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

ВАРИАНТ 22

Студент гр. 1383 _____ Харитонов Н.М.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Изучить организацию ветвящихся процессов и разработать программу на языке Ассемблер.

Задание.

Разработать на языке Ассемблера программу, которая по заданным целочисленным значениям параметров a, b, i, k вычисляет: a) значения функций i1 = 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 маршруты выполнения программы, а также различные знаки параметров a и b.

Выполнение работы.

Были произведены следующие упрощения:

1. -(6*i+8) = -(6*i-4)-12

2. 3*(i+2) = 3*i+6

3. 9-3*(i-1) = 9-(3*i+6-9)

1) Это необходимо для ускорения подсчета второй функции, т.к. не надо еще раз получать “6*i”

2) Данная вещь сделана, чтобы можно было добавлять “i” из памяти, не прибавляя к ней “2” каждый раз, а сразу в конце прибавив “6”.

3) Использование полученного значения для ускорения подсчета второй функции, аналогично пункту “1)”.

Таблица 1. Протокол lr3

№	Исходные данные	Вывод программы
1	a=6, b=4, i=3, k=0	$FFF2_{16} = -14_{10}$ $FFE6_{16} = -26_{10}$ $0028_{16} = 40_{10}$
2	a=4, b=6, i=3, k=0	$000F_{16} = 15_{10}$ $0003_{16} = 3_{10}$ $0012_{16} = 18_{10}$
3	a=6, b=4, i=-1, k = 0	$000A_{16} = 10_{10}$ $FFFE_{16} = -2_{10}$ $0008_{16} = 8_{10}$
4	a=4, b=6, i=-1, k=0	$0003_{16} = 3_{10}$ $000F_{16} = 15_{10}$ $0012_{16} = 18_{10}$
5	a=6, b=4, i=3, k=3	$FFF2_{16} = -14_{10}$ $FFE6_{16} = -26_{10}$ $FFE6_{16} = -26_{10}$

Выводы.

В ходе выполнения работы была изучена организация ветвящихся процессов и разработана эффективная программа на языке Ассемблера.

ПРИЛОЖЕНИЕ А
КОД ПРОГРАММЫ

lr3.asm:

```
AStack SEGMENT STACK
    dw 16 DUP(?)
AStack ENDS

DATA SEGMENT
    a dw 6
    b dw 4
    i dw 3
    k dw 3
    func1 dw 0
    func2 dw 0
    func3 dw 0
DATA ENDS

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

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 vet1 ; a > b
    shl cx, 1
    add cx, i
    add cx, 6
    mov func1, cx
    sub cx, 9
    mov dx, 9
    sub dx, cx
    mov func2, dx
    jmp cont1
vet1:
    shl cx, 1
    add cx, i
    shl cx, 1
    sub cx, 4
    neg cx
    mov func1, cx
    sub cx, 12
    mov func2, cx
cont1:
    mov ax, func1
    mov bx, func2
    mov cx, k
    cmp cx, 0
    je vet2
    cmp ax, bx
    jl lower
    mov ax, bx
    jmp finish
```

```
lower:
    jmp finish
vet2:
    add ax, bx
    cmp ax, 0
    jle negative
    jmp finish
negative:
    neg ax
finish:
    mov func3, ax
    mov ax, func1
    mov bx, func2
    mov cx, func3
    ret
Main ENDP
CODE ENDS
END Main
```

ПРИЛОЖЕНИЕ В
ЛИСТИНГ

lr3.lst:

```
Microsoft (R) Macro Assembler Version 5.10
11/6/22 18:49:37

Page     1-1


0000                    AStack SEGMENT STACK
0000  0010[                 dw 16 DUP(?)
      ????
                    ]

0020                    AStack ENDS

0000                    DATA SEGMENT
0000  0006                  a dw 6
0002  0004                  b dw 4
0004  0003                  i dw 3
0006  0003                  k dw 3
0008  0000                  func1 dw 0
000A  0000                  func2 dw 0
000C  0000                  func3 dw 0
000E                    DATA ENDS

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

```
0000                Main PROC FAR
0000  1E                push ds
0001  2B C0              sub ax, ax
0003  50                push ax
0004  B8 ---- R          mov ax, DATA
0007  8E D8              mov ds, ax
                  ;-------------------------
0009  A1 0000 R          mov ax, a
000C  8B 0E 0004 R           mov cx, i
0010  3B 06 0002 R           cmp ax, b
0014  7F 1C              jg vet1 ; a > b
0016  D1 E1              shl cx, 1
0018  03 0E 0004 R           add cx, i
001C  83 C1 06               add cx, 6
001F  89 0E 0008 R           mov func1, cx
0023  83 E9 09               sub cx, 9
0026  BA 0009                mov dx, 9
0029  2B D1              sub dx, cx
002B  89 16 000A R           mov func2, dx
002F  EB 19 90               jmp cont1
0032                vet1:
0032  D1 E1                  shl cx, 1
0034  03 0E 0004 R           add cx, i
0038  D1 E1                  shl cx, 1
003A  83 E9 04               sub cx, 4
003D  F7 D9                  neg cx
003F  89 0E 0008 R           mov func1, cx
0043  83 E9 0C               sub cx, 12
0046  89 0E 000A R           mov func2, cx
```

```
004A                cont1:
004A  A1 0008 R         mov ax, func1
004D  8B 1E 000A R          mov bx, func2
0051  8B 0E 0006 R          mov cx, k
0055  83 F9 00              cmp cx, 0
```



```
0058  74 0C             je vet2
005A  3B C3             cmp ax, bx
005C  7C 05             jl lower
005E  8B C3             mov ax, bx
0060  EB 10 90              jmp finish
0063                lower:
0063  EB 0D 90              jmp finish
0066                vet2:
0066  03 C3             add ax, bx
0068  3D 0000               cmp ax, 0
006B  7E 03             jle negative
006D  EB 03 90              jmp finish
0070                negative:
0070  F7 D8             neg ax
0072                finish:
0072  A3 000C R         mov func3, ax
0075  A1 0008 R         mov ax, func1
0078  8B 1E 000A R          mov bx, func2
007C  8B 0E 000C R          mov cx, func3
```

```
0080  CB                ret
0081                Main ENDP
0081                CODE ENDS
                  END Main
```

Microsoft (R) Macro Assembler Version 5.10 11/6/22 18:49:37

Symbols-1

Segments and Groups:

N a m e	Length	Align	Combine Class
ASTACK	0020	PARA	STACK
CODE	0081	PARA	NONE
DATA	000E	PARA	NONE

Symbols:

N a m e	Type	Value	Attr
A	L WORD	0000	DATA
B	L WORD	0002	DATA
CONT1	L NEAR	004A	CODE

```
FINISH . . . . . .      L NEAR   0072
CODE
FUNC1  . . . . . .      L WORD   0008
DATA
FUNC2  . . . . . .      L WORD   000A
DATA
FUNC3  . . . . . .      L WORD   000C
DATA

I  . . . . . .      L WORD   0004
DATA

K  . . . . . .      L WORD   0006
DATA

LOWER  . . . . . .      L NEAR   0063
CODE

MAIN . . . . . .      F PROC   0000
CODELength = 0081

NEGATIVE . . . . . .      L NEAR   0070
CODE

VET1 . . . . . .      L NEAR   0032
CODE
VET2 . . . . . .      L NEAR   0066
CODE
```

```
@CPU . . . . . .      TEXT  0101h

@FILENAME  . . . . . .      TEXT  lr3

@VERSION . . . . . .      TEXT  510



     74 Source  Lines
     74 Total   Lines
     22 Symbols

  48070 + 461237 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```