

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

Звіти до комп'ютерних практикумів дисципліни

«Системне програмне забезпечення»

Прийняв
доцент кафедри ІІІ
Лісовиченко О.І.
“26” травня 2023 р.

Виконав
Студент групи ІІІ-11
Прищеп В.С.

Київ-2023

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:
 - а) як дійсне число (наприклад: $5/3 = 1,666667$) – підвищена складність;
 - б) окремо цілу частину та остачу (наприклад: $5/3 = 1$ остача 2) – середня складність;
 - в) окремо цілу частину та остачу як дріб (наприклад: $5/3 = 1 \frac{2}{3}$) – 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.

$$21. \quad Z = \begin{cases} 5*((2+x)^{-1}+3*(1+x)^{-1}) & \text{якщо } x > 0 \\ 5 & \text{якщо } x = 0 \\ 5x^2 / (1-x) & \text{якщо } x < 0 \end{cases}$$

Текст програми:

```
STSEG SEGMENT PARA STACK "STACK"
DB 64 DUP("STACK")
STSEG ENDS
```

```
DSEG SEGMENT PARA PUBLIC "DATA"
x dw 0
nom dw 0
denom dw 0
inarr db 7,?,7 dup (" $")
input_tip db 13, 10, "x in [-114;-1], {0} or [1;254] => $"
is_negative db 0
num dw 0
digit dw 0
is_error db 0
error_message_1 db "Invalid symbol(s)!$"
error_message_2 db "Number out of diapason!$"
DSEG ENDS
```

```
CSEG SEGMENT PARA PUBLIC "CODE"
ASSUME CS:CSEG, DS:DSEG, SS:STSEG
```

```
main proc
```

;ds initialisation

mov ax, dseg

mov ds, ax

;showing a tip

lea dx, input_tip

mov ah, 9

int 21h

;input of number

call input_digit

call str_to_int

cmp is_error, 1

je raise_error_1

cmp is_error, 2

je raise_error_2

;calculation

mov ax, num

mov x, ax

call calc_res

cmp is_error, 2

je raise_error_2

;printing of the result

cmp num, 0

je fraction

call result_print

mov al, ''

int 29h

cmp nom, 0

jne fraction

jmp end_program

;printing of the fraction

fraction:

mov ax, nom

mov num, ax

call result_print

mov al, '/'

int 29h

mov ax, denom

mov num, ax

call result_print

jmp end_program

;invalid symbol(s) error

raise_error_1:

LEA dx, error_message_1

MOV ah,9

INT 21h

jmp end_program

;wrong number error

raise_error_2:

LEA dx, error_message_2

MOV ah,9

INT 21h

;program finishing

end_program:

mov AH, 4CH

int 21H

ret

main endp

input_digit proc

;input a string

lea dx, inarr

mov ah, 10

int 21h

;print a new line after the input

mov al,10

int 29h

mov al,13

int 29h

ret

input_digit endp

str_to_int proc

;load the address of first element of the array

mov si, offset inarr + 2

mov cx, 0

;check if inarr is empty

mov ax, 0

mov al, inarr + 1

cmp al, 0

je error_1

;converting of the array into number

convert_loop:

;check if it is the end of inarr

```
mov ax, 0
mov al, inarr + 1
cmp al, cl
je last
```

;set element of inarr to al

```
mov al, [si]
```

;check if the character is between "0" and "9"

```
cmp al, '0'
jl check_minus
cmp al, '9'
jg error_1
inc cx
inc si
```

;jump for converting character to digit

```
jmp convert_to_digit
```

;check character for "-"

check_minus:

```
cmp al, '-'
jne error_1
```

;check if it is the first element in inarr

```
cmp cx, 0
jne error_1
mov is_negative, 1
inc cx
inc si
jmp convert_loop
```

;converting character to digit

convert_to_digit:

```
sub al, '0'
mov digit, ax
mov bx, 10
mov ax, num
mul bx
jc error_2
js error_2
```

```
    mov num, ax
    mov ax, digit
    add num, ax
    jc error_2
    js error_2
    jmp convert_loop
```

;Used wrong symbol(s)

```
error_1:
    mov is_error, 1
    jmp finish
```

;Inputed number out of range

```
error_2:
    mov is_error, 2
    jmp finish
```

;check if the number is negative

```
last:
    cmp is_negative, 1
    jne finish
    neg num
```

finish:

```
    ret
```

str_to_int endp

calc_res proc

;compare x with 0

```
    cmp x, 0
    je x_zero
    jl x_less
```

;x>0 $z=5*(4*x+7)/((x+1)*(x+2))$

```
    mov ax, x
    add ax, 1
    jc error
    mov bx, ax
    add ax, 1
    jc error
    mul bx
    jc error
    mov denom, ax
    mov ax, x
    mov bx, 4
```

```
mul bx
jc error
add ax, 7
jc error
mov bx, 5
mul bx
jc error
mov bx, denom
div bx
mov num, ax
mov ax, dx
mov nom, ax
jmp finishes
```

;x=0 z=5

```
x_zero:
    mov ax, 5
    mov num, ax
    jmp finishes
```

;x<0 z=5*x^2/(1-x)

```
x_less:
    mov ax, x
    neg ax
    jo error
    add ax, 1
    jc error
    mov denom, ax
    sub ax, 1
    mov bx, ax
    mul bx
    jc error
    mov bx, 5
    mul bx
    jc error
    mov bx, denom
    div bx
    mov num, ax
    mov ax, dx
    mov nom, ax
    jmp finishes
```

;x out of diapason

```
error:
    mov is_error, 2
```

```

    finishs:
        ret
calc_res endp

result_print proc

;prepare to split number into digits
    mov ax, num
    xor cx, cx
    mov bx, 10

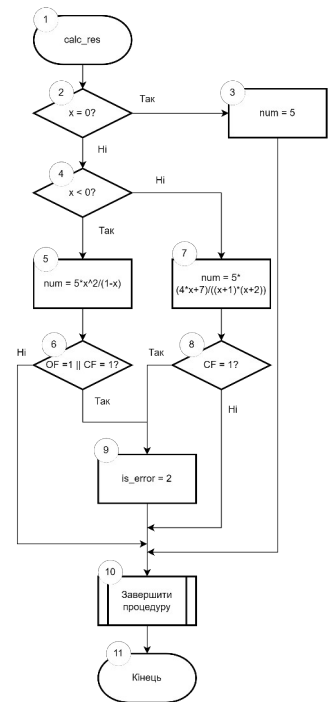
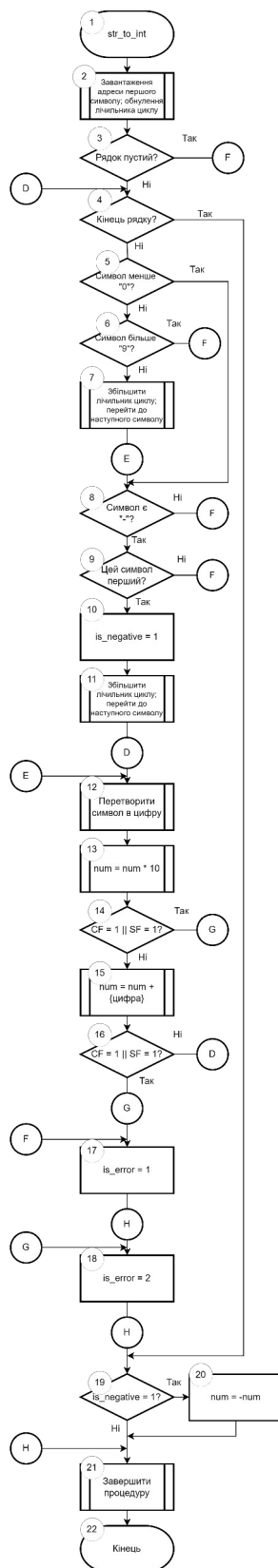
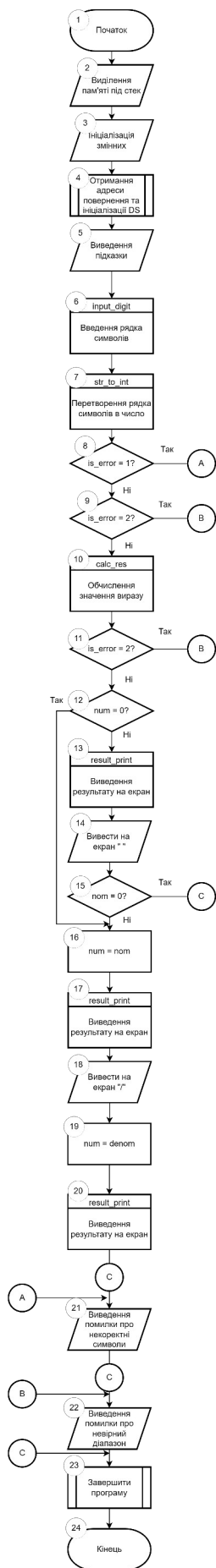
;split it into digits and write it down
m1:
    xor dx, dx
    div bx
    add dl, '0'
    push dx
    inc cx
    test ax, ax
    jnz m1

;print these digits in correct order
m2:
    pop ax
    int 29h
    loop m2
    ret
result_print endp

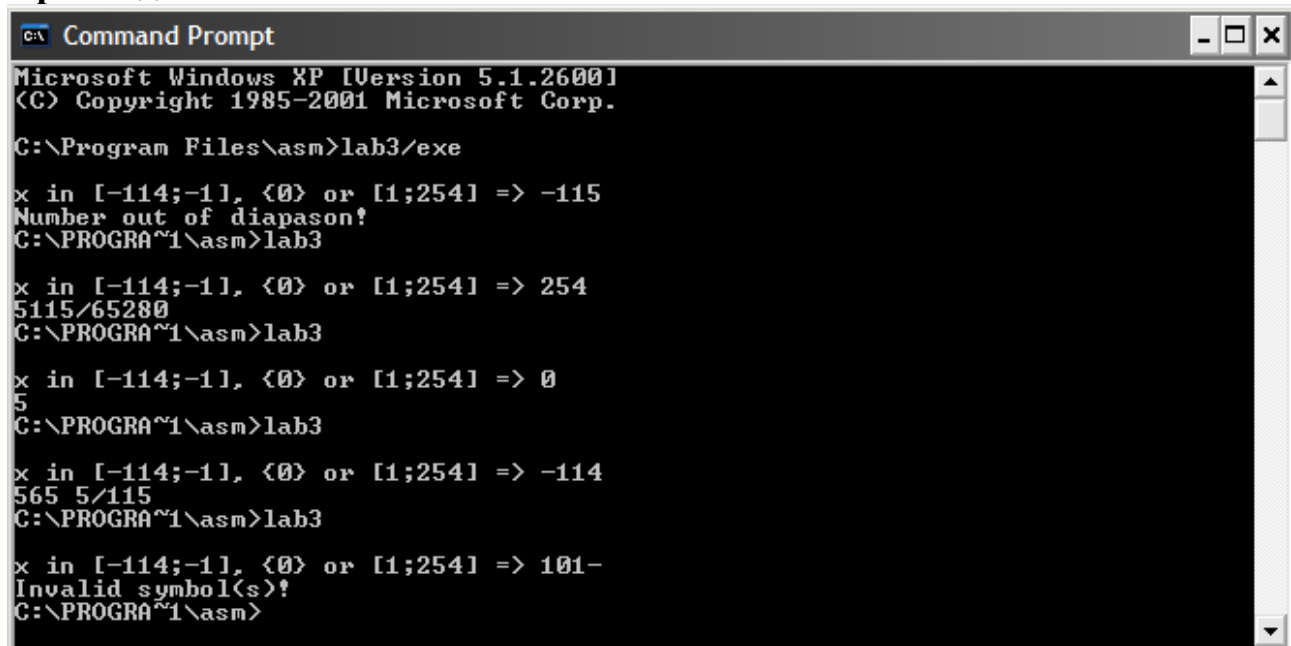
CSEG ENDS
END MAIN

```

Схема функціонування програми:



Приклад виконання:



```
C:\ Command Prompt
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Program Files\asm>lab3/exe

x in [-114;-1], {0} or [1;254] => -115
Number out of diapason!
C:\PROGRAM FILES\asm>lab3

x in [-114;-1], {0} or [1;254] => 254
5115/65280
C:\PROGRAM FILES\asm>lab3

x in [-114;-1], {0} or [1;254] => 0
5
C:\PROGRAM FILES\asm>lab3

x in [-114;-1], {0} or [1;254] => -114
565 5/115
C:\PROGRAM FILES\asm>lab3

x in [-114;-1], {0} or [1;254] => 101-
Invalid symbol(s)!
C:\PROGRAM FILES\asm>
```

У першому та п'ятому викликах були введені значення поза діапазоном та не число відповідно, що призвело до помилки. У всіх інших викликах були введені значення для всіх випадків, які передбаченні програмою.

У третьому виклику $x = 0$, за умовами функції $y = 5$.

У другому виклику $x = 254$. Отже $y = 5 * ((2 + x)^{-1}) + 3 * (1 + x)^{-1} = 5 * (1 / (256) + 3 / (255)) = 5 * (255 + 3 * 256) / (256 * 255) = 5115 / 65280$.

У четвертому виклику $x = -114$, отже $y = 5 * x^2 / (1 - x) = 5 * (-114)^2 / (1 + 114) = 64980 / 115 = 565 \frac{5}{115}$.

Бачимо, що підраховані вручну результати співпадають з результатами виконання програми.

Висновок:

1. Реалізована можливість введення користувачем значення x .
2. Реалізована процедура, що обчислює значення функції за введеними значенням x .
3. Реалізовано вивід на екран результату обчислень. Результати виконання програми в прикладі перевірені вручну.
4. Якщо є ділення, то результат виводиться так: окремо ціла частину та остача як дріб (наприклад: $5/3 = 1 \frac{2}{3}$).
5. Програма має захист від некоректного введення вхідних даних (символи, переповнення), що було продемонстровано в звіті.