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 № 3

Програмування розгалужених алгоритмів

Загальні положення

Викладені в лекційному матеріалі.

Завдання комп’ютерного практикуму №3

Написати програму, яка повинна мати наступний функціонал:

1. Можливість введення користувачем значень x, y, t, a, b за необхідності.
2. Обчислювати значення функції за введеними значеннями.
3. Виводити на екран результат обчислень.
4. Якщо є ділення, то результат дозволяється виводити:

а) як дійсне число (наприклад: $\frac{5}{3}$ = 1,666667) – підвищена складність;

б) окремо цілу частину та остачу (наприклад: $\frac{5}{3}$ = 1 остача 2) – середня складність;

в) окремо цілу частину та остачу як дріб (наприклад: $\frac{5}{3} = 1\frac{2}{3}$) – середня складність.

5. Програма повинна мати захист від некоректного введення вхідних даних (символи, переповнення, ділення на 0 і т.і.)

Номер завдання вибирати за останніми двома числами номеру в заліковій книжці.

17.
$$Z=\begin{cases} x^3 / y & \text{якщо } x > 0, y > 0 \\ x / 2y & \text{якщо } x > 0, y < 0 \\ 3x^2 & \text{якщо } y = 0 \\ 1 & \text{в інших випадках} \end{cases}$$

Текст програми

```
STREG SEGMENT PARA STACK "STACK"
  dw 64 DUP (' ?' )
STREG ENDS

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 ".$"
DSEG ENDS

CSEG SEGMENT PARA PUBLIC "CODE"
ASSUME ds:DSEG, cs:CSEG, ss:STREG

READ PROC FAR

start:
  PUSH ds
  MOV ax,0
  PUSH ax
  MOV ax,DSEG
  MOV ds,ax

  MOV x,0
  MOV y,0
  MOV fl,0
  MOV len,0

  ; print message for x
  MOV dx, offset messtr_x
  MOV ah,9
  int 21h

  ; read number x
  LEA dx, numstr_x
  MOV ah,10
  int 21h
```

```
  MOV al,10
  int 29h

  MOV ax,0
  MOV al,[numstr_x+1]
  MOV len,ax

  MOV si,2
  MOV al,numstr_x[si]

  ; translate x to number
  CMP al,2Dh
  jne no_minus_x

  inc si
  MOV fl,1
  dec len

no_minus_x:
  MOV cx,len
  MOV bx,10

cycle_x:
  MOV ax,x
  IMUL bx
  jo error_x
  MOV x,ax
  MOV ax,0
  MOV al,numstr_x[si]
  SUB al,30h
  CMP al,0
  jl error_x

  CMP al,9
  ja error_x
  ADD x,ax
  CMP x,40
  ja error_x
  inc si
  loop cycle_x

  CMP fl,1
  jne read_y
  CMP x,40
  ja error_x
  NEG x
  jmp read_y
```

```
error_x:
  MOV dx,offset error_mes
  MOV ah,9
  int 21h
  jmp main_finish

read_y:
  PUSH ds
  MOV ax,0
  PUSH ax
  MOV ax,DSEG
  MOV ds,ax

  MOV y,0
  MOV fl,0
  MOV len,0

  ; print message for y
  MOV dx, offset messtr_y
  MOV ah,9
  int 21h

  ; read number y
  LEA dx, numstr_y
  MOV ah,10
  int 21h

  MOV al,10
  int 29h

  MOV ax,0
  MOV al,[numstr_y+1]
  MOV len,ax

  MOV si,2
  MOV al,numstr_y[si]

  ; translate y to number
  CMP al,2Dh
  jne no_minus_y

  inc si
  MOV fl,1
  dec len

no_minus_y:
  MOV cx,len
```

```
  MOV bx,10

cycle_y:
  MOV ax,y
  IMUL bx
  jo error
  MOV y,ax
  MOV ax,0
  MOV al,numstr_y[si]
  SUB al,30h
  CMP al,0
  jl error
  CMP al,9

  ja error
  ADD y,ax
  CMP y,40
  ja error
  inc si
  loop cycle_y

  CMP fl,1
  jne finish
  CMP y,40
  ja error
  NEG y
  jmp finish

finish:
  jmp ret_read

error:
  MOV dx,offset error_mes
  MOV ah,9
  int 21h
  jmp main_finish


READ ENDP

WRITE PROC FAR

  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

  CMP ost,0
  jne ost_is
  jmp main_finish

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:
  MOV ax,drib
```

```
  MOV x, ax
  MOV ost,0
  CALL WRITE
WRITE ENDP

F1 PROC FAR
  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
  CALL WRITE
F1 ENDP

F2 PROC FAR
  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
  NEG y
  CALL WRITE

F2 ENDP

F3 PROC FAR
  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
  CALL WRITE
F3 ENDP

F4 PROC FAR
  MOV x, 1
  CALL WRITE
F4 ENDP

MAIN PROC NEAR
  CALL READ
ret_read:
  CMP y,0
  je third
  CMP x,0
  jg check
  CALL F4
check:
  CMP y,0
  jg first
  CMP y,0
  jl second
first:
  CALL F1
second:
  CALL F2
third:
  CALL F3
main_finish:
  MOV ah,4Ch
  int 21h
MAIN ENDP
CSEG ENDS
END MAIN
```

Схема функціонування програми

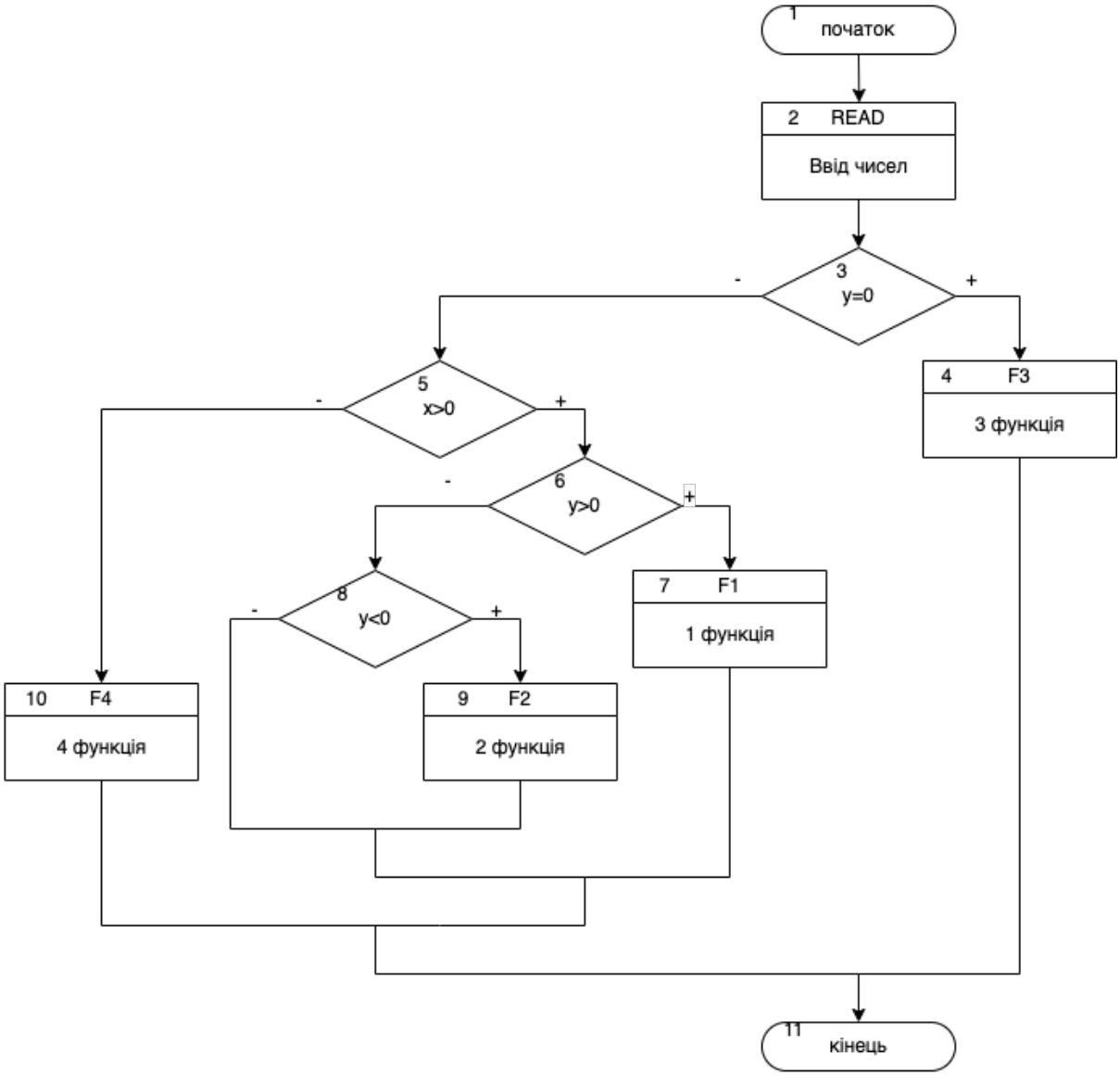


Рисунок 3.1 Схема функціонування головної процедури

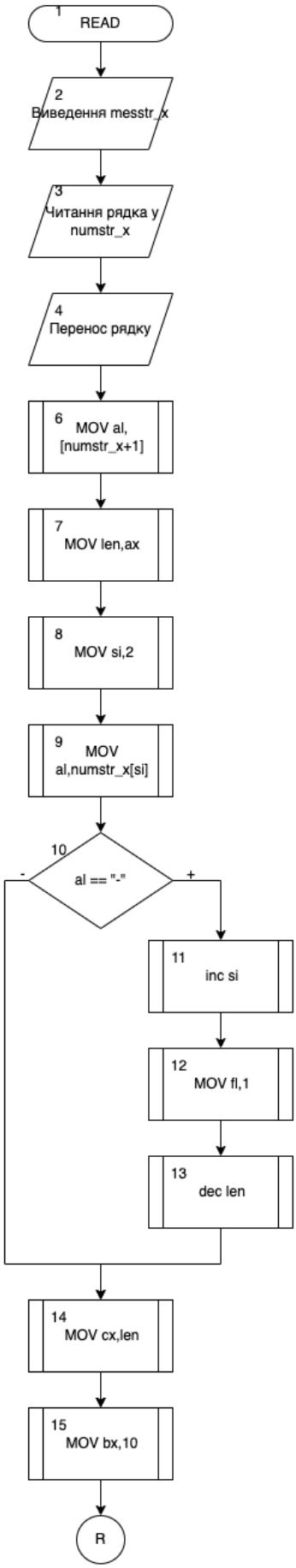
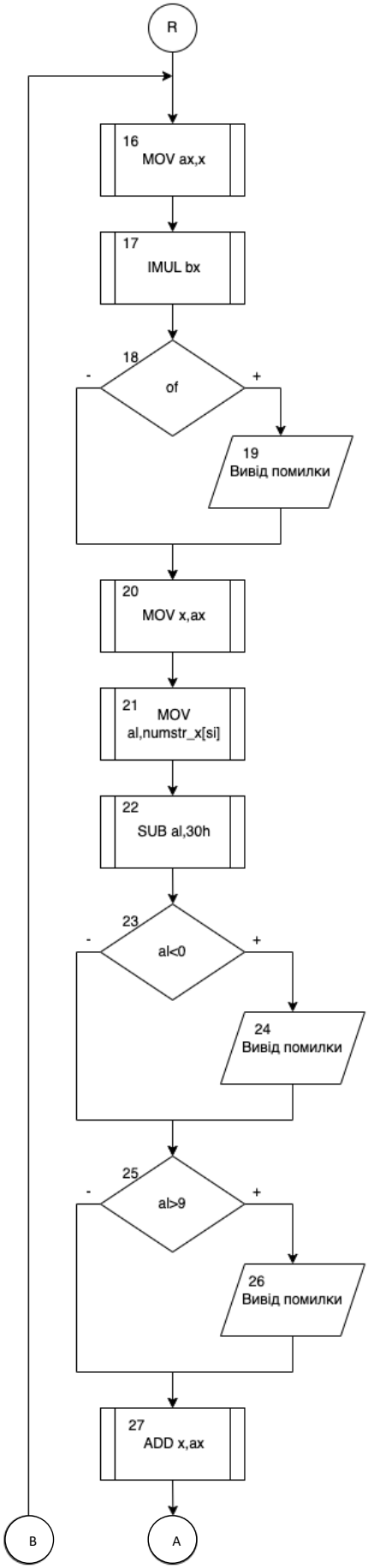
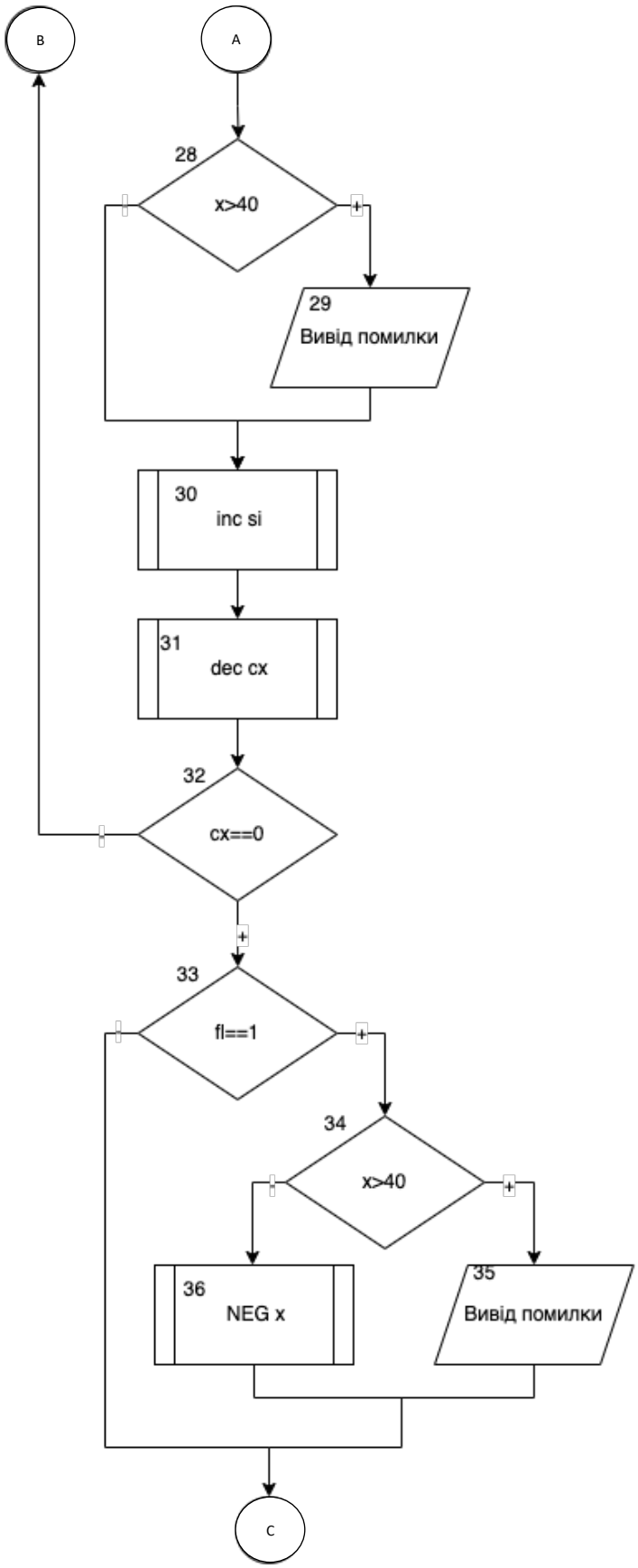


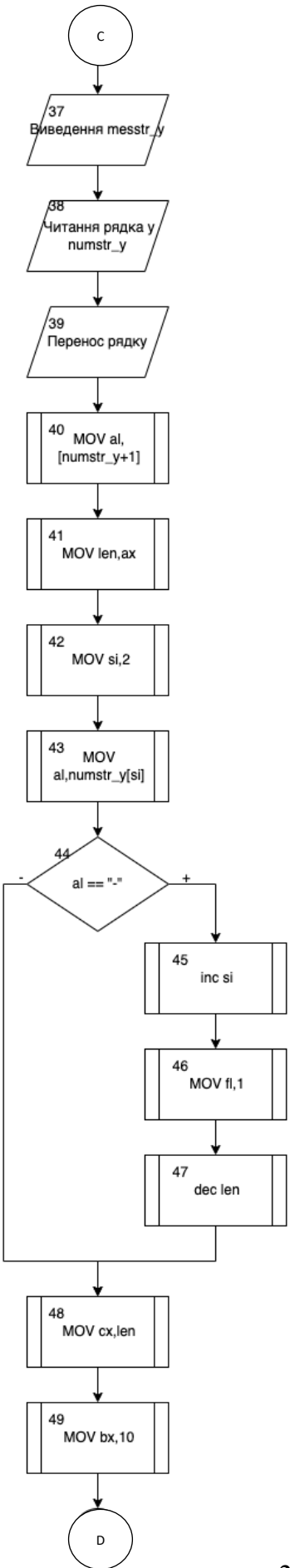
Рисунок 3.2 Схема функціонування процедури читання числа



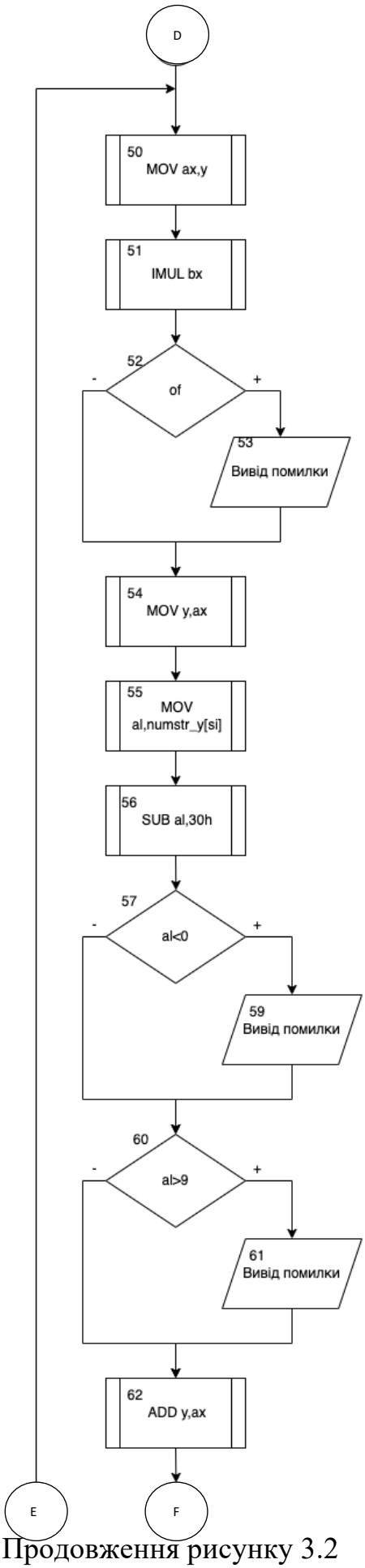
Продовження рисунку 3.2



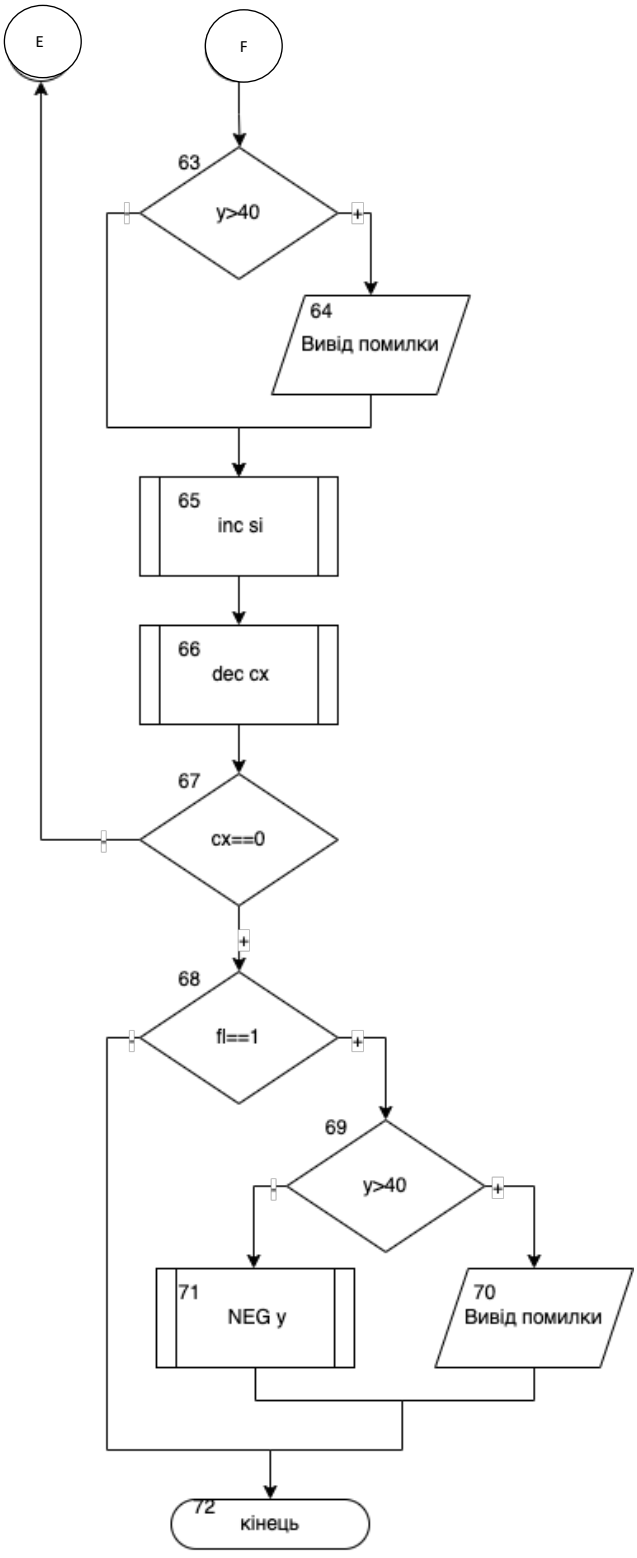
Продовження рисунку 3.2



Продовження рисунку 3.2



Продовження рисунку 3.2



Продовження рисунку 3.2

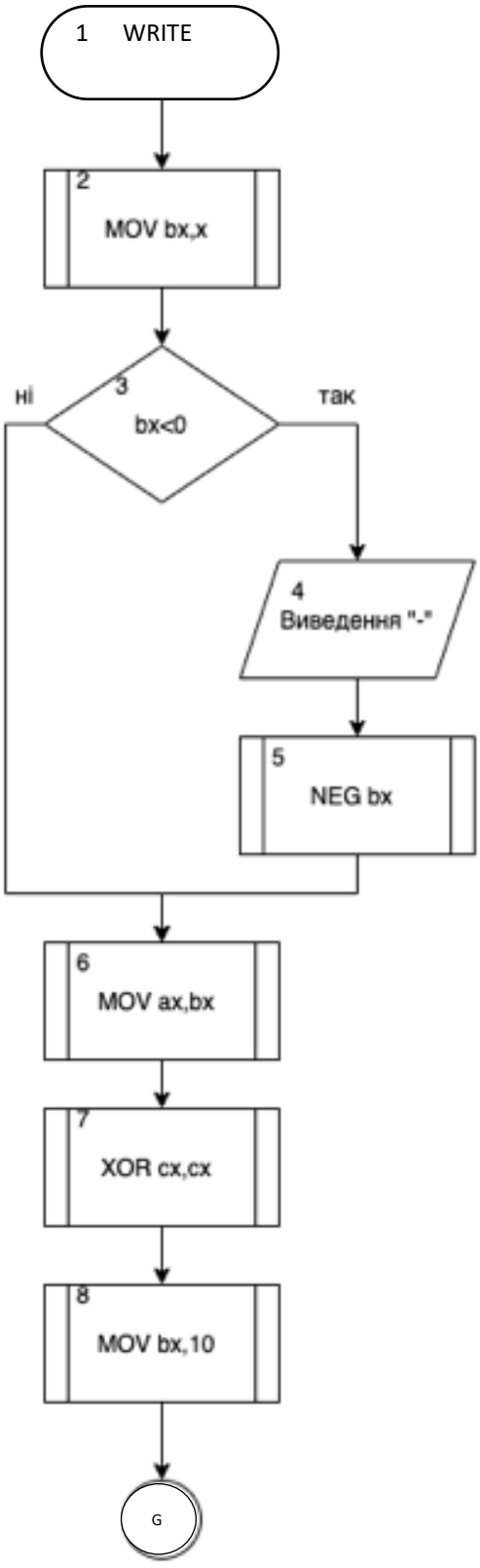
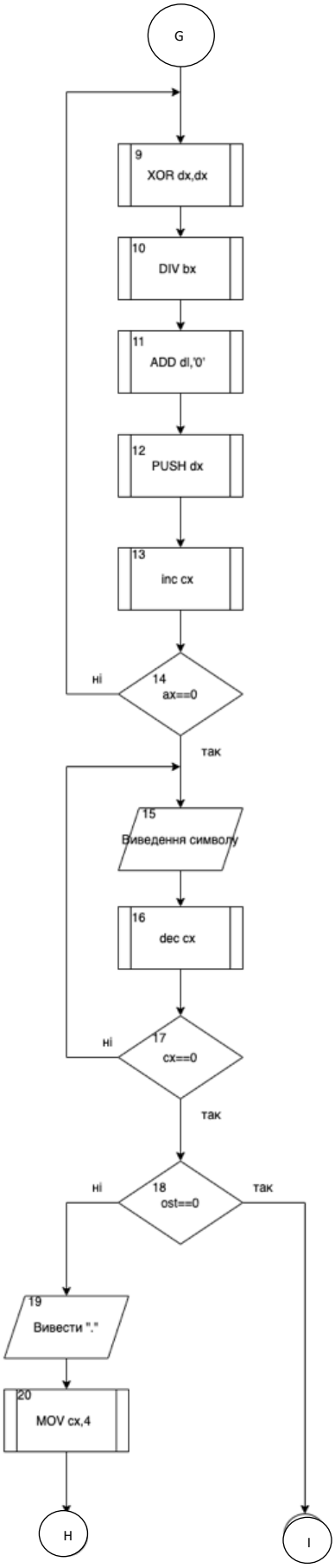
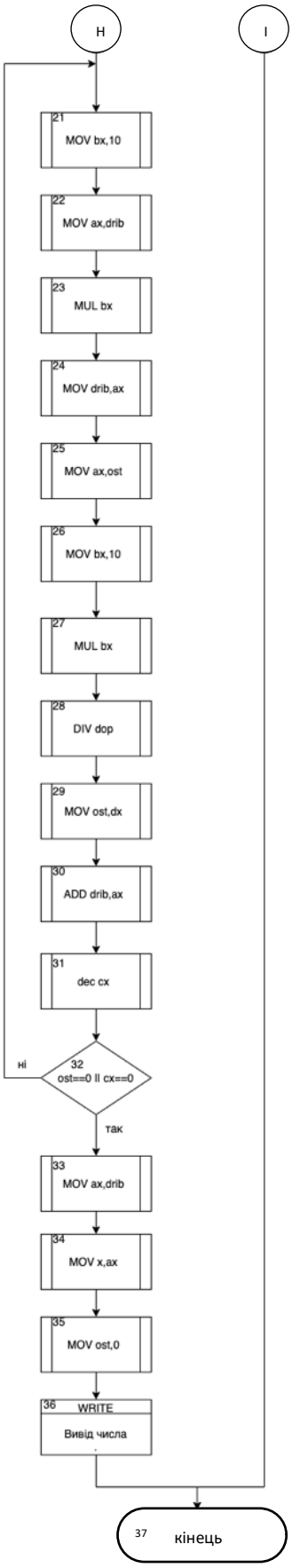


Рисунок 3.3 Схема функціонування процедури виведення числа



Продовження рисунку 3.3



Продовження рисунку 3.3

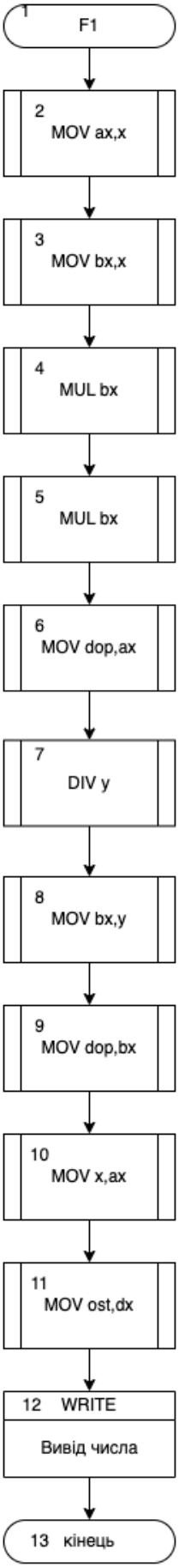


Рисунок 3.4 Схема функціонування процедури 1ї функції

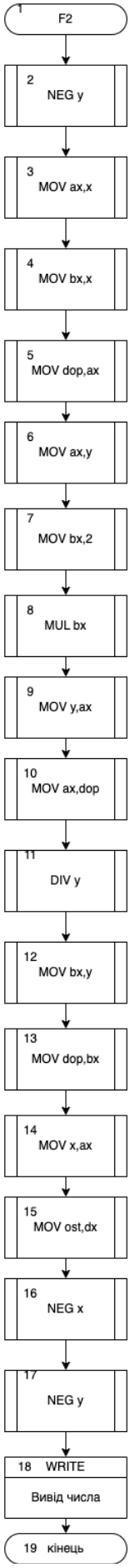


Рисунок 3.5 Схема функціонування процедури 2ї функції

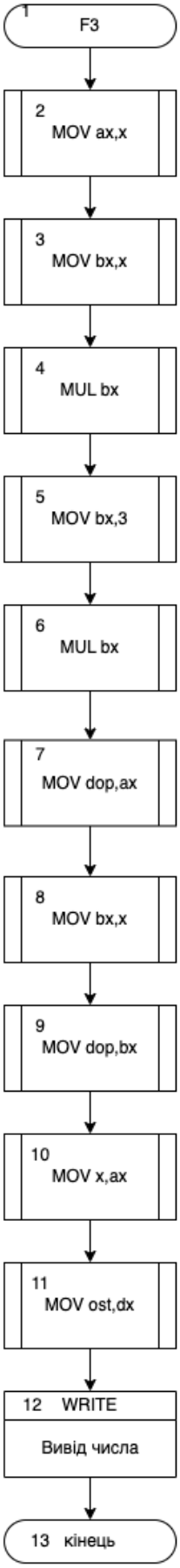
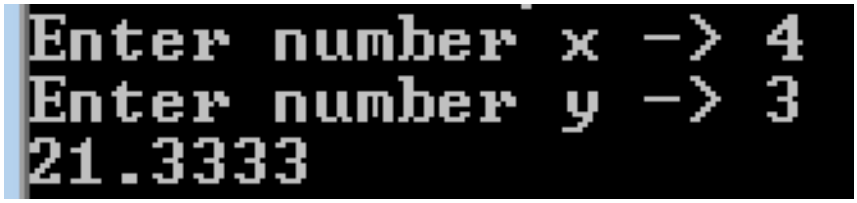
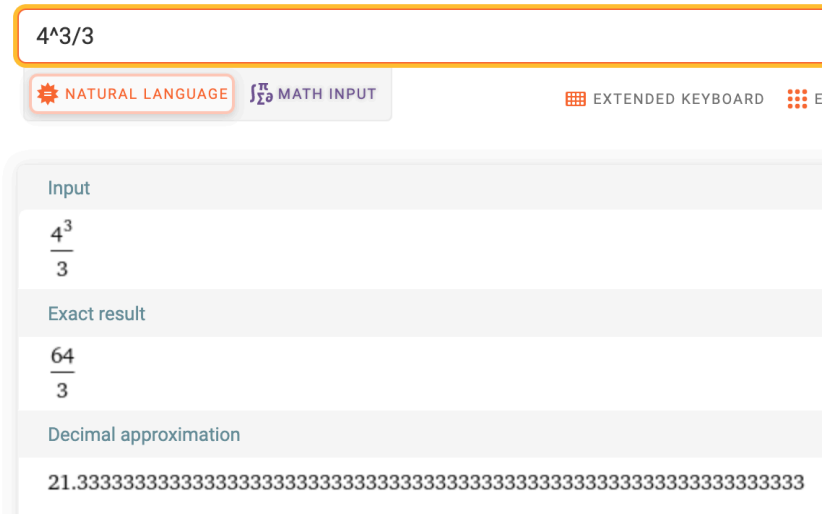


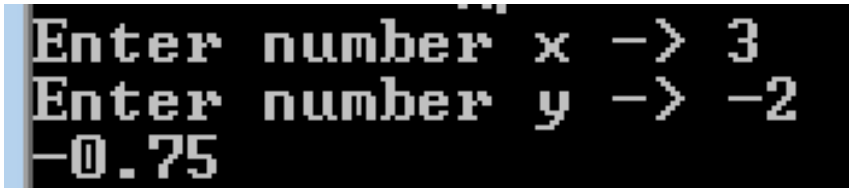
Рисунок 3.6 Схема функціонування процедури 3ї функції

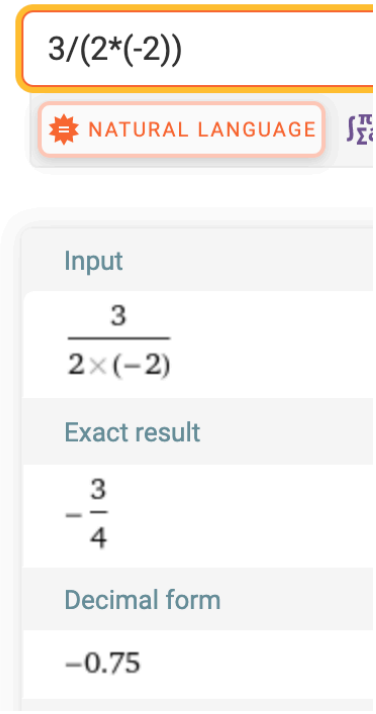
Рисунок 3.7 Схема функціонування процедури 4ї функції

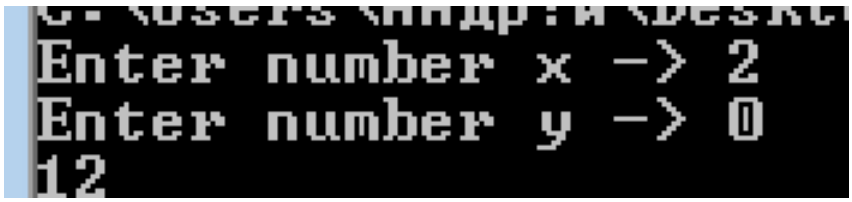
Приклади виконання програми

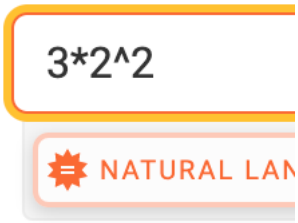


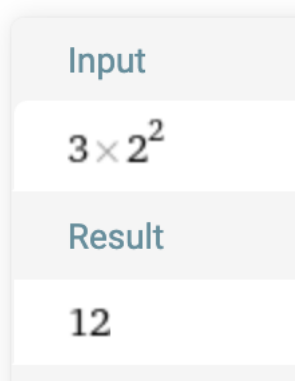


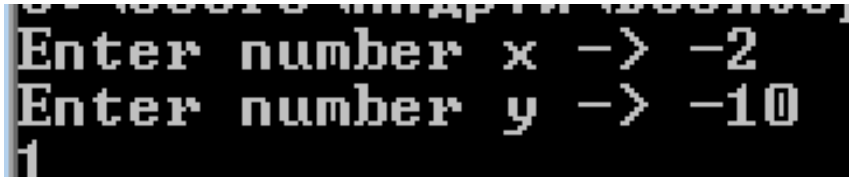












Висновок: Під час виконання комп’ютерного практикуму мною було створено програму, яка розраховує значення функції, що має відмінні правила розрахунку при значеннях: x>0&&y>0, x>0&&y<0, y=0, інші значення; та було протестовано програму на різних значеннях. Програма працює на проміжку від -40 до 40 для x та y.