

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

Хулап О.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 .

$$/ 2*(i+1) -4, \quad a > b$$

$$f6 = <$$

$$\backslash 5 - 3*(i+1), \text{ где } a \leq b$$

$$/ - (6*i+8), \text{ где } a > b$$

$$f8 = <$$

$$\backslash 9 - 3*(i-1), \text{ где } a \leq b$$

$$/ \min(i1,i2), \text{ где } k=0$$

$$f1 = <$$

$$\backslash \max(i1,i2), \text{ где } k \neq 0$$

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.

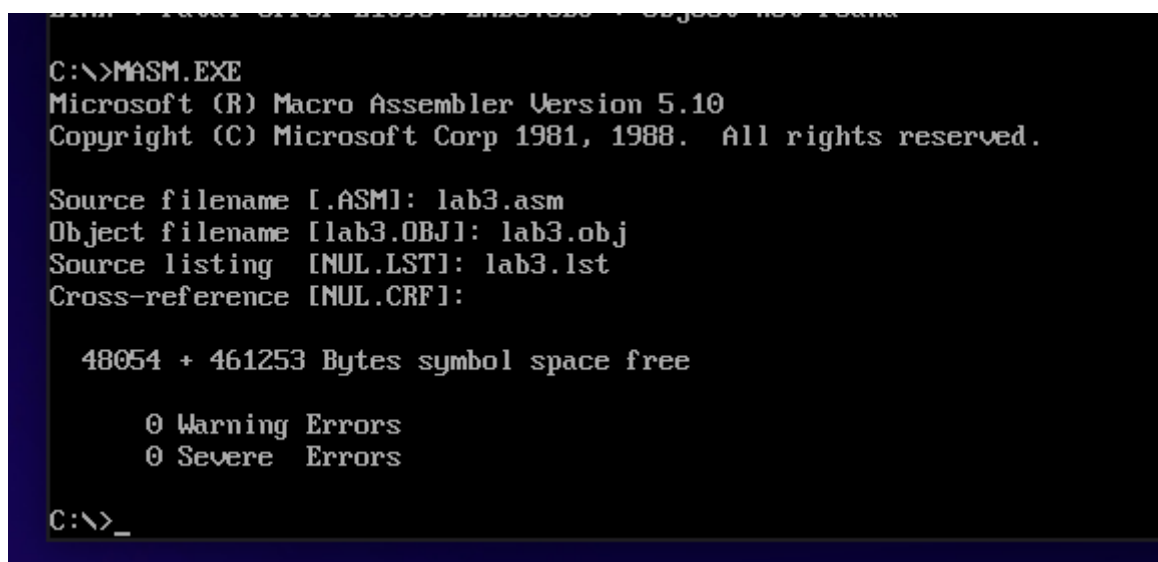
В сегменте данных заданы метки для переменных a, b, i, k, i1, i2, res использующихся в программе. Их значения изначально равны 0, в режиме отладки их можно менять.

В программе запрещено использовать процедуры, поэтому функции были реализованы с помощью фрагментов кода, размеченных метками, с безусловными переходами на них. Вызов функции безусловным переходом jmp к метке данной функции.

В функциях есть ветвление. То есть 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



```
C:\>MASM.EXE
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Source filename [.ASM]: lab3.asm
Object filename [lab3.OBJ]: lab3.obj
Source listing [NUL.LST]: lab3.lst
Cross-reference [NUL.CRF]:

48054 + 461253 Bytes symbol space free

0 Warning Errors
0 Severe Errors

C:\>_
```

```

;      / 2*(i+1) -4 , iðè a>b
;f6 = <
;      \ 5 - 3*(i+1), iðè a<=b

;      / - (6*i+8) , iðè a>b
;f8 = <
;      \ 9 -3*(i-1), iðè a<=b

;      / min(i1,i2), iðè k=0
;f1 = <
;      \ max(i1,i2), iðè k/=0

```

```

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

;Â÷ëäîå f1 è f2

mov ax, a

mov cx, i

cmp ax, b

jg a_greater

; a<=b

sal cx, 1 ; $i \ll 1 = i * 2$

add cx, i ; $i * 2 + i = i * 3$

neg cx ; $-i * 3$

add cx, 2 ; $2 - i * 3 = 5 - 3(i + 1)$

mov i1, cx

add cx, 8

mov i2, cx ; $10 - i * 3 = 9 - 3 * (i - 1)$

jmp F3

a_greater:

```
mov ax, 2
neg ax
sal cx, 1
sub ax, cx;  $-2 + 2*i = 2*(i + 1) - 4$ 
mov i1, ax

mov ax, 8
neg ax
add cx, i ;  $i*2 + i = 3*i$ 
sal cx, 1 ;  $3i*2$ 
neg cx ;  $-6*i$ 
sub ax, cx ;  $-8 - 6*i = -(6*i + 8)$ 
mov i2, ax
```

F3:

```
mov ax, k
cmp ax, 0 ;  $k \neq 0$ 
JNe K_NOT_ZERO ;  $k \neq 0$ , ôi îa K_NOT_ZERO
```

; $k = 0$

```
mov ax, i1 ;  $ax = i1$ 
cmp ax, i2
JGe MIN ;  $i1 \geq i2$  ôi îa MIN
mov res, ax ;  $res = ax = i1$ 
jmp EXIT
```

MIN:

mov res, bx

jmp EXIT

K_NOT_ZERO:

mov ax, i1 ; ax = i1

cmp ax, i2

JGe MAX ; nếu i1 >= i2 thì là MAX

mov res, bx ; res = bx = i2

jmp EXIT

MAX:

mov res, ax

EXIT:

int 20h

Main ENDP

CODE ENDS

END Main