

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

Андреева Е.А.

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 $i2 = f2(a,b,i)$;

б) 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 .

$$\begin{aligned} f1 &= \begin{cases} / 15-2*i, & \text{при } a>b \\ \backslash 3*i+4, & \text{при } a\leq b \end{cases} \\ f2 &= \begin{cases} / -(4*i+3), & \text{при } a>b \\ \backslash 6*i-10, & \text{при } a\leq b \end{cases} \end{aligned} \quad f3 = \begin{cases} / \min(i1,i2), & \text{при } k=0 \\ \backslash \max(i1,i2), & \text{при } k\neq 0 \end{cases}$$

Выполнение работы

1. Из таблицы получен вариант набора функций, которые необходимо реализовать, приведенного в каталоге Задания.

2. Программа протранслирована с различными значениями переменных, результат выполнения набора функций зафиксирован в таблице;

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

В сегменте данных заданы метки для переменных a , b , i , k , $i1$, $i2$, res использующихся в программе. Их значения изначально равны 0, в режиме отладки их можно менять.

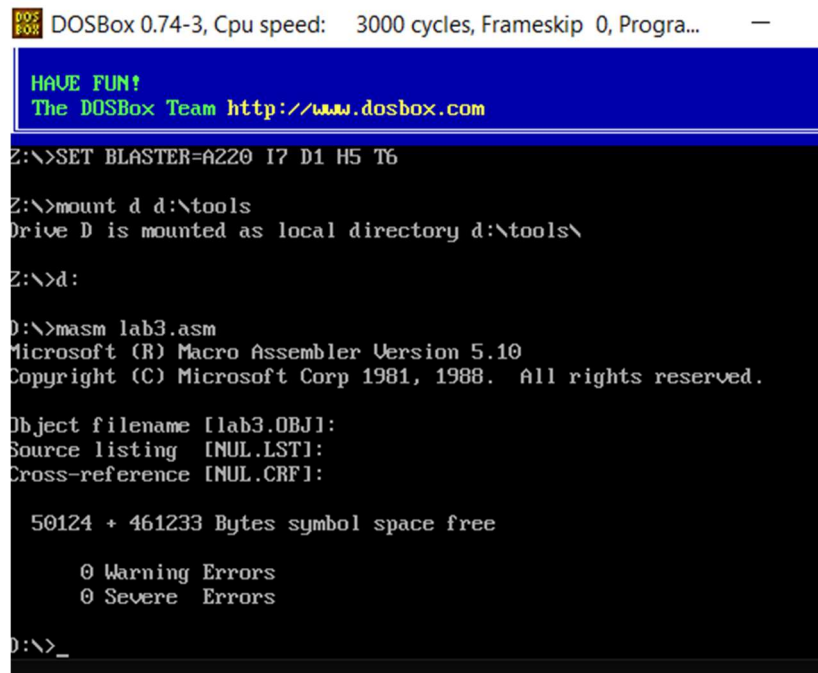
Так как запрещено использование процедур в программе, функции были реализованы при помощи фрагментов кода, размеченных метками, с безусловными переходами на них. “Вызов” функции безусловным переходом `jmp` к метке данной функции.

Функции содержат ветвление. Их поведение зависит от состояния переменных a , b , следовательно функции $f1$ и $f2$ логически подразделяются на три части:

сравнение переменных a и b и две ветви, переход в которые осуществляется непосредственно после сравнения.

Исполнение функции $f3$ происходит аналогичным образом, но использует значения рассчитанные в двух предыдущих функциях

Запуск программы



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...  
HAVE FUN!  
The DOSBox Team http://www.dosbox.com  
Z:\>SET BLASTER=A220 I7 D1 H5 T6  
  
Z:\>mount d d:\tools  
Drive D is mounted as local directory d:\tools\  
  
Z:\>d:  
  
D:\>masm lab3.asm  
Microsoft (R) Macro Assembler Version 5.10  
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.  
  
Object filename [lab3.OBJ]:  
Source listing [NUL.LST]:  
Cross-reference [NUL.CRF]:  
  
50124 + 461233 Bytes symbol space free  
  
0 Warning Errors  
0 Severe Errors  
  
D:\>_
```

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: lab3.asm

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

```
AStack    SEGMENT    STACK
           DW 32 DUP(0)
AStack    ENDS
```

```
DATA      SEGMENT
```

```
i        DW    0
a        DW    0
b        DW    0
k        DW    0
```

```
i1       DW    0           ;f1
i2       DW    0           ;f2
res      DW    0           ;f3
```

```
DATA      ENDS
```

```
CODE SEGMENT
```

```
Main      PROC    FAR
           mov     AX,DATA
           mov     DS,AX
```

```
;Вычисление f1 и f2
```

```
           mov ax,a      ;ax = a
           mov dx,b      ;dx = b
           mov cx,i      ;cx = i
           cmp ax,dx     ;Сравнение значений a и b
           jg PART1      ;если a>b то на PART1
```

```
;если a<=b:
```

```
           mov ax,i      ;ax = i
           sal cx,1       ;cx = i*2
```

```

add cx,ax ;cx = 2*i + i = 3*i
mov ax,4 ;ax = 4
add cx,ax ;cx = 3*i + 4
mov i1,cx ;i1(f1) = cx = 3i + 4

mov cx,i ;cx = i
mov ax, i ;ax = i
sal ax,1 ;ax = 2i
sal cx,1 ;cx = 2i
sal cx,1 ;cx = 4i
add cx,ax ;cx = 4i + 2i = 6i
mov ax, 10 ;ax = 10
sub cx, ax ;cx = cx - ax = 6i - 10
mov i2,cx ;i2(f2) = cx = 6i - 10
jmp PART2 ;идем на PART2

```

```

PART1: ;если a>b
mov cx,i ;cx = i
sal cx,1 ;i = i*2
mov ax,15 ;ax = 15
sub ax,cx ;ax = ax - cx = 15 - 2i
mov i1,ax ;i1(f1) = cx = 15 - 2i

mov ax,cx ;ax = 2*i
sal cx,1 ; = 2i*2 = 4i
mov ax,3 ;ax = 3
add ax,cx ;ax = ax + cx = 3 + 4i
neg ax ;ax = -(3 + 4i)
mov i2,ax ;i2(f2) = cx = -(3 + 4i)

```

;Вычисление f3

```

PART2:
mov ax,k
mov bx,0

cmp ax,bx ;сравниваем k и 0
JNe PART4 ;если k не равно 0, то на PART4

```

;если k = 0

```
mov ax,i1 ;ax = i1
mov bx,i2 ;bx = i2
cmp ax,bx
JGe PART3 ;если i1 >= i2 то на PART3
```

```
mov res,ax ;res = ax = i1
jmp ENDPART
```

PART3:

```
mov res,bx ;res = bx = i2
jmp ENDPART
```

PART4:

```
                                ;если k не равно 0
mov ax,i1 ;ax = i1
mov bx,i2 ;bx = i2
cmp ax,bx
JGe PART5 ;если i1 >= i2 то на PART5

mov res,bx ;res = bx = i2
jmp ENDPART
```

PART5: ;если i1 >= i2

```
mov res,ax ;res = ax = i1
```

ENDPART:

```
int 20h
```

```
Main      ENDP
CODE      ENDS
          END Main
```