­­­­­НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт до комп’ютернего практикуму з дисципліни

«Системне програмне забезпечення»

Прийняв

асистент кафедри ІПІ

Пархоменко А.В.

“21” травня 2023 р.

Виконав

Студент групи ІП-15

Мєшков А. І.

Київ – 2023

**Комп‘ютерний практикум № 5**

**Макрозасоби мови асемблер**

**Загальні положення**

Викладені в лекційному матеріалі.

**Завдання комп’ютерного практикуму №5**

Скласти програму на нижче наведені завдання:

1) переписати програму комп‘ютерного практикуму №2 з використанням одного макроса;

2) переписати програму комп‘ютерного практикуму №3 з використанням макросів та передачею параметрів в них;

3) переписати одну програму (на вибір студента) комп‘ютерного практикуму №4 з використанням макросів та залученням міток в тілі макросу.

**Текст програми**

Макрозасоби

**read.asm**

READ MACRO num, numstr, fl, len, error\_mes, errorFl

local no\_minus, cycle, error1,finish

;read number

MOV num,0

LEA dx,numstr

MOV ah,10

int 21h

MOV al,10

int 29h

MOV ax,0

MOV al, [numstr+1]

MOV len,ax

MOV si,2

MOV al,numstr[si]

;translate to print

CMP al,2Dh

jne no\_minus

inc si

MOV fl,1

dec len

no\_minus:

MOV cx, len

MOV bx,10

cycle:

MOV ax,num

IMUL bx

jo error1

MOV num,ax

MOV ax,0

MOV al,numstr[si]

SUB al,30h

CMP al,0

jl error1

CMP al,9

ja error1

ADD num,ax

CMP num,32768

ja error1

inc si

loop cycle

CMP fl,1

jne finish

CMP num,32699

ja error1

NEG num

jmp finish

error1:

ERROR error\_mes, errorFl

finish:

ENDM

**write.asm**

WRITE MACRO num

local m1,m2,m3

MOV bx,num

OR bx,bx

jns m1

MOV al,'-'

int 29h

neg bx

m1:

MOV ax,bx

XOR cx,cx

MOV bx,10

m2:

XOR dx,dx

DIV bx

ADD dl,'0'

PUSH dx

inc cx

TEST ax,ax

jnz m2

m3:

POP ax

int 29h

loop m3

ENDM

**error.asm**

ERROR MACRO error\_mes,errorFl

MOV dx,offset error\_mes

MOV ah,9

int 21h

MOV al,10

int 29h

ENDM

**ost.asm**

WRITE\_OST MACRO point\_mes, drib, ost, dop

local ost\_is,ost\_write,cycle\_final,ost\_end

CMP ost,0

jne ost\_is

jmp ost\_end

ost\_is:

MOV dx,offset point\_mes

MOV ah,9

int 21h

MOV cx,4

ost\_write:

MOV bx,10

MOV ax,drib

MUL bx

MOV drib,ax

MOV ax,ost

MOV bx,10

MUL bx

DIV dop

MOV ost,dx

ADD drib,ax

CMP ost,0

je cycle\_final

loop ost\_write

cycle\_final:

WRITE drib

ost\_end:

ENDM

**f1.asm**

F1 MACRO x, y, dop, ost, point\_mes, drib

MOV ax,x

MOV bx,x

MUL bx

MUL bx

MOV dop,ax

DIV y

MOV bx,y

MOV dop,bx

MOV x,ax

MOV ost,dx

WRITE x

WRITE\_OST point\_mes, drib, ost, dop

ENDM

**f2.asm**

F2 MACRO x,y,dop,ost,point\_mes,drib

NEG y

MOV ax,x

MOV bx,x

MOV dop,ax

MOV ax,y

MOV bx,2

MUL bx

MOV y,ax

MOV ax,dop

DIV y

MOV bx,y

MOV dop,bx

MOV [x],ax

MOV ost,dx

NEG x

WRITE x

WRITE\_OST point\_mes,drib,ost,dop

ENDM

**f3.asm**

F3 MACRO x,dop,ost,point\_mes,drib

MOV ax,x

MOV bx,x

MUL bx

MOV bx,3

MUL bx

MOV dop,ax

MOV bx,x

MOV dop,bx

MOV [x],ax

MOV ost,dx

WRITE x

WRITE\_OST point\_mes,drib,ost,dop

ENDM

**f4.asm**

F4 MACRO x

MOV x, 1

WRITE x

ENDM

**rd.asm**

READ\_ODN\_MAS MACRO countOdn\_mes, errorFl, readNumber, countOdn, odnMas, elemMas\_mes, arrow, memoryCx, i, numstr, fl,len,error\_mes, writeNumber

local point2, readMas, point3, toError2,point4,point5,point6,point7,point3not\_er,not\_er,toError0,point8

point2:

MOV dx,offset countOdn\_mes

MOV ah,9

int 21h

MOV errorFl,0

READ readNumber, numstr,fl,len,error\_mes,errorFl

CMP errorFl,1

jne point4

jmp point2

point4:

CMP readNumber,1

jge point5

jmp toError2

point5:

CMP readNumber,100

jle point6

jmp toError2

point6:

MOV ax,readNumber

MOV countOdn,ax

MOV cx,countOdn

MOV i,1

MOV si,0

LEA di,odnMas

readMas:

jmp point3not\_er

point3:

inc cx

point3not\_er:

MOV dx,offset elemMas\_mes

MOV ah,9

int 21h

MOV ax,i

MOV writeNumber,ax

MOV memoryCx,cx

WRITE writeNumber

MOV dx,offset arrow

MOV ah,9

int 21h

MOV errorFl,0

READ readNumber,numstr,fl,len,error\_mes,errorFl

CMP errorFl,1

jne not\_er

jmp toError0

not\_er:

MOV cx,memoryCx

MOV ax,readNumber

MOV [di],ax

ADD i,1

ADD di,2

dec cx

CMP cx,0

je point7

jmp readMas

point7:

jmp point8

toError2:

ERROR error\_mes,errorFl

jmp point2

toError0:

jmp point3

point8:

ENDM

**wd.asm**

WRITE\_ODN\_MAS MACRO countOdn, writeNumber, memoryCx, space

local write\_mas

MOV cx,countOdn

MOV si,0

write\_mas:

MOV dx,odnMas[si]

MOV writeNumber,dx

MOV memoryCx,cx

WRITE writeNumber

MOV cx, memoryCx

MOV dx,offset space

MOV ah,9

int 21h

ADD si,2

loop write\_mas

ENDM

**Комп’ютерний практикум 2:**

STREG SEGMENT PARA STACK "STACK"

DW 64 DUP ('?')

STREG ENDS

INCLUDE read.asm

INCLUDE write.asm

INCLUDE error.asm

DSEG SEGMENT PARA PUBLIC "DATA"

len dw 0

messtr db "Enter number -> $"

numstr db 7,?,7 dup('?')

num dw 0

fl db 0

error\_mes db "Error$"

errorFl dw 0

DSEG ENDS

CSEG SEGMENT PARA PUBLIC "CODE"

MAIN PROC FAR

ASSUME ds:DSEG, cs:CSEG, ss:STREG

PUSH ds

MOV ax, 0

PUSH ax

MOV ax,DSEG

MOV ds, ax

;print message

MOV dx,offset messtr

MOV ah, 9

int 21h

READ num,numstr,fl,len,error\_mes,errorFl

CMP num, -32668

jl final

CMP errorFl,1

je final

SUB num,8

WRITE num

final:

RET

MAIN ENDP

CSEG ENDS

END MAIN

**Комп’ютерний практикум 3:**

STREG SEGMENT PARA STACK "STACK"

dw 128 DUP (' ?' )

STREG ENDS

INCLUDE read.asm

INCLUDE write.asm

INCLUDE error.asm

INCLUDE f1.asm

INCLUDE f2.asm

INCLUDE f3.asm

INCLUDE f4.asm

INCLUDE ost.asm

DSEG SEGMENT PARA PUBLIC "DATA"

len dw 0

messtr\_x db "Enter number x -> $"

numstr\_x db 7,?,7 dup('?')

messtr\_y db "Enter number y -> $"

numstr\_y db 7,?,7 dup('?')

x dw 0

y dw 0

dop dw 0

ost dw 0

drib dw 0

fl db 0

error\_mes db "Error$"

point\_mes db ".$"

m db "End$"

errorFl dw 0

DSEG ENDS

CSEG SEGMENT PARA PUBLIC "CODE"

MAIN PROC FAR

ASSUME ds:DSEG, cs:CSEG, ss:STREG

PUSH ds

MOV ax,0

PUSH ax

MOV ax,DSEG

MOV ds,ax

MOV dx, offset messtr\_x

MOV ah,9

int 21h

READ x,numstr\_x,fl,len,error\_mes,errorFl

PUSH ds

MOV ax,0

PUSH ax

MOV ax,DSEG

MOV ds,ax

MOV y,0

MOV fl,0

MOV len,0

MOV errorFl, 0

MOV dx, offset messtr\_y

MOV ah,9

int 21h

READ y,numstr\_y,fl,len,error\_mes,errorFl

CMP errorFl,1

jne p0

RET

p0:

CMP x,40

jle p1

jmp main\_final

p1:

CMP x,0

jge p2

jmp fourth

p2:

CMP y,0

jne p3

jmp third

p3:

CMP y,0

jg p4

jmp second

p4:

F1 x, y, dop, ost, point\_mes, drib

jmp main\_final

fourth:

F4 x

jmp main\_final

third:

F3 x, dop, ost, point\_mes, drib

jmp main\_final

second:

F2 x, y, dop, ost, point\_mes, drib

jmp main\_final

main\_final:

MOV ah,4Ch

int 21h

RET

MAIN ENDP

CSEG ENDS

END MAIN

**Комп’ютерний практикум 4:**

STSEG SEGMENT PARA STACK "STACK"

dw 64 DUP ( '?' )

STSEG ENDS

INCLUDE rd.asm

INCLUDE read.asm

INCLUDE write.asm

INCLUDE error.asm

INCLUDE wd.asm

DSEG SEGMENT PARA PUBLIC "DATA"

readNumber dw 0

fl dw 0

errorFl dw 0

len dw 0

numstr db 7,?,7 dup('?')

error\_mes db "Error$"

countOdn dw 0

countOdn\_mes db "Enter count -> $"

odnMas dw 100 dup (0)

elemMas\_mes db "Enter element number $"

arrow db " -> $"

space db " $"

i dw 0

writeNumber dw 0

memoryCx dw 0

sum dw 0

sum\_mes db "Sum: $"

DSEG ENDS

CSEG SEGMENT PARA PUBLIC "CODE"

MAIN PROC FAR

ASSUME ds:DSEG, cs:CSEG, SS:STSEG

PUSH ds

MOV ax, 0

PUSH ax

MOV ax,DSEG

MOV ds,ax

READ\_ODN\_MAS countOdn\_mes, errorFl, readNumber, countOdn, odnMas, elemMas\_mes, arrow, memoryCx, i, numstr, fl,len,error\_mes, writeNumber

WRITE\_ODN\_MAS countOdn, writeNumber, memoryCx, space

MOV al,10

int 29h

MOV cx,countOdn

MOV sum,0

MOV si,0

summary:

MOV ax,odnMas[si]

ADD sum,ax

jo toError3

ADD si,2

loop summary

MOV dx,offset sum\_mes

MOV ah,9

int 21h

MOV ax,sum

MOV writeNumber,ax

WRITE writeNumber

RET

toError3:

ERROR error\_mes, errorFl

RET

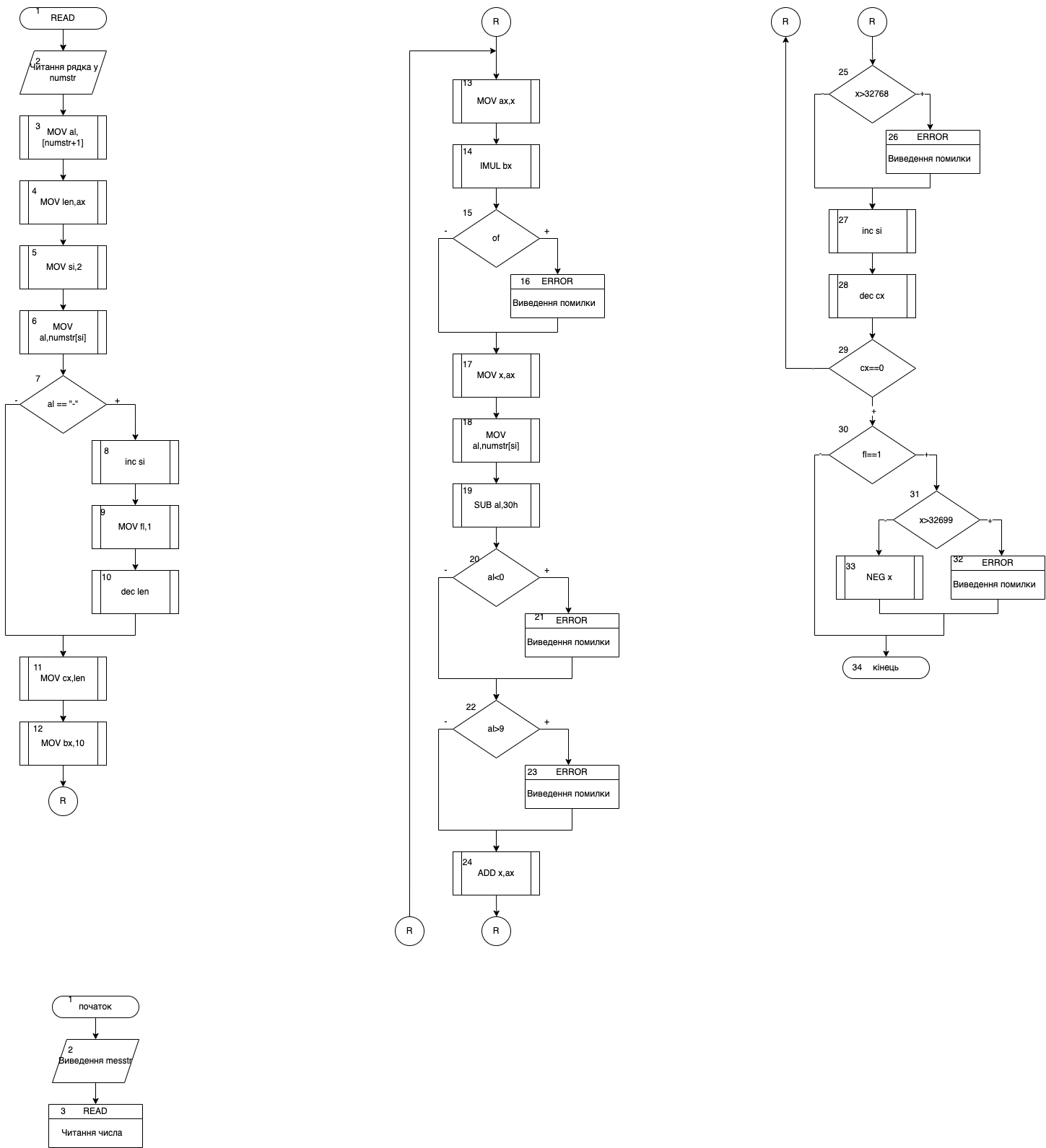
MAIN ENDP

CSEG ENDS

END MAIN

Схема функціонування програми

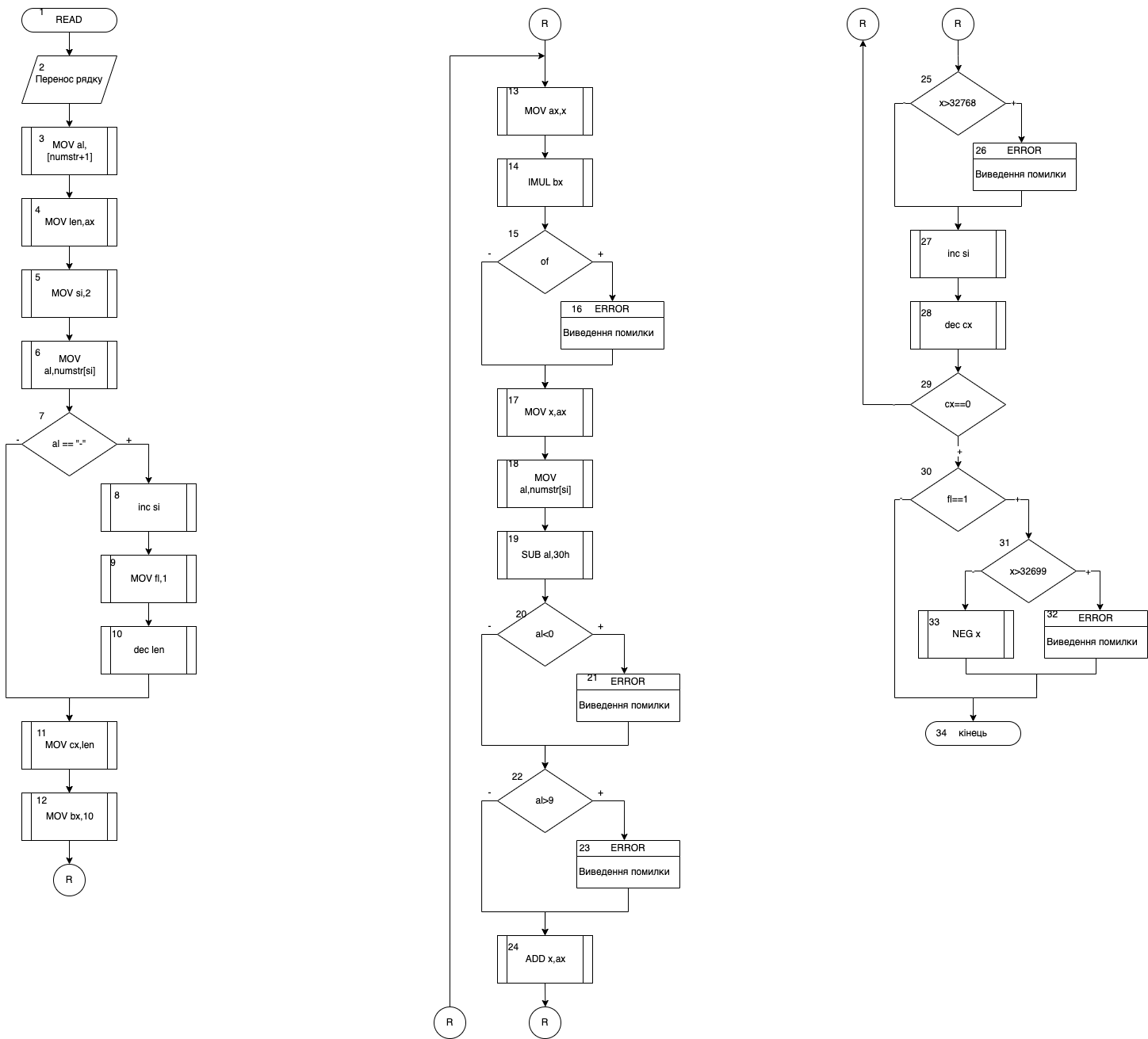
**read.asm**



R2

R1

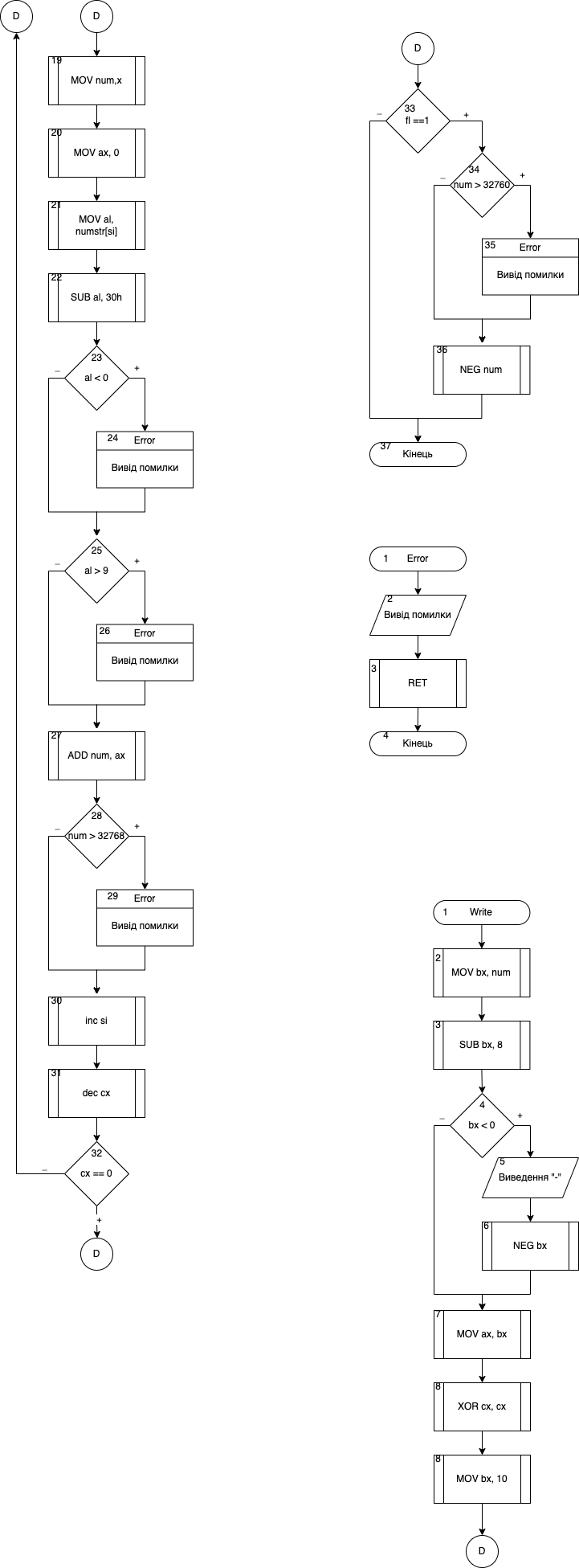
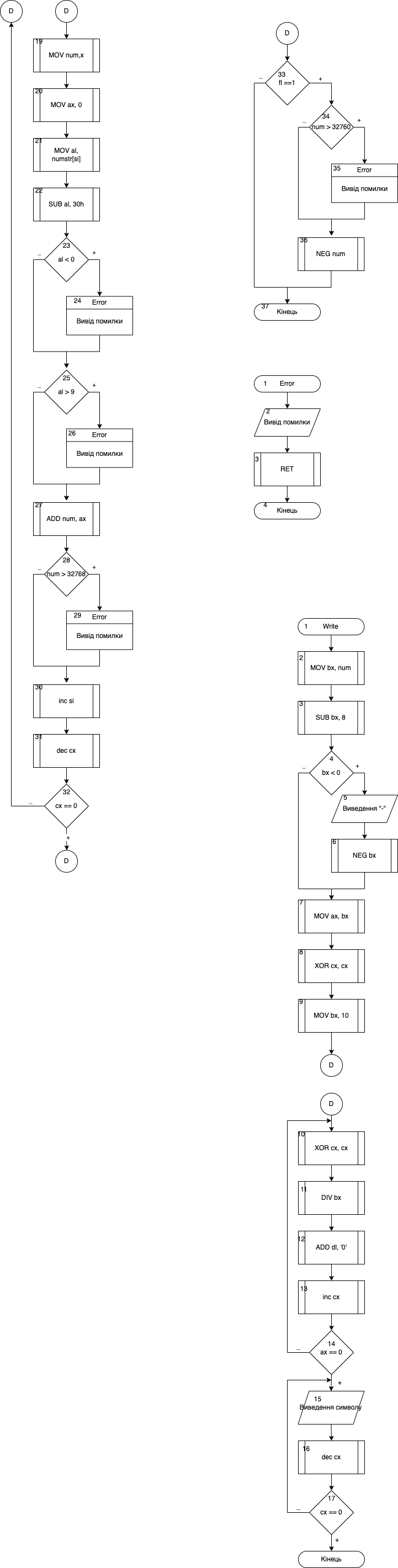
Рисунок 5.1 Схема функціонування макрозасобу читання числа

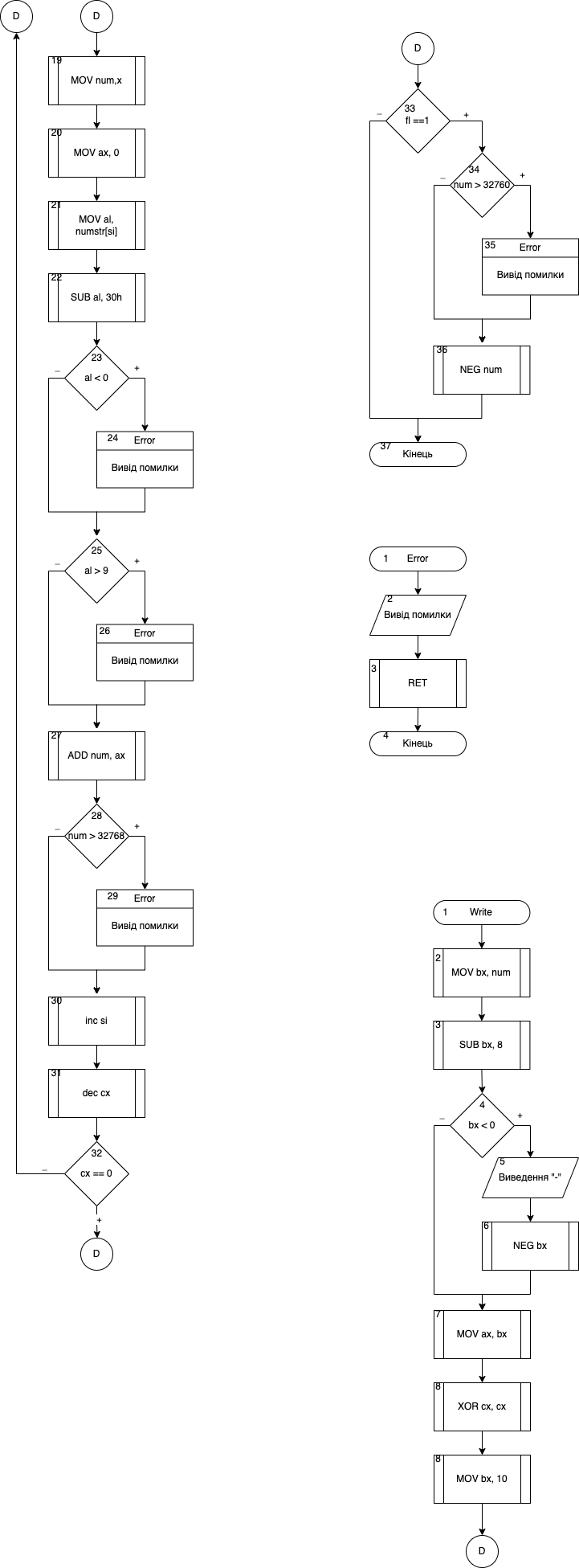


R2

R1

Продовження рис.5.1

**write.asm**

****

|  |
| --- |
|  |
| Рисунок 5.2 Схема функціонування макрозасобу виведення числа |
|  |

**error.asm**

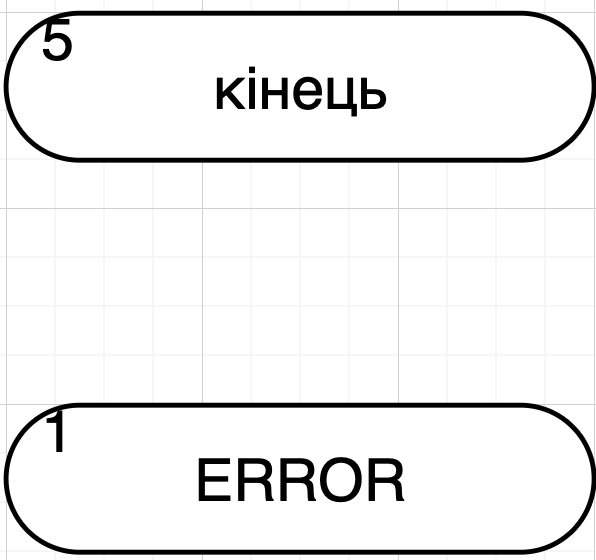
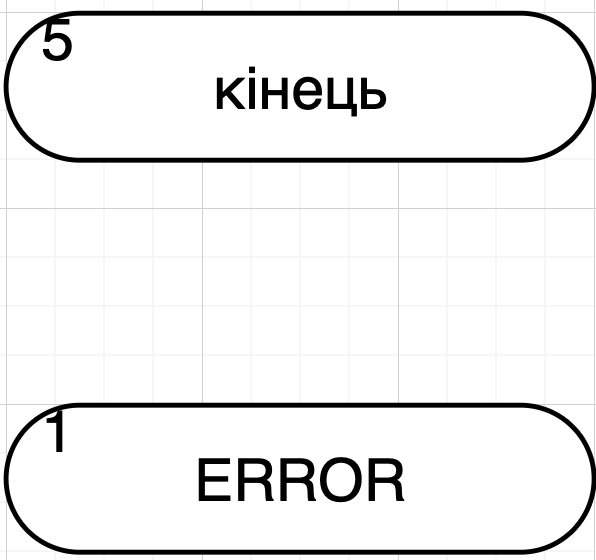
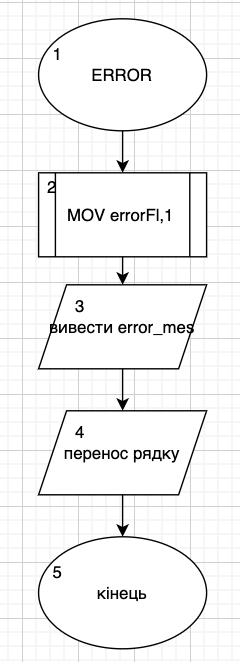


Рисунок 5.3 Схема функціонування макрозасобу виведення помилки

**ost.asm**

O1

O

3

2

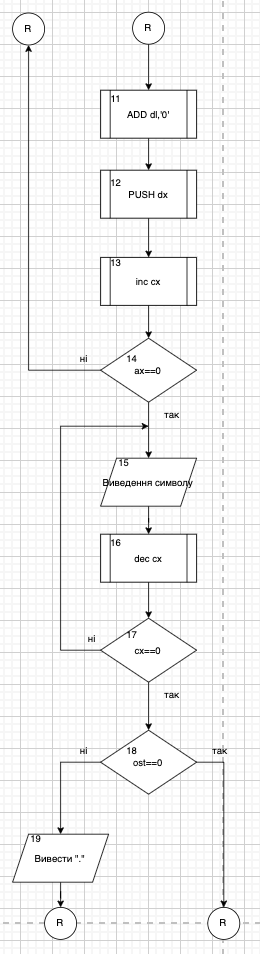
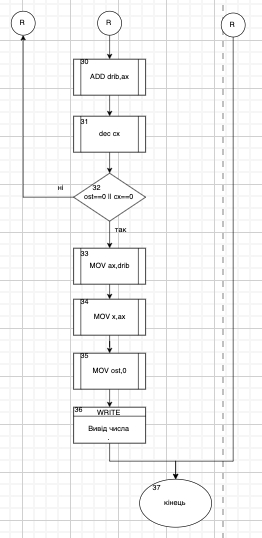
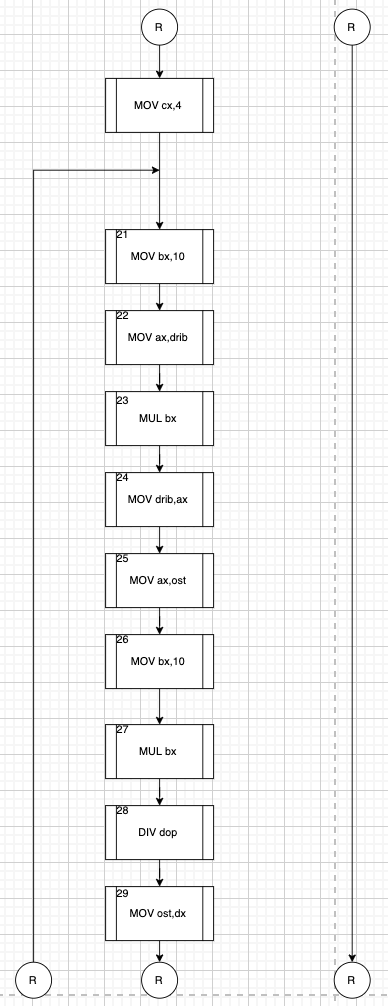


Рисунок 5.4 Схема функціонування макрозасобу виведення остачі

****

O4

O4

O3

O3

O2

O2

O1

O

16

20

19

18

17

15

14

13

12

11

10

9

8

7

7

6

5

4

Продовження рис.5.4

**f1.asm**

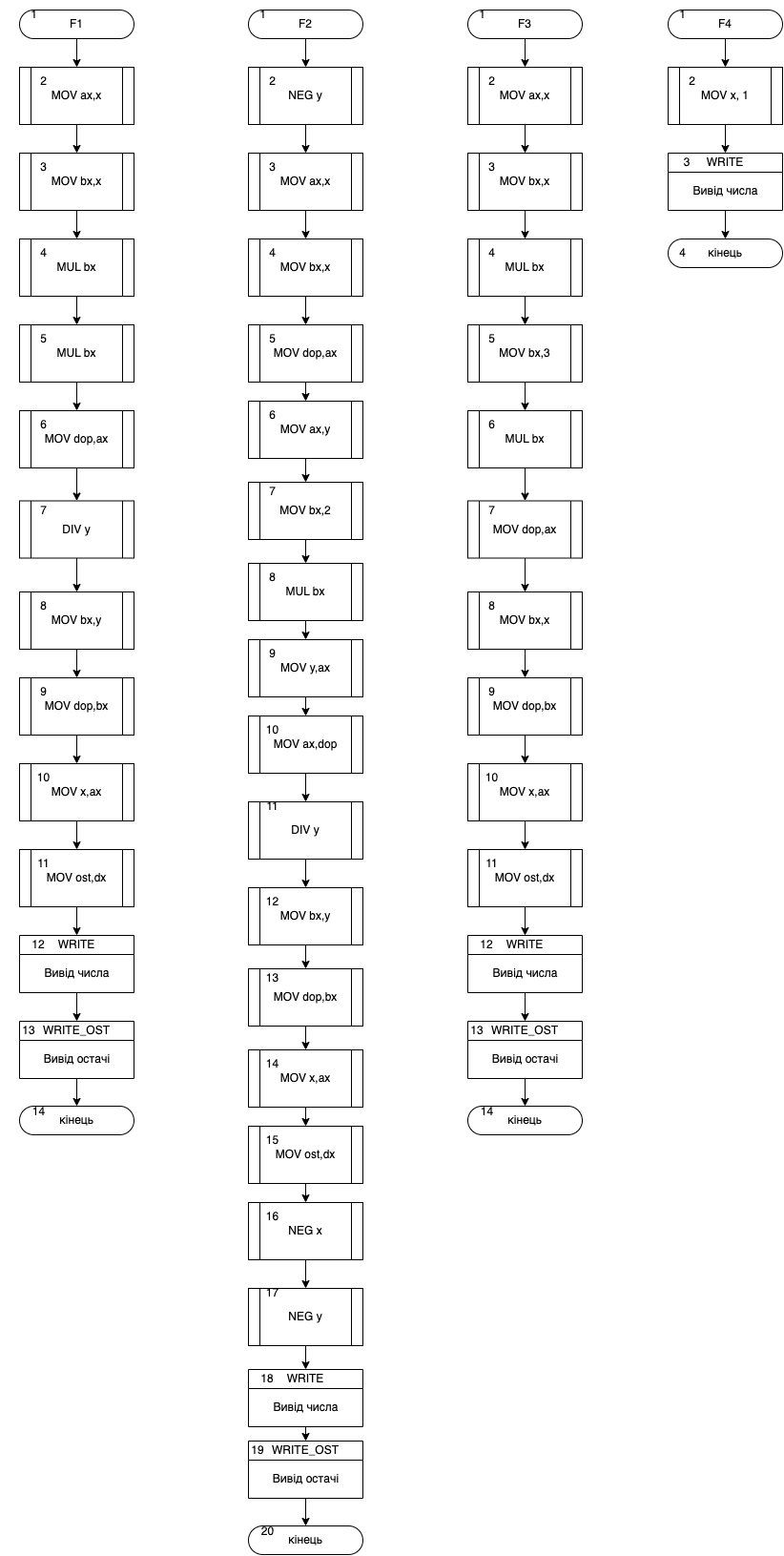
****

Рисунок 5.5 Схема функціонування макрозасобу 1ї функції

**f2.asm**

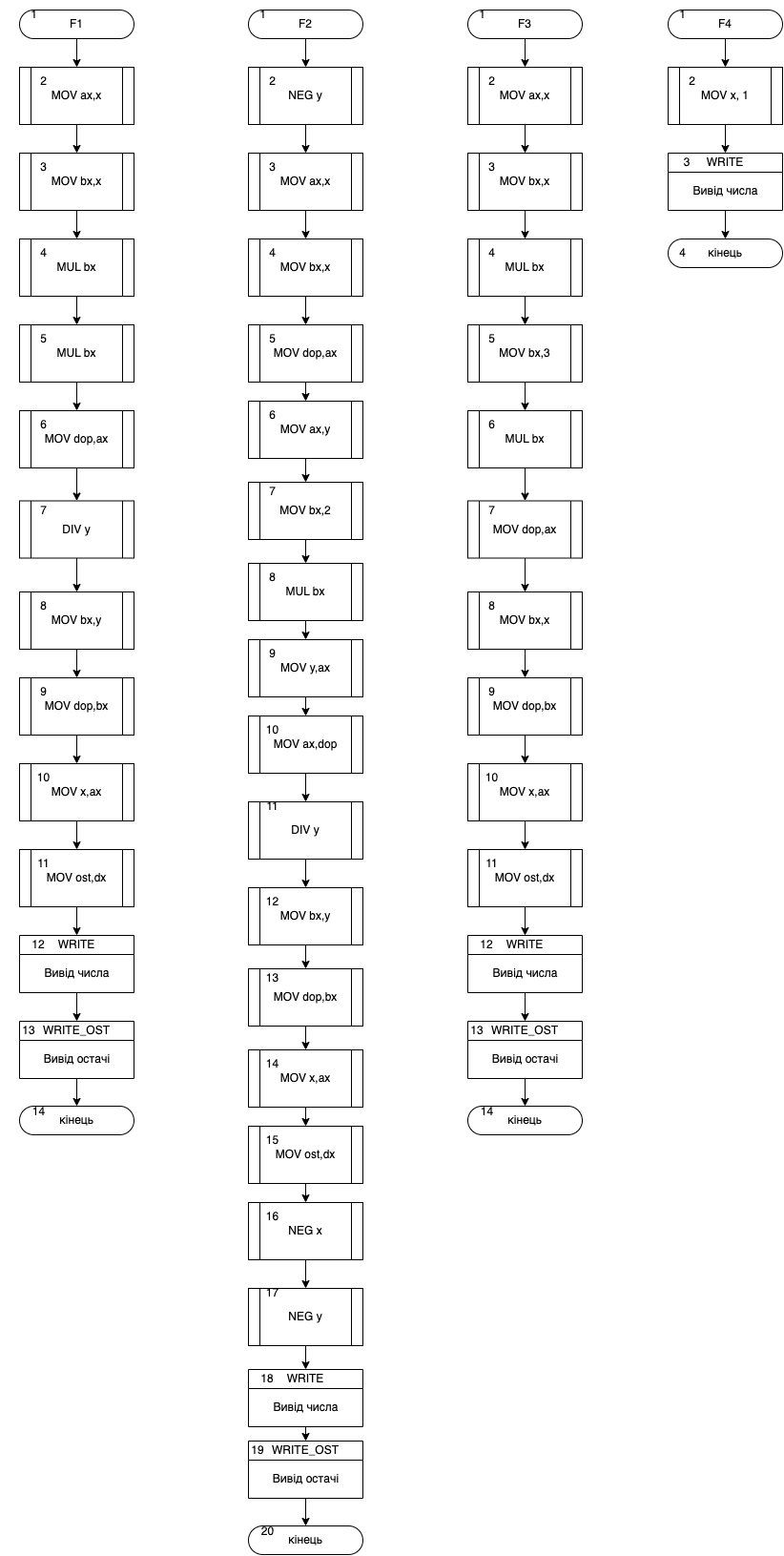
****

Рисунок 5.6 Схема функціонування макрозасобу 2ї функції

**f3.asm**

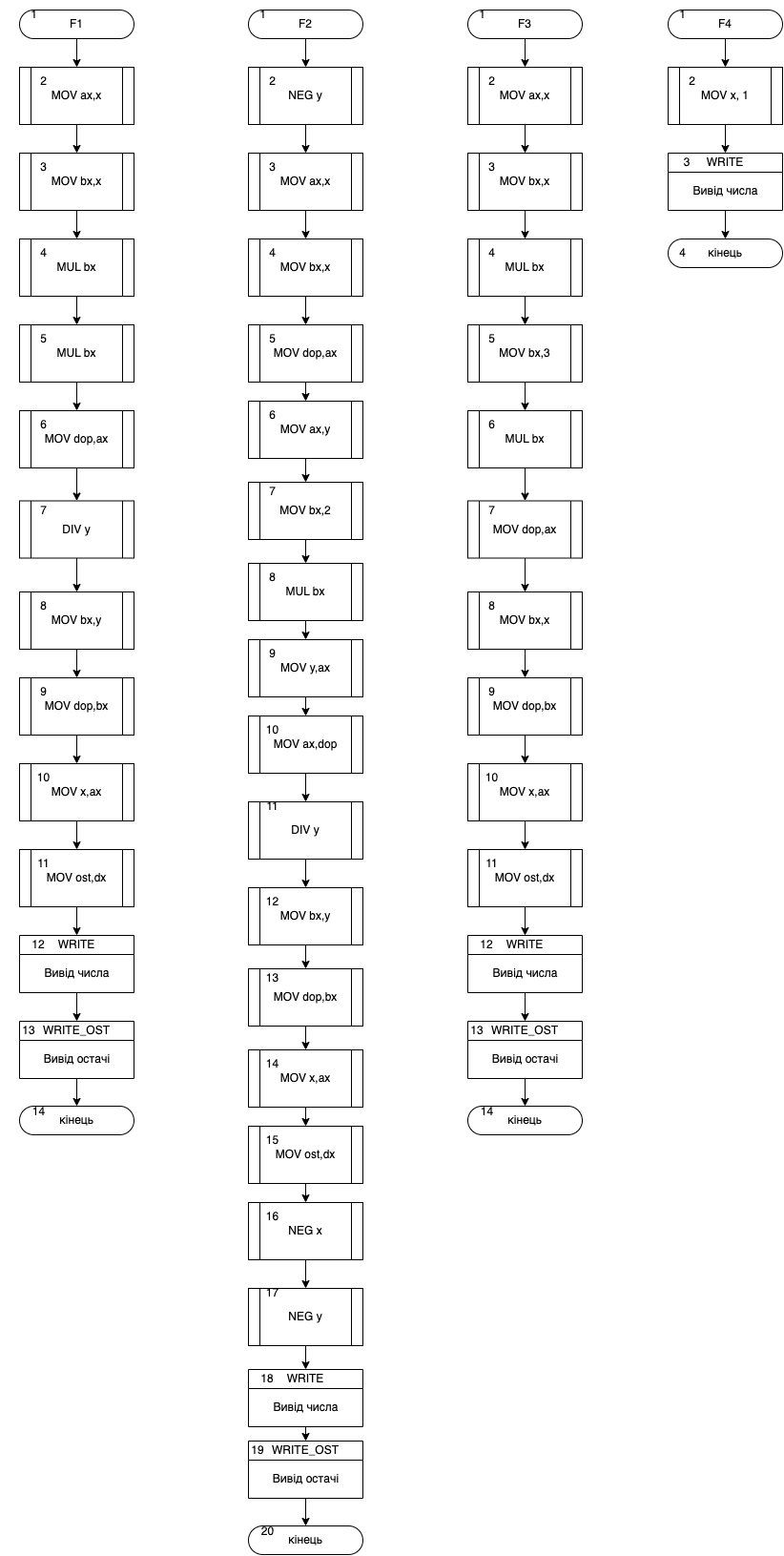
****

Рисунок 5.7 Схема функціонування макрозасобу 3ї функції

**f4.asm**

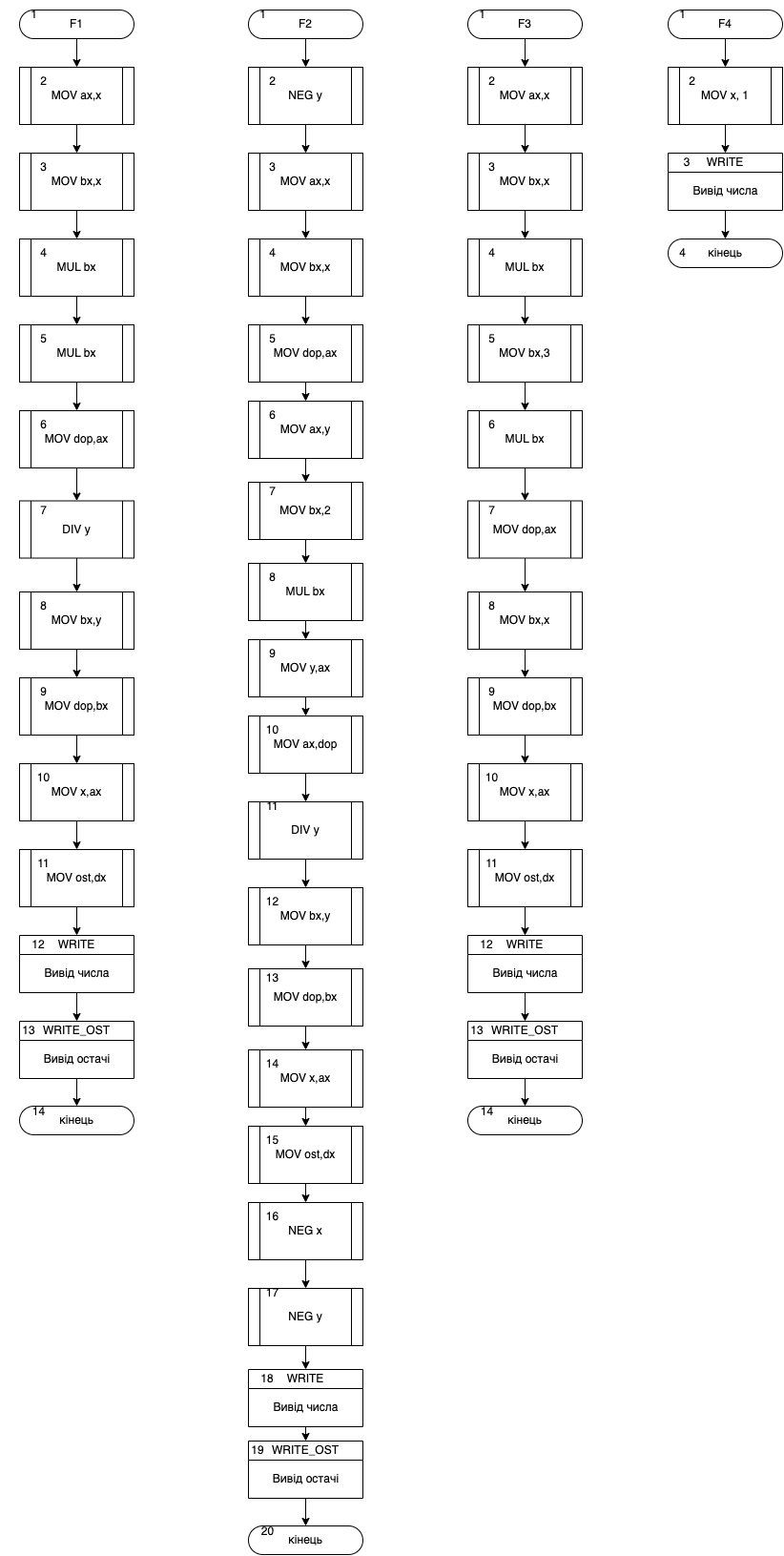
****

Рисунок 5.8 Схема функціонування макрозасобу 4ї функції

**rd.asm**

|  |
| --- |
| V |
| Рисунок 5.9 Схема функціонування макрозасобу читання одновимірного масиву |

|  |
| --- |
| V1  V |
| Продовження рисунку 5.9 |
| V1 |
| Продовження рисунку 5.9 |

**wd.asm**

|  |
| --- |
|  |
| Рисунок 5.10 Схема функціонування макрозасобу виведення одномірного масиву |

**Комп’ютерний практикум 2:**

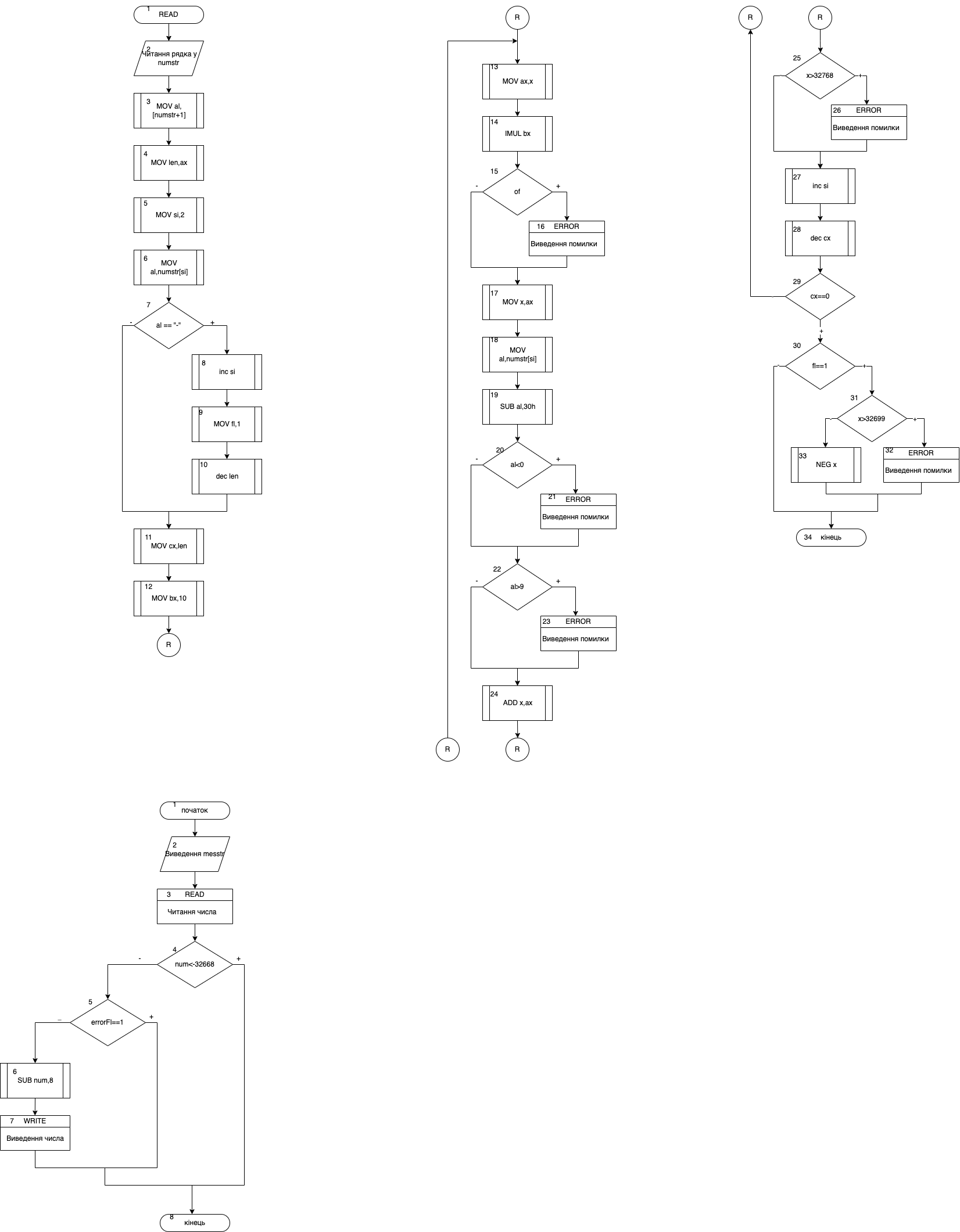
****

Рисунок 5.11 Схема функціонування головної програми КП№2

**Комп’ютерний практикум 3:**

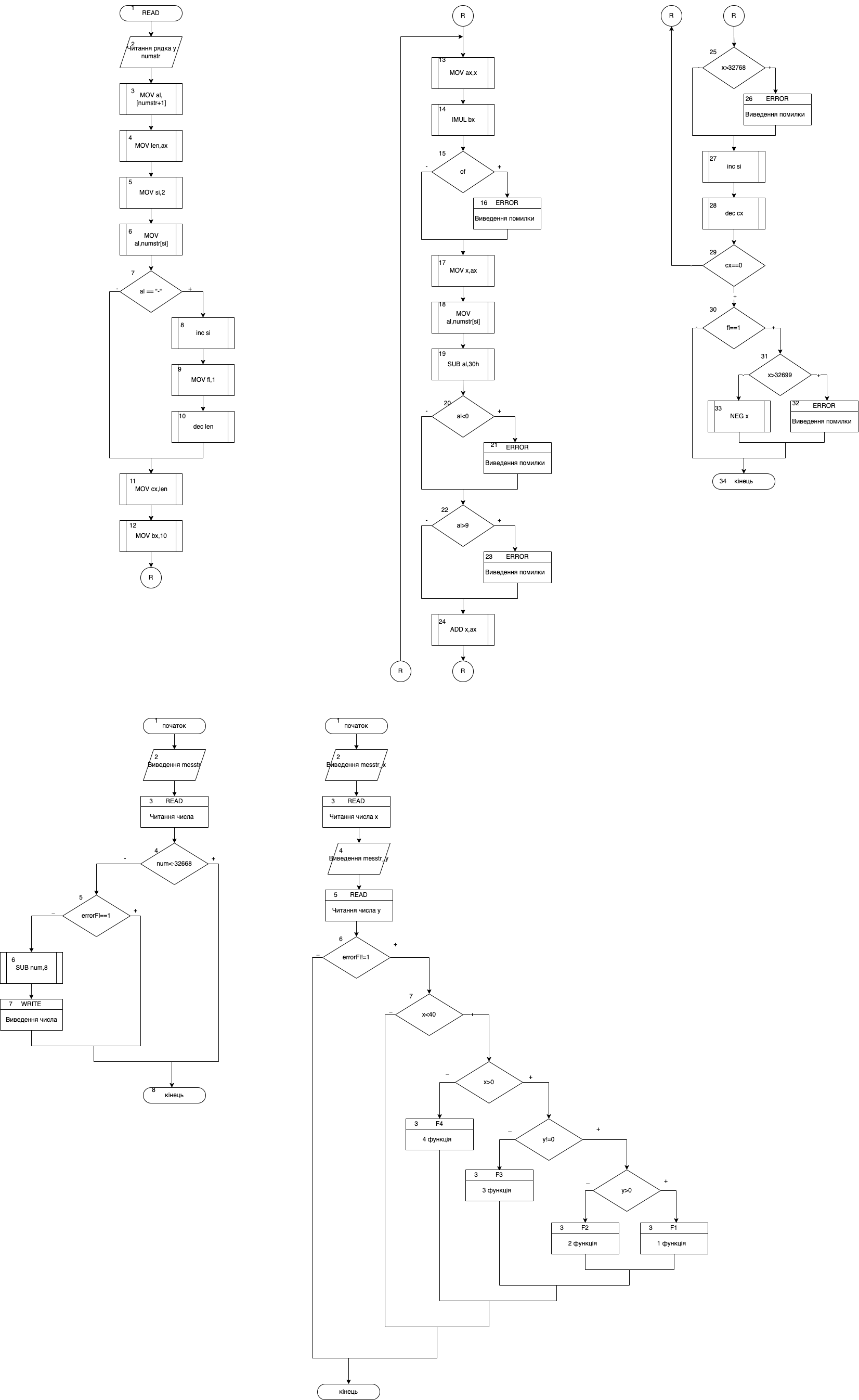
****

Рисунок 5.12 Схема функціонування головної програми КП№3

**Комп’ютерний практикум 4:**

|  |
| --- |
|  |
| Рисунок 5.13 Схема функціонування головної програми КП№4 |

**Приклади виконання програми**

****

|  |
| --- |
| Рисунок 5.14 Приклад роботи програми комп’ютерного практикуму 2 |
|  |
| Рисунок 5.15 Приклад роботи програми комп’ютерного практикуму 3 |
|  |
| Рисунок 5.16 Приклад роботи програми комп’ютерного практикуму 4 |

**Висновок:** Під час виконання комп’ютерного практикуму мною було створено програми, що виконують завдання 2,3 та частково 4 комп’ютерного практикуму з використанням макросів. Було протестовано програми.