

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної  
техніки  
Кафедра інформатики та програмної інженерії

*Звіт до комп'ютерного практикуму з дисципліни*  
*«Системне програмне забезпечення»*

**Прийняв**  
**асистент кафедри ІІІ**  
**Пархоменко А.В.**  
**“29” травня 2023 р.**

**Виконав**  
**Студент групи ІІІ-14**  
**Хільчук А.В.**

## Комп'ютерний практикум №3

**Тема:** програмування розгалужених алгоритмів.

### Завдання:

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:

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

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

Вигляд функції:

19.	$Z = \begin{cases} ax^2 + b / x & \text{якщо } x > 0 \\ a + 2b & \text{якщо } x = 0 \\ ax^2 - bx & \text{якщо } x < 0 \end{cases}$
-----	--

### Виконання:

*Текст програми:*

```
STSEG SEGMENT PARA STACK 'STACK'
    DB 256 DUP ('STACK')
STSEG ENDS
DSEG SEGMENT PARA PUBLIC 'DATA'
    bufer DB 7, 0, 7 dup (0)
    prompt DB 'Enter a number: $'
    validation_failed DB 13,10,'Wrong input. Ending the execution...$'
    result_zalishok db ' zalishok $'
    prompt_x db 'Please, enter x $'
    prompt_a db 13,10,'Please, enter a $'
    prompt_b db 13,10,'Please, enter b $'
    prompt_overflow db 13,10,'Overflow has occurred! Ending program execution $'
    newline db 13,10,'$'
    a dw 0
    b dw 0
    x dw 0
```

```
        chiselnik dw 0
DSEG ENDS
CSEG SEGMENT PARA PUBLIC 'CODE'

MAIN PROC FAR
    ASSUME CS:CSEG, DS:DSEG, SS:STSEG
    MOV AX, DSEG
    MOV DS, AX

    call input_values
    call calculate_function
    call print_result
    call stop_exec
MAIN ENDP
```

```
input_values proc
```

```
    MOV DX, offset prompt_x
    MOV AH, 9
    INT 21h
```

```
    call single_read_input
    mov x,AX
```

```
    MOV DX,offset prompt_a
    MOV AH, 9
    INT 21h
```

```
    call single_read_input
    mov a,AX
```

```
    MOV DX,offset prompt_b
    MOV AH, 9
    INT 21h
```

```
    call single_read_input
    mov b,AX
```

```
    ret
```

```
input_values endp
```

```
single_read_input proc
```

```

; read the input
lea dx, bufer
MOV ah, 10
int 21h
call validate_input
call bufer_chars_to_digits
call bufer_digits_to_number
ret

```

```

single_read_input endp

```

```

bufer_chars_to_digits PROC
    MOV CL, [bufer+1]
    LEA SI, bufer+2

    cmp byte ptr [si], 2Dh
    jne NO_MINUS
    sub CL, 1
    inc si
NO_MINUS:
    FOR_LOOP_TO_DIGITS:
        sub byte ptr [si], 48
        inc si;
        LOOP FOR_LOOP_TO_DIGITS    ; decrement CX and jump to
FOR_LOOP if CX is not zero
    ret
bufer_chars_to_digits ENDP

```

```

bufer_digits_to_number PROC
    LEA SI, bufer+2;
    mov ax, 0
    mov cx, 10
    cmp byte ptr [bufer+2], '-'
    jne FOR_LOOP_TO_NUMBERS
    INC SI
    FOR_LOOP_TO_NUMBERS:
        MUL CX

        ADD AL, [SI]
        INC SI

        cmp byte ptr [si], 13

```

jne FOR\_LOOP\_TO\_NUMBERS

cmp byte ptr[bufer+2], '-'

jne TO\_NUMBER\_END

neg ax

TO\_NUMBER\_END:

ret

bufer\_digits\_to\_number ENDP

bx\_result\_print proc

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

ret

bx\_result\_print endp

stop\_exec proc

MOV AH, 4Ch

INT 21h

stop\_exec endp

validate\_input proc

MOV CL, [bufer+1]

LEA SI, bufer+2

cmp byte ptr [si],2Dh

jne FOR\_LOOP\_VALIDATION

sub CL,1

inc si

FOR\_LOOP\_VALIDATION:

cmp byte ptr [si],48

JL VALIDATION\_FAILURE

cmp byte ptr [si],57

JGE VALIDATION\_FAILURE

inc si;

LOOP FOR\_LOOP\_VALIDATION

ret

VALIDATION\_FAILURE:

MOV DX, OFFSET validation\_failed

MOV AH, 9

INT 21h

call stop\_exec

validate\_input endp

calculate\_function proc

cmp [x],0

JE x\_is\_zero

JS x\_less\_than\_zero

mov ax, x

imul x

JO func\_overflow

imul x

JO func\_overflow

imul a

JO func\_overflow

add ax,b

JO func\_overflow

mov chiselnik, ax

ret

x\_is\_zero:

```

    mov ax,a
    add ax,b
    JO func_overflow
    add ax,b
    JO func_overflow

    mov chiselnik, ax
    ret
x_less_than_zero:
    mov ax,a
    imul x
    jo func_overflow
    sub ax,b
    imul x
    jo func_overflow
    mov chiselnik, ax
    ret
func_overflow:
    MOV DX,offset prompt_overflow
    MOV AH, 9
    INT 21h

    call stop_exec
calculate_function endp

print_result proc
    MOV DX,offset newline
    MOV AH, 9
    INT 21h
    mov bx,chiselnik
    call bx_result_print
    cmp byte ptr [x], 0
    jg drobom
    ret
drobom:

    MOV DL, '/'
    MOV AH, 02h
    INT 21h

    mov bx,x
    call bx_result_print

    MOV DL, '='

```

MOV AH, 02h  
INT 21h

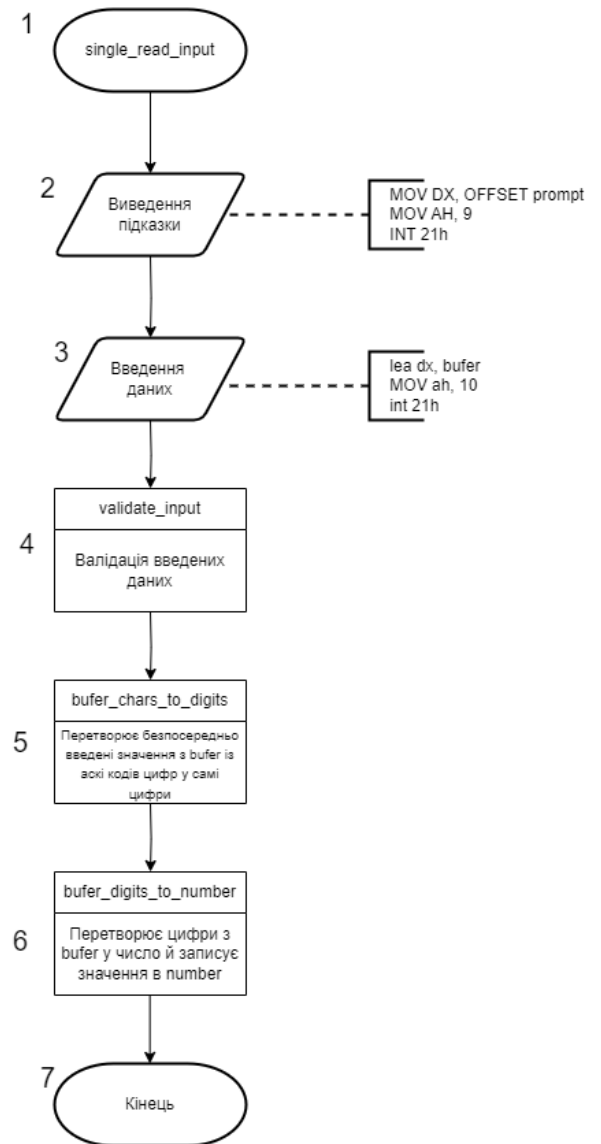
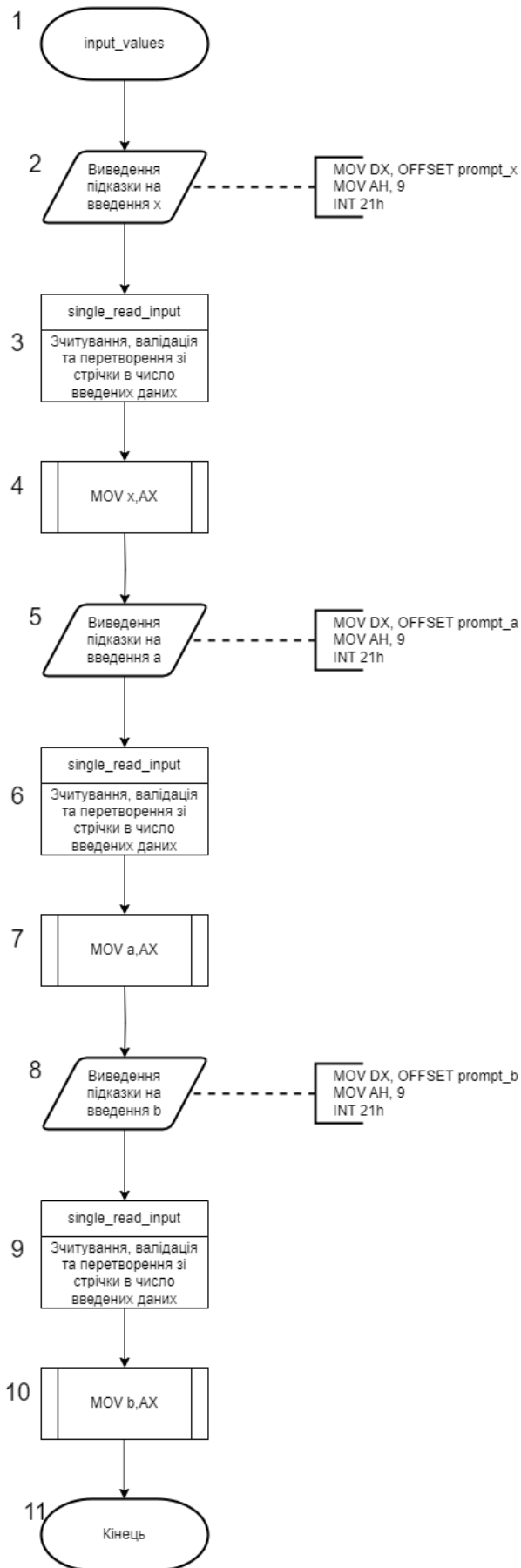
mov dx,0  
mov ax,chiselnik  
div x  
mov bx,ax  
push dx  
call bx\_result\_print  
pop dx  
cmp dx,0  
je print\_end  
push dx  
MOV DX,offset result\_zalishok  
MOV AH, 9  
INT 21h  
pop bx  
call bx\_result\_print  
print\_end:  
ret  
print\_result endp

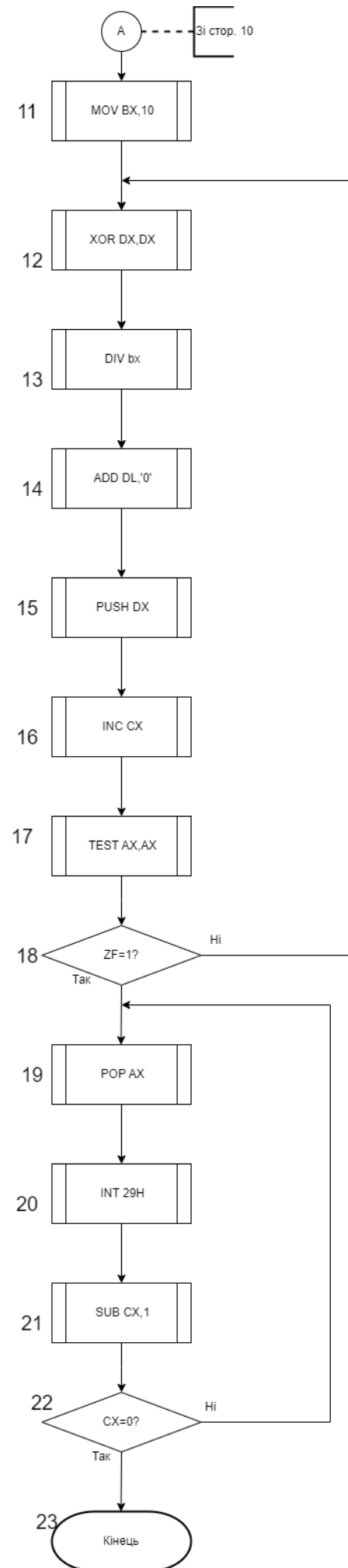
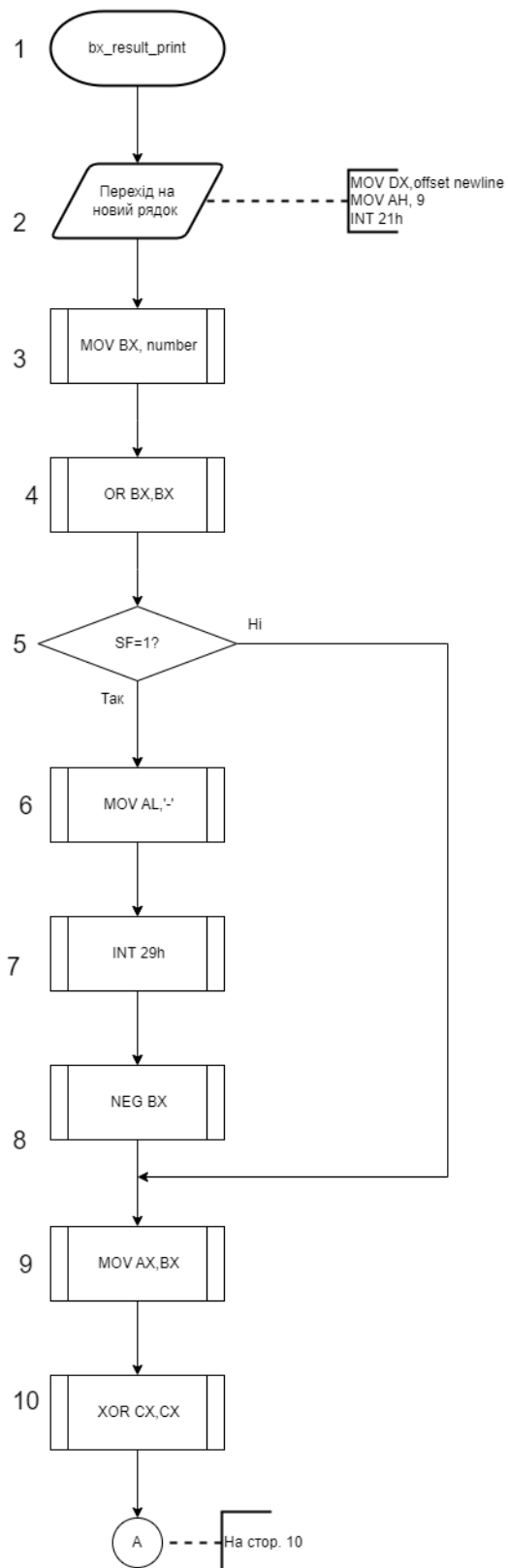
CSEG ENDS

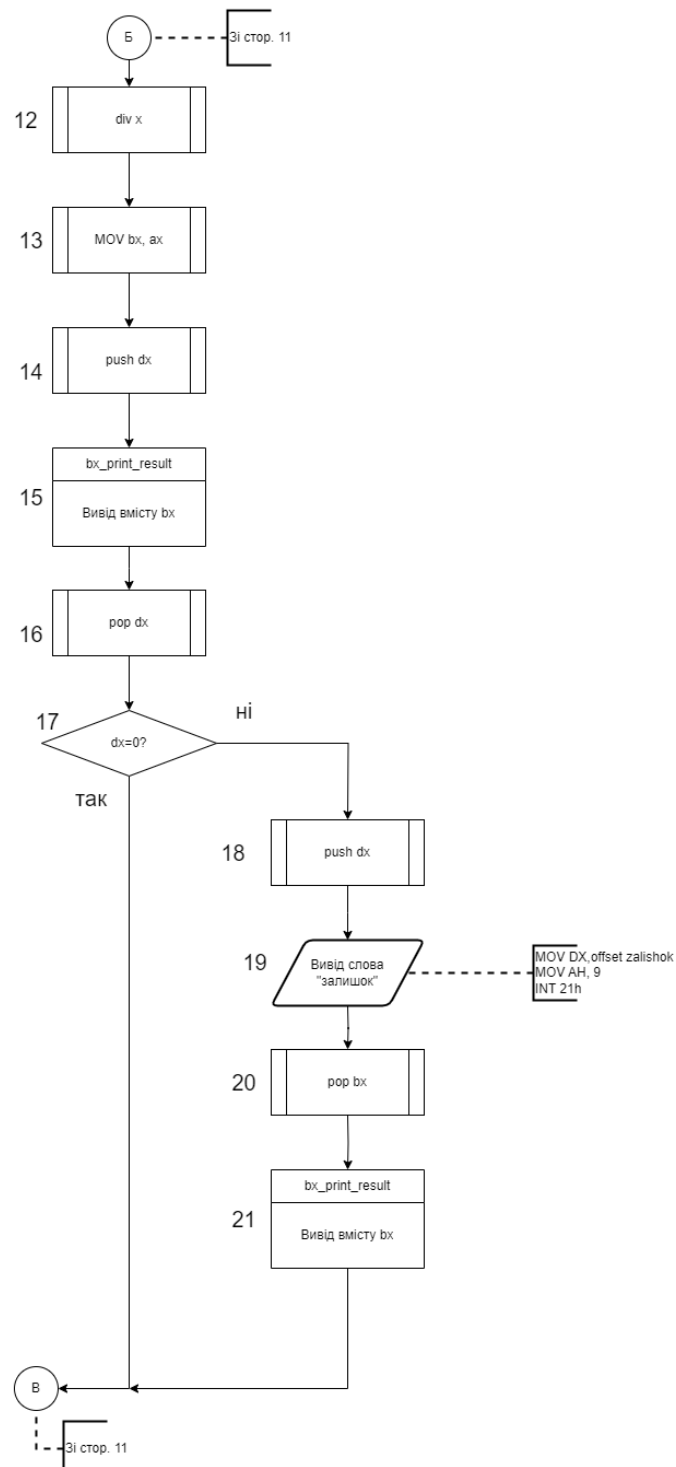
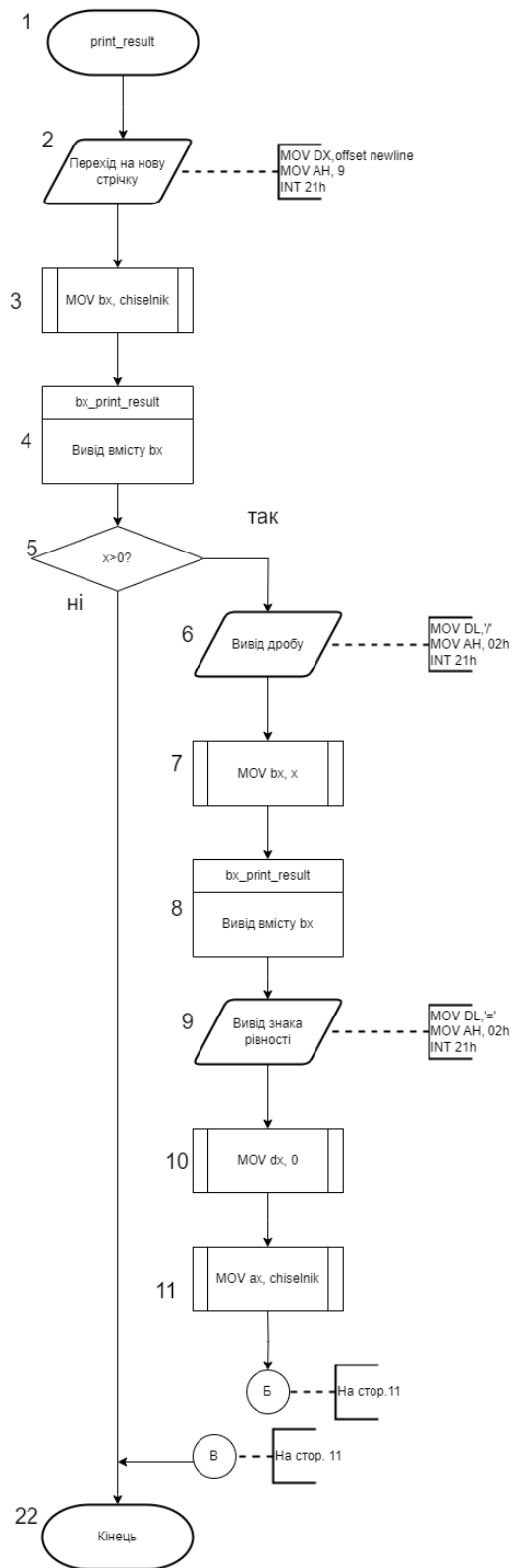
END MAIN

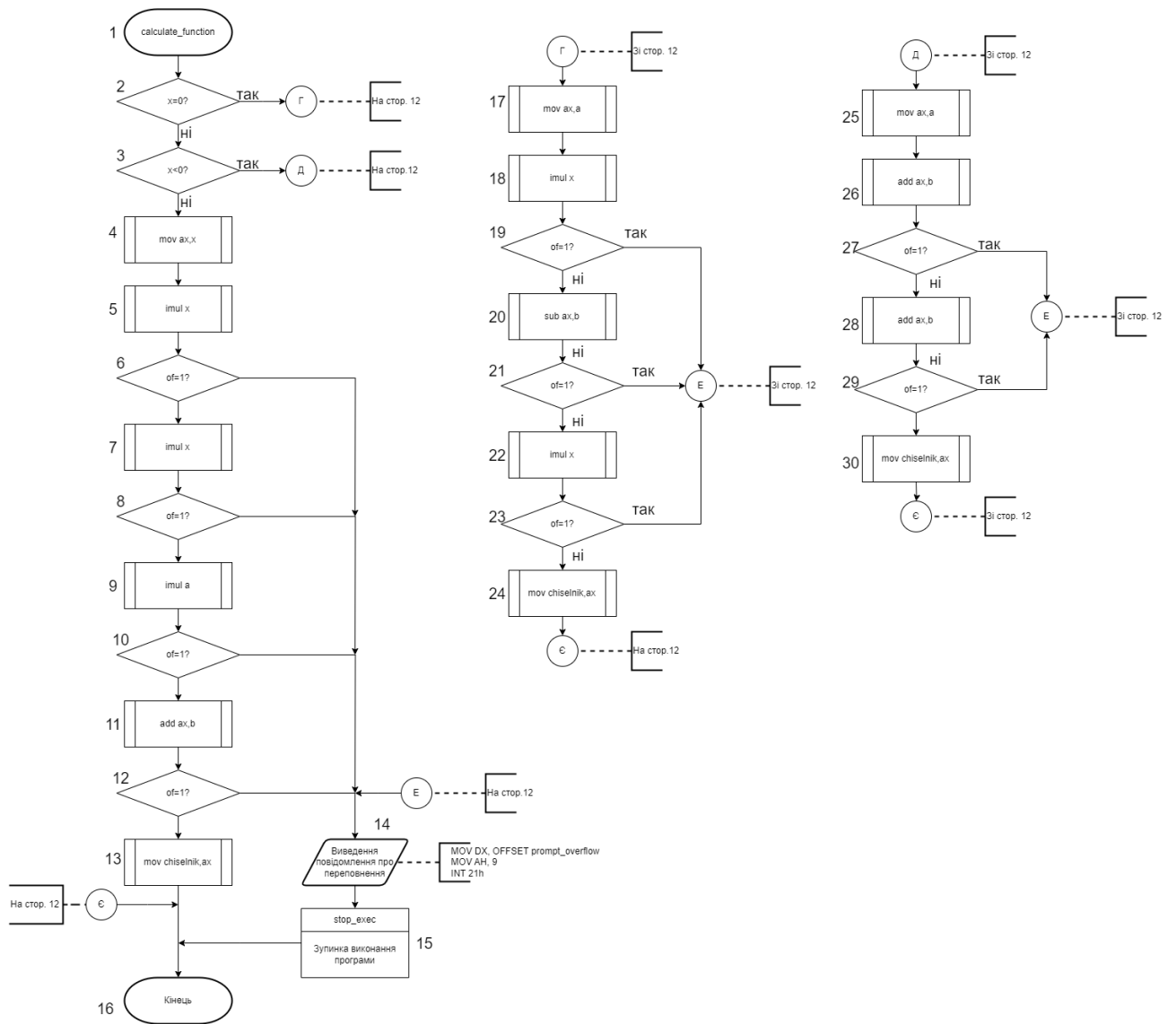
**Схема функціонування програми:**

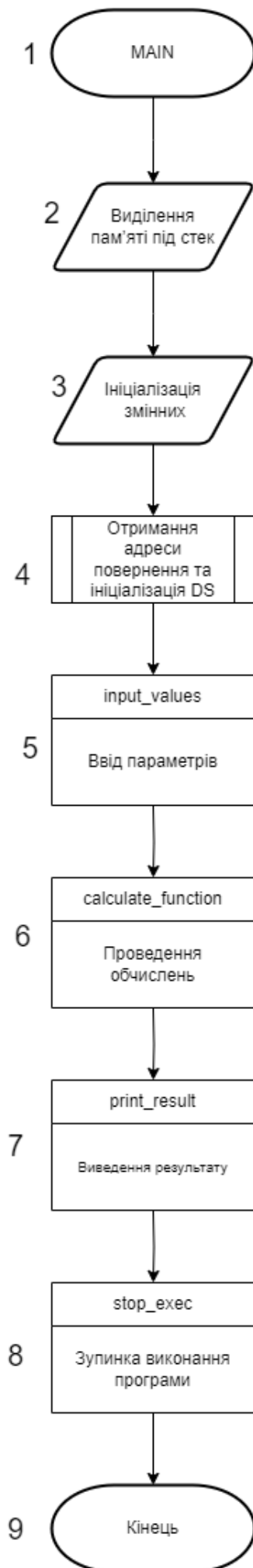
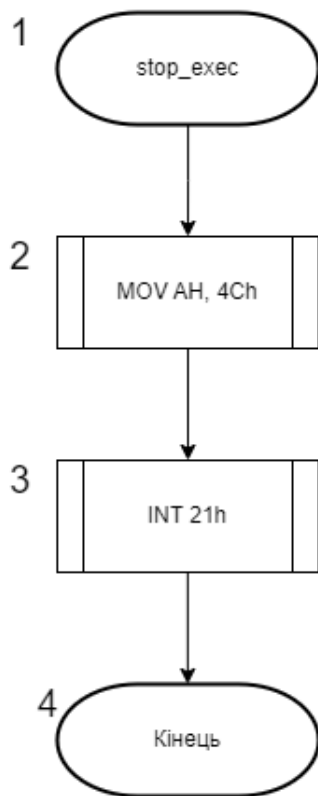




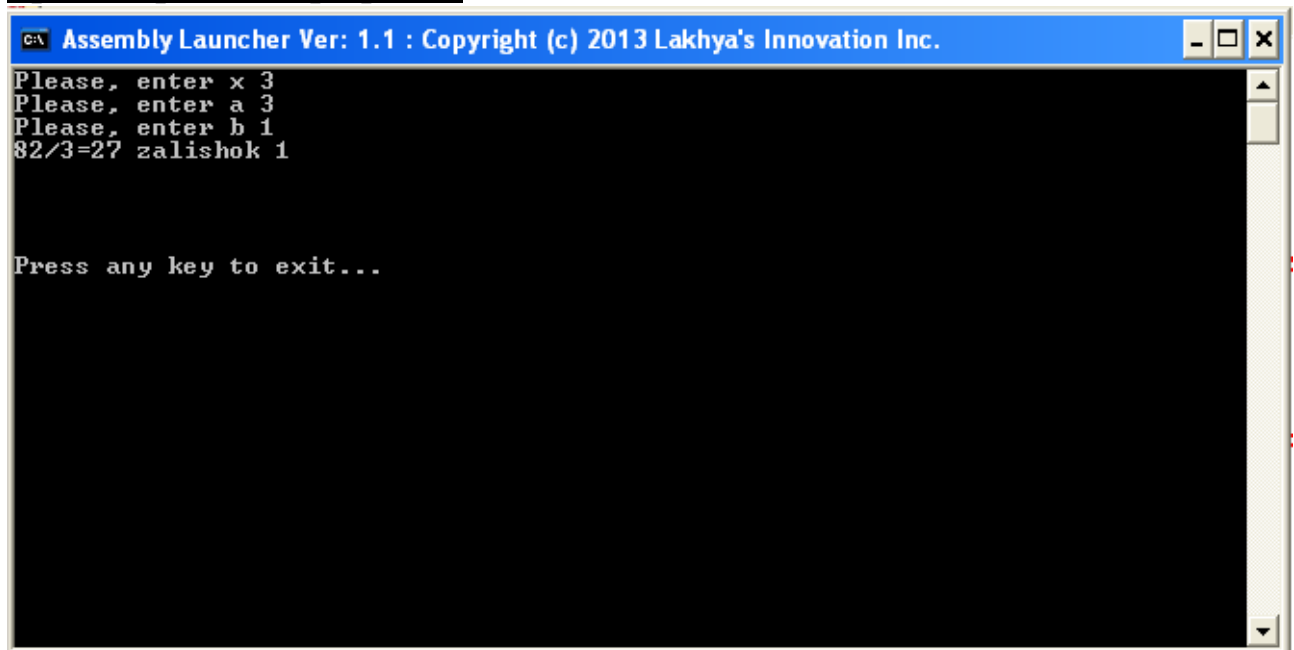








**Приклад роботи програми:**

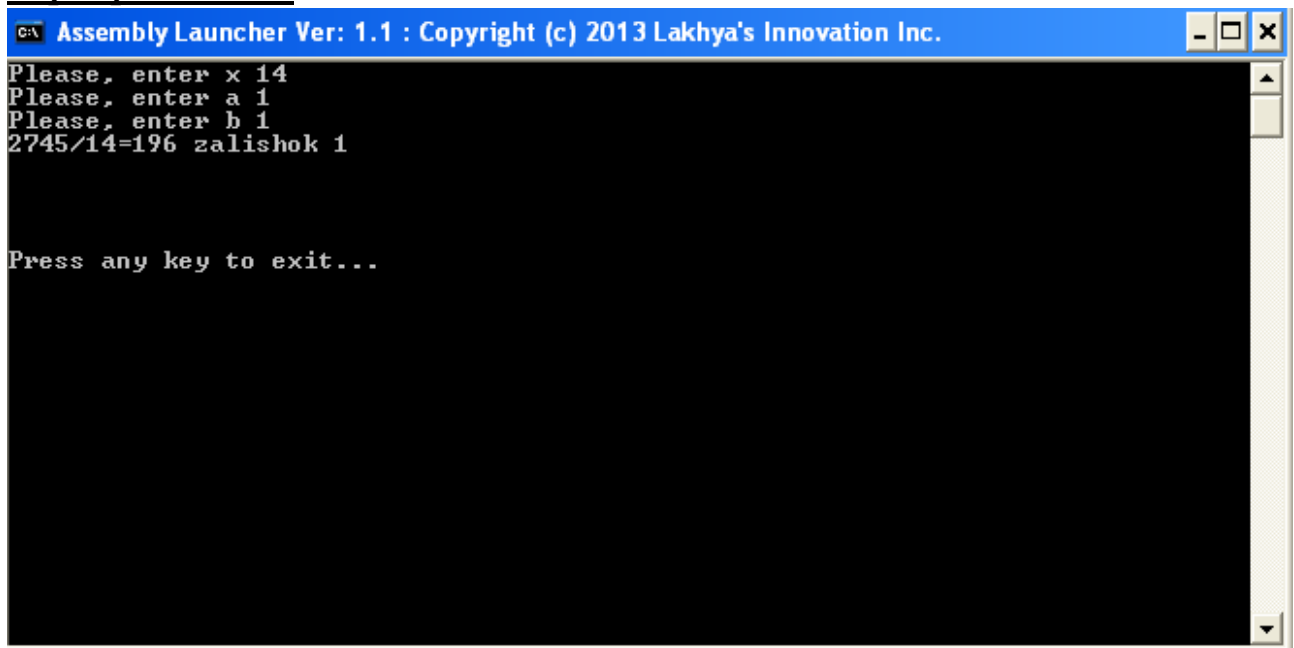


C:\ Assembly Launcher Ver: 1.1 : Copyright (c) 2013 Lakhya's Innovation Inc.

```
Please, enter x 3
Please, enter a 3
Please, enter b 1
82/3=27 zalishok 1

Press any key to exit...
```

**Перевірочні дані:**



C:\ Assembly Launcher Ver: 1.1 : Copyright (c) 2013 Lakhya's Innovation Inc.

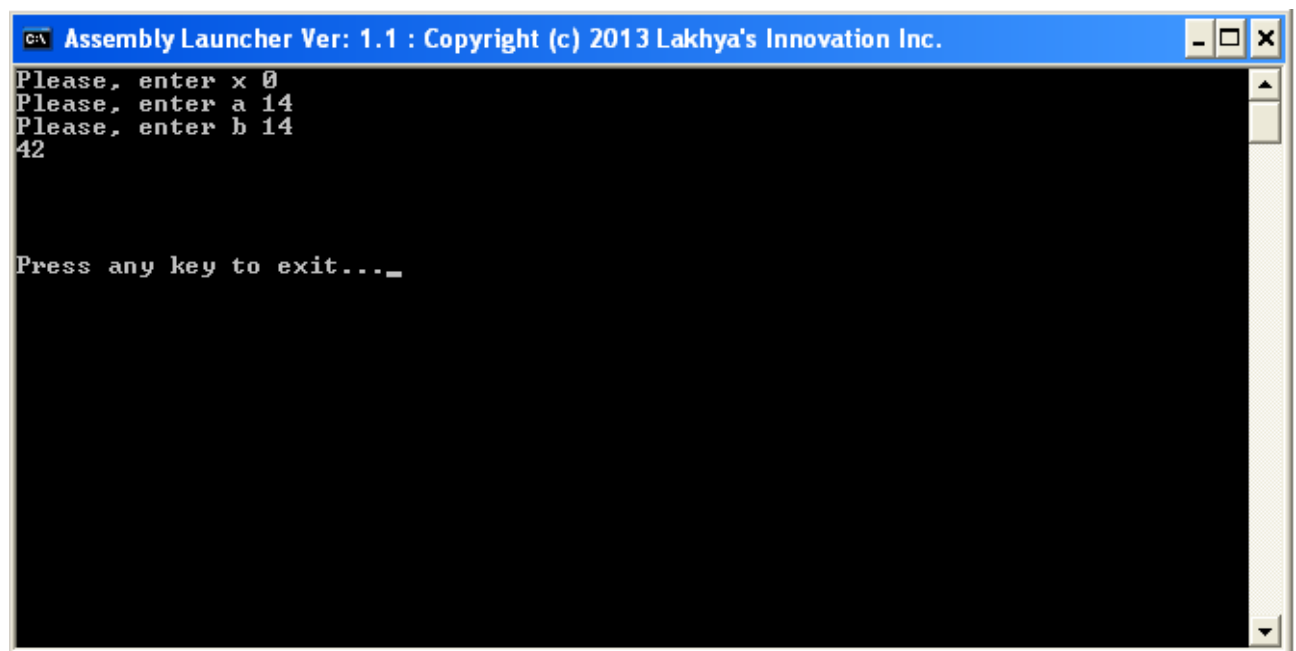
```
Please, enter x 14
Please, enter a 1
Please, enter b 1
2745/14=196 zalishok 1

Press any key to exit...
```



```
C:\> Assembly Launcher Ver: 1.1 : Copyright (c) 2013 Lakhya's Innovation Inc.
Please, enter x -14
Please, enter a 1
Please, enter b 1
210

Press any key to exit...._
```



```
C:\> Assembly Launcher Ver: 1.1 : Copyright (c) 2013 Lakhya's Innovation Inc.
Please, enter x 0
Please, enter a 14
Please, enter b 14
42

Press any key to exit...._
```

## Висновок:

Отож, у ході виконання лабораторної роботи було проаналізовано поставлене завдання, ознайомлено з теоретичної базою завдання та створено програмне забезпечення для обрахунку встановленої функції у залежності від введених значень. Було створено низку функцію у рамках декомпозиції завдання, а саме: функції вводу, валідації, переведення масиву цифр у число, а також введення усіх значень, що необхідні для виконання обрахунків; функції переведення числа у послідовність аскі кодів його цифр, виведення цих цифр, а також функцію виведення результату обчислень у форматі “5/3=1 zalishok 2”. Систему було побудовано з урахуванням найвірогідніших відмов: неправильного вводу, переповнення, ін.. Під час виконання лабораторної роботи було набуто практичних навичок роботи з командами умовних переходів та

порівняння. Урешті-решт, було проведено тестування створеного ПЗ та побудовано на основі коду програми блок-схему.