

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

Токун Г.С.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Изучить представление и обработку целых чисел, организацию ветвящихся процессов.

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

Вариант 9

(f1) $f2 = -(4*i+3)$, при $a>b$ | $6*i-10$, при $a\leq b$

(f2) $f4 = -(6*i-4)$, при $a>b$ | $3*(i+2)$, при $a\leq b$

(f3) $f7 = |i1| + |i2|$, при $k<0$ | $\max(6, |i1|)$, при $k\geq 0$

Выполнение работы.

Для работы программы были созданы сегмент стека AStack, сегмент данных Data, в котором хранятся переменные a , b , i , k , $i1$, $i2$ и $result$, сегмент кода Code.

В сегменте кода была написана процедура Main, в которой написаны инструкции для завершения работы программы, 11 меток для вычисления значений функций:

В метке function2 регистру bx присваивается значение равное $2i$, а cx - $3i$ для дальнейшего вычисления значений функций, сравнивается значение a и b. Если a строго больше b, то выполняется переход к метке greater1, иначе выполняются инструкции в метке greater2.

В метках greater1, greater2 вычисляются значения функций f1, f2, их значения присваиваются соответствующим переменным в метке function2_end, в этой же метке вычисляется модуль i1. После завершения команд одной из них, выполняется метка function3, в которой сравнивается значение k с нулём. Если k нестрого больше нуля, то выполняется метка positive_k, иначе negative_k.

В метке negative_k вычисляется модуль i2, выполняется метка positive_i, в которой суммируются значения двух функций i1 и i2.

Затем выполняется метка function3_end, в которой в переменную result сохраняется значение третьей функции.

В метке positive_k уже вычисленное значение модуля i1 сравнивается с 6, далее снова выполняется метка function3_end.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	a = 1, b = 2, i = 3 k = 4	i1 = 8, i2 = 15, result = 8	Программа работает правильно
2.	a = 2, b = -1, i = 3 k = - 1	i1 = -15, i2 = -14, result = 29	Программа работает правильно
3.	a = 1, b = 2, i = 3 k = -1	i1 = 8, i2 = 15, result = 23	Программа работает правильно
4.	a = 2, b = 1, i = 3 k = 4	i1 = -15 , i2 = -14, result = 15	Программа работает правильно

Выводы.

Были изучены представление и обработка целых чисел, организация ветвящихся процессов.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab3.asm

```
AStack SEGMENT STACK
    DW 12 DUP(?)
AStack ENDS

DATA SEGMENT
    a DW 2
    b DW 1
    i DW 3
    k DW -1
    i1 DW 0
    i2 DW 0
    result DW 0
DATA ENDS

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

Main    PROC FAR
    push ds
    push ax
    mov ax, DATA
    mov ds, ax
function12:
    mov ax, a
    mov bx, i
    shl bx, 1
    mov cx, i
    add cx, bx
    cmp ax, b
    jg greater_a
greater_b:
    mov bx, cx
    add cx, 6
    shl bx, 1
    sub bx, 10
    jmp function12_end
greater_a:
    shl bx, 1
    neg bx
    sub bx, 3
    shl cx, 1
    neg cx
    add cx, 4
function12_end:
    mov i1, bx
    mov i2, cx
    mov ax, i1
    cmp ax, 0
    jge function3
```

```

        neg ax
function3:
        cmp k, 0
        jge positive_k
negative_k:
        mov bx, i2
        cmp bx, 0
        jge positive_i2
        neg bx
positive_i2:
        add ax, bx
        jmp function3_end
positive_k:
        cmp ax, 6
        jge function3_end
        mov ax, 6
function3_end:
        mov result, ax
        ret
Main    ENDP
CODE    ENDS
        END Main

```

ПРИЛОЖЕНИЕ В

Название файла: lab3.lst

#Microsoft (R) Macro Assembler Version 5.10
11:44:16

11/7/22

Page

1-1

```
0000          AStack SEGMENT STACK
0000 000C[          DW 12 DUP(?)
      ????
      ]

0018          AStack ENDS

0000          DATA SEGMENT
0000 0001          a DW 1
0002 0002          b DW 2
0004 0003          i DW 3
0006 FFFF          k DW -1
0008 0000          i1 DW 0
000A 0000          i2 DW 0
000C 0000          result DW 0
000E          DATA ENDS

0000          CODE SEGMENT
      ASSUME CS:CODE, DS:DATA, SS:AStack

0000          Main      PROC FAR
0000 1E              push ds
0001 50              push ax
0002 B8 -----R      mov ax, DATA
0005 8E D8            mov ds, ax
0007          function12:
0007 A1 0000 R          mov ax, a
000A 8B 1E 0004 R      mov bx, i
000E D1 E3            shl bx, 1
0010 8B 0E 0004 R      mov cx, i
0014 03 CB            add cx, bx
0016 3B 06 0002 R      cmp ax, b
001A 7F 0D            jg greater_a
001C          greater_b:
001C 8B D9            mov bx, cx
001E 83 C1 06          add cx, 6
0021 D1 E3            shl bx, 1
0023 83 EB 0A          sub bx, 10
0026 EB 0F 90          jmp function12_end
0029          greater_a:
0029 D1 E3            shl bx, 1
002B F7 DB            neg bx
002D 83 EB 03          sub bx, 3
0030 D1 E1            shl cx, 1
0032 F7 D9            neg cx
0034 83 C1 04          add cx, 4
```

```

0037          function12_end:
0037  89 1E 0008 R          mov i1, bx
003B  89 0E 000A R          mov i2, cx
003F  A1 0008 R          mov ax, i1
0042  3D 0000          cmp ax, 0
0045  7D 02          jge function3
0047  F7 D8          neg ax
0049          function3:

```

```

#Microsoft (R) Macro Assembler Version 5.10
11:44:16

```

11/7/22

Page

1-2

```

0049  83 3E 0006 R 00          cmp k, 0
004E  7D 10          jge positive_k
0050          negative_k:
0050  8B 1E 000A R          mov bx, i2
0054  83 FB 00          cmp bx, 0
0057  7D 02          jge positive_i2
0059  F7 DB          neg bx
005B          positive_i2:
005B  03 C3          add ax, bx
005D  EB 09 90          jmp function3_end
0060          positive_k:
0060  3D 0006          cmp ax, 6
0063  7D 03          jge function3_end
0065  B8 0006          mov ax, 6
0068          function3_end:
0068  A3 000C R          mov result, ax
006B  CB          ret
006C          Main      ENDP
006C          CODE      ENDS

```

```

END Main
#Microsoft (R) Macro Assembler Version 5.10
11:44:16

```

11/7/22

Symbols-1

Segments and Groups:

	N a m e	Length	Align	Combine Class
ASTACK	0018	PARA	STACK
CODE	006C	PARA	NONE
DATA	000E	PARA	NONE

Symbols:

	N a m e	Type	Value	Attr
A	L WORD	0000	DATA
B	L WORD	0002	DATA
FUNCTION12	L NEAR	0007	CODE

FUNCTION12_END	L NEAR	0037	CODE	
FUNCTION3	L NEAR	0049	CODE	
FUNCTION3_END	L NEAR	0068	CODE	
GREATER_A	L NEAR	0029	CODE	
GREATER_B	L NEAR	001C	CODE	
I	L WORD	0004	DATA	
I1	L WORD	0008	DATA	
I2	L WORD	000A	DATA	
K	L WORD	0006	DATA	
MAIN	F PROC	0000	CODE	Length =
006C				
NEGATIVE_K	L NEAR	0050	CODE	
POSITIVE_I2	L NEAR	005B	CODE	
POSITIVE_K	L NEAR	0060	CODE	
RESULT	L WORD	000C	DATA	
@CPU	TEXT	0101h		
@FILENAME	TEXT	lab3		
@VERSION	TEXT	510		
#Microsoft (R) Macro Assembler Version 5.10				11/7/22
11:44:16				

Symbols-2

71 Source Lines
71 Total Lines
25 Symbols

47998 + 461309 Bytes symbol space free

0 Warning Errors
0 Severe Errors