# 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. 1303 Чубан Д.В.

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:

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

1. 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

Были реализованы функции из Каталога Заданий, соответствующие 3 Варианту. Реализованная программа протранслирована с различными тестовыми данными.

Для выполнения данного задания были использованы такие команды общего назначения как:

Команды передачи данных.

* 1. *mov* – присваивание

Двоичные арифметические команды.

1. *add* - сложение
2. *sub* - вычитание
3. *cmp* – сравнение
4. *neg* – смена знака Команды побитового сдвига.

1) *sal* - арифметический сдвиг влево Команды передачи управления.

1. *jmp* – команда безусловного перехода
2. *Int* - вызов программного прерывания
3. *jg(jump greater)* - выполняет короткий переход, если первый операнд больше второго операнда при выполнении операции сравнения с помощью команды cmp.
4. *jl(jump less)* - выполняет короткий переход, если первый операнд меньше второго операнда при выполнении операции сравнения с помощью команды cmp.

Для реализации ветвления в программе использовались метки. Метка - это символьное имя, обозначающее ячейку памяти, которая содержит некоторую команду.

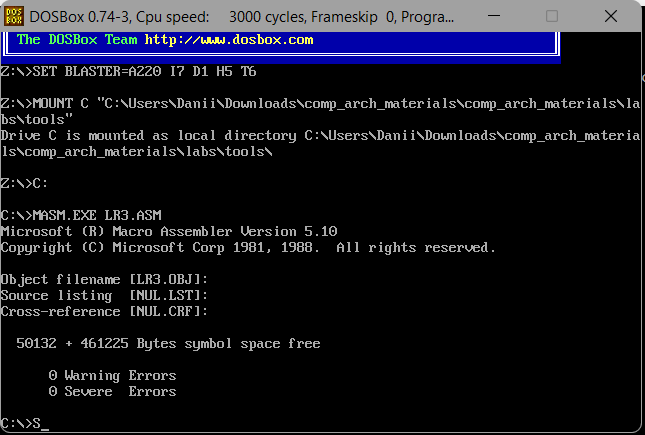


рис.1 -Трансляция программы

# Тестирование

Программа выполнена в пошаговом режиме под управлением отладчика с фиксацией значений используемых переменных.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тестируемый случай | Функции для данного случая | Данные | |
| входные | выходные |
| 1 | a > b k = 0 | f1 = 15 – 2\*i f2 = 7 - 4\*i  f3 = abs(i1 - i2) | a = 1, b = 0  k = 0  i = 1 | f1 = 13 = 000D f2 = 3 = 0003 f3 = 10 = 000A |
| 2 | a > b k = -1 | f1 = 15 – 2\*i f2 = 7 - 4\*i  f3 = max(i1,10-i2) | a = 1, b = 0  k = -1  i = 1 | f1 = 13 = 000D f2 = 3 = 0003 f3 = 13 = 000D |
| 3 | a <= b k = 0 | f1 = 3\*i + 4 f2 = 8 -6\*i  f3 = abs(i1 - i2) | a = 1, b = 1  k = 0  i = 1 | f1 = 7 = 0007  f2 = 2 = 0002  f3 = 5 = 0005 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | a <= b k = -1 | f1 = 3\*i + 4 f2 = 8 -6\*i  f3=max(i1,10-i2) | a = 1, b = 1  k = -1  i = 1 | f1 = 7 = 0007  f2 = 2 = 0002  f3 = 8 = 0008 |

# Выводы

В ходе выполнения лабораторной работы были получены навыки разработки программы с заданными целочисленными значениями на языке программирования Ассемблер.

**ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММЫ**

Название файла: lb3.asm

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

**AStack SEGMENT STACK**

**DW 12 DUP('!')**

**AStack ENDS**

|  |  |  |
| --- | --- | --- |
| **DATA** |  | **SEGMENT** |
| **a DW** | **0** |  |
| **b DW** | **0** |  |
| **i DW** | **0** |  |
| **k DW** | **0** |  |

**i1 DW 0**

**i2 DW 0**

**res DW 0**

**DATA ENDS CODE SEGMENT**

**Main PROC FAR**

**push DS sub AX,AX push AX**

**mov AX, DATA mov DS, AX**

**mov AX,a mov CX,i cmp AX,b jg True**

**sal CX,1 add CX,i mov i1,CX**

**add i1,4**

**sal CX,1 mov i2,8 sub i2,CX**

**jmp F3**

**True:**

**sal CX,1 mov i1,15 sub i1,CX**

**sal CX,1 mov i2,7 sub i2,CX**

**F3:**

**cmp k,0 jl K\_LESS mov AX,i1 sub AX,i2 mov res,AX cmp res,0 jl NEGRES**

**jmp QUIT**

**NEGRES:**

**neg res jmp QUIT**

**K\_LESS:**

**mov AX,10 sub AX,i2 cmp i1,AX**

**jg i1\_GREATER mov res,AX jmp QUIT**

**i1\_GREATER:**

**mov AX,i1 mov res,AX**

**QUIT:**

**int 20**

**Main ENDP**

**CODE ENDS**

**END Main**