5. Трансляция программы



Исходный код программы представлен в приложении А.

Приложение А

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

AStack    SEGMENT  STACK
          DW 12 DUP(0)
AStack    ENDS

DATA      SEGMENT

a   DW  ?
b   DW  ?
i   DW  ?
k   DW  ?

i1  DW  ?       ;f3
i2  DW  ?       ;f6
res DW  ?       ;f4
DATA      ENDS

CODE SEGMENT

Main      PROC  FAR
        push DS
        sub ax,ax
        push ax
        mov   AX,DATA
        mov   DS,AX
        ;Вычисление f1 и f2
        mov ax,a    ;ax = a
        mov cx,i    ;cx = i
        cmp ax,b    ;Сравнение значений a и b
        jg PART1    ;если a>b то на PART1

        ;если a<=b:
        add cx,i   ;cx=2i
        add cx,i   ;cx=3i
        sal cx,1    ;cx = 6i
        mov ax,8    ;ax = 8
        sub ax,cx   ;ax = 8 - 6i
        mov i1,ax   ;i1(f1) = cx = 8-6i

        shr ax, 1   ;ax = 4 - 3i
        sub ax, 2  ;ax = ax - 2 = 2 - 3i
        mov i2,ax   ;i2(f2) = ax = 2-3i
        jmp PART2   ;идем на PART2

PART1:          ;если a>b
        mov cx,i    ;cx = i
        sal cx,1    ;i = i*4
        sal cx,1
        mov ax,7   ;ax = 7
        sub ax,cx   ;ax = ax - cx = 7 - 4i
        mov i1,ax   ;i1(f1) = cx = 7 - 4i

        sal ax,1 ; ax=3-2i
        sub ax,1   ;ax = ax - cx = 2 - 2i
        neg ax
```

```
        mov i2,ax   ;i2(f2) = cx = -(2 - 2i)

        ;Вычисление f3
PART2:
        mov ax,k

        cmp ax,0    ;сравниваем k и 0
        JGe PART4   ;если k больше или равно 0, то на PART4

                ;если k<0
        mov ax,i1   ;ax = i1
        cmp ax,i2
        JGe PART3   ;если i1 >= i2 то на PART3

        sub ax,i1
        cmp ax,2
        JGe PART6

        mov res,ax  ;res = ax
        jmp ENDPART


PART3:
        sub ax,i2 ; вычисляем разность i1 и i2 если i1 больше
        cmp ax,2
        JGe PART6   ;если модуль больше 2, то на PART6

        mov res,ax  ;res = ax
        jmp ENDPART

PART4:
                ;если k больше или равно 0,
        mov bx,i2
        neg i2
        cmp i2,-6
        JGe PART5   ;если i2 >= -6 то на PART5

        mov res,-6  ;res = -6
        jmp ENDPART

PART5:          ;если i2 >= -6
        mov res,bx  ;res = i2
        jmp ENDPART
PART6:
        mov res,2  ;res = 2
        jmp ENDPART

ENDPART:
        int 20h

Main      ENDP
CODE      ENDS
          END Main
```