Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**ЛАБОРАТОРНА РОБОТА № 6**

з дисципліни «Системне програмування» на тему

«Арифметичні і логічні операції з цілими числами. Масиви.»

Виконав: Перевірив:

Студент 2 курсу ФІОТ-у доц. Павлов В.Г.

групи ІМ-33

Козарезов Кирил Олександрович

номер у списку групи - 10

Київ 2025

**Мета роботи**: Вивчення арифметичних і логічних команд Асемблера і здобуття навиків виконання розрахунків з елементами масивів.

**Порядок виконання роботи**

1. Вивчити арифметичні команди з плаваючою комою.

2. Розробити програму на мові Асемблер, в якій згідно з індивідуальним варіантом завдання виконуються обчислення значення арифметичного вираження із застосуванням команд співпроцесора **ix87** з подальшим виведенням результату у віконному інтерфейсі.

3. Для всіх варіантів передбачити завдання значень вхідних змінних у форматі **double (DQ)**, проміжних результатів обчислень – у форматі **long double (DT)**, а кінцевих - знову у форматі **double**.

4. Розрахунки (п. 2, 3) повторити в програмі для 5 значень змінних, причому всі вхідні значення задати дійсними числами у вигляді одновимірних масивів.

5. Для перевірки правильності виконання розрахунків і результатів, що виводяться, заздалегідь виконати контрольні розрахунки. Проміжні і остаточні результати контрольних розрахунків з точністю не гірше точності розрахунків у комп'ютеру привести в звіті.

6. Виконати відладку програми шляхом порівняння розрахованих програмою результатів з контрольними прикладами. Лістинг розробленої програми і скріншоти розрахунків по всіх контрольних прикладах привести в звіті по лабораторній роботі.

7. Зробити висновки по лабораторній роботі.

**Виконання роботи**

Формула для розрахунку за 2 варіантом: (4\*c + d - 1.0) / (b - tan(a/2)).

Контрольні розрахунки

1. Знаменник і чисельник > 0

a = -5,5  
b = 4,8  
c = 2,7

d = -1,8

Обчислюємо:

(4\*2,7- 1,8 - 1) / (4,8 - tg(-5,5/2)) =8 / 4,8+tg(2,75) =  
= 8 /( 24/5)+tg(11/4)= 40/(24+5tg(11/4))= 1,82353550893042

1. Чисельник > 0, а знаменник < 0

a = -2.7

b = 5,6

c = -1,2

d = 2,9

Обчислємо:

(4\*(-1,2)+ 2,9 - 1) / (5,6- tg(-2,7/2)) =(29/10) / 5,6+tg(1,35) =  
= (29/10) /( 28/5)+tg(27/20)= -(29/56+10tg(27/20)= -0,288407363790067

1. Знаменник і чисельник <0

Вхідні параметри:

A = -1.047

B = -1.5

C = -0.25

D = 0.5

Обчислюємо:

(4\*-0.25+ 0.5-1)/(-1.5-tg(-1.047/2))=(-3/2)/((-3/2)+tg(0,5235))=-3/-3+2tg(1047/2000) = 1,6255203679091277

1. Чисельник <0 та знаменник >0

a = -2,8

b = 1,5

c = -1,5

d = 0,5

(4\*-1.5+ 0.5-1)/(1.5-tg(-2.8/2))= (-6,5) / ((3/2)-tg(1,4)) = -13/ (3+2tg(7/5)) = -0.890669156664613

Оскільки формула містить тригонометричну функцію, що має свою область визначення, а саме tg(x) – множина всіх дійсних чисел, де x != pi/2 + pi\*n, додамо приклад з порушенням цієї області. Також оскільки pi – ірраціональне число, візьмемо його наближене значення з восьма знаками після коми pi = 3,141592

1. Порушення області визначення tg(-pi/2)

a = 3.141593

b = -0,5236

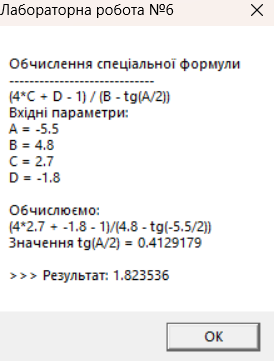
c = 1,5

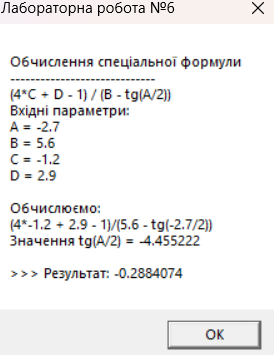
d = 3.1

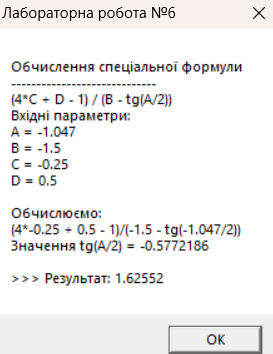
(4\*1.5+3,1-1)/(-0.5236-tg(3.141593/2))= 8.1 /(-0.5236- tg(1.5707965))

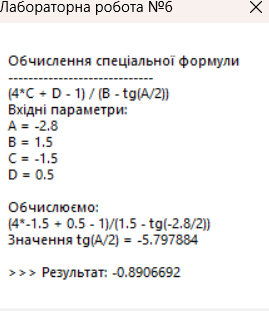
або ж 8.1 / (0.5236- tg(-pi/2)

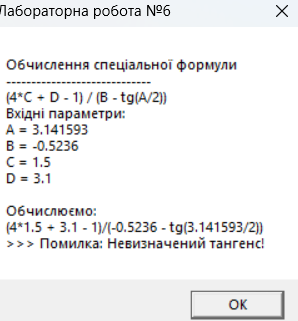
Робота програми











Порівнюючи результати виконання програми із контрольними розрахунками можна сказати, що програма працює коректно. Проте на відміну від контрольних розрахунків, комп’ютер виконує округлення до 6 знаків після коми. Тож перевірена робота програми у таких випадках:

1. Знаменник і чисельник > 0
2. Чисельник < 0, а знаменник > 0
3. Чисельник > 0, а знаменник < 0
4. Знаменник і чисельник < 0
5. Порушення області визначення tg(-pi/2)

Лістинг програми

.386

.model flat, stdcall

option casemap:none

include \masm32\include\masm32rt.inc

.data?

KozarezovOutBuf db 1024 dup(?)

KozarezovTempBuf db 256 dup(?)

KozarezovStrA db 32 dup(?)

KozarezovStrB db 32 dup(?)

KozarezovStrC db 32 dup(?)

KozarezovStrD db 32 dup(?)

KozarezovStrRes db 32 dup(?)

KozarezovStrTanVal db 32 dup(?)

KozarezovTemp dq ?

KozarezovNumer dq ?

KozarezovDenom dq ?

KozarezovTanVal dq ?

.data

KozarezovTitle db "Лабораторна робота №6", 0

KozarezovHeader db "Обчислення спеціальної формули", 13, 10

db "-----------------------------", 13, 10, 0

KozarezovFormula db "(4\*C + D - 1) / (B - tg(A/2))", 13, 10, 0

KozarezovInputFmt db "Вхідні параметри:", 13, 10

db "A = %s", 13, 10

db "B = %s", 13, 10

db "C = %s", 13, 10

db "D = %s", 13, 10, 13, 10, 0

KozarezovCalcFmt db "Обчислюємо:", 13, 10

db "(4\*%s + %s - 1)/(%s - tg(%s/2))", 13, 10, 0

KozarezovTanValFmt db "Значення tg(A/2) = %s", 13, 10, 13, 10, 0

KozarezovResultFmt db ">>> Результат: %s", 13, 10, 0

KozarezovErrorDivZero db ">>> Помилка: Ділення на нуль!", 13, 10, 0

KozarezovErrorTanDomain db ">>> Помилка: Невизначений тангенс!", 13, 10, 0

KozarezovDataA dq -5.5, -2.7, -1.047, -2.8, 3.141593

KozarezovDataB dq 4.8, 5.6, -1.5, 1.5, -0.5236

KozarezovDataC dq 2.7, -1.2, -0.25, -1.5, 1.5

KozarezovDataD dq -1.8, 2.9, 0.5, 0.5, 3.1

KozarezovFour dq 4.0

KozarezovOne dq 1.0

KozarezovTwo dq 2.0

KozarezovPi dq 3.141592

KozarezovPiDiv2 dq 1.570796

KozarezovEpsilon dq 0.000001

.code

KozarezovDivZero:

invoke wsprintf, addr KozarezovOutBuf, addr KozarezovHeader

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovFormula

invoke wsprintf, addr KozarezovTempBuf, addr KozarezovInputFmt,

addr KozarezovStrA, addr KozarezovStrB,

addr KozarezovStrC, addr KozarezovStrD

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovTempBuf

invoke wsprintf, addr KozarezovTempBuf, addr KozarezovCalcFmt,

addr KozarezovStrC, addr KozarezovStrD,

addr KozarezovStrB, addr KozarezovStrA

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovTempBuf

invoke FloatToStr, KozarezovTanVal, addr KozarezovStrTanVal

invoke wsprintf, addr KozarezovTempBuf, addr KozarezovTanValFmt, addr KozarezovStrTanVal

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovTempBuf

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovErrorDivZero

jmp KozarezovShowResult

KozarezovTanError:

invoke wsprintf, addr KozarezovOutBuf, addr KozarezovHeader

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovFormula

invoke wsprintf, addr KozarezovTempBuf, addr KozarezovInputFmt,

addr KozarezovStrA, addr KozarezovStrB,

addr KozarezovStrC, addr KozarezovStrD

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovTempBuf

invoke wsprintf, addr KozarezovTempBuf, addr KozarezovCalcFmt,

addr KozarezovStrC, addr KozarezovStrD,

addr KozarezovStrB, addr KozarezovStrA

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovTempBuf

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovErrorTanDomain

jmp KozarezovShowResult

KozarezovBuildOutput:

invoke wsprintf, addr KozarezovOutBuf, addr KozarezovHeader

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovFormula

invoke wsprintf, addr KozarezovTempBuf, addr KozarezovInputFmt,

addr KozarezovStrA, addr KozarezovStrB,

addr KozarezovStrC, addr KozarezovStrD

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovTempBuf

invoke wsprintf, addr KozarezovTempBuf, addr KozarezovCalcFmt,

addr KozarezovStrC, addr KozarezovStrD,

addr KozarezovStrB, addr KozarezovStrA

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovTempBuf

invoke FloatToStr, KozarezovTanVal, addr KozarezovStrTanVal

invoke wsprintf, addr KozarezovTempBuf, addr KozarezovTanValFmt, addr KozarezovStrTanVal

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovTempBuf

invoke wsprintf, addr KozarezovTempBuf, addr KozarezovResultFmt, addr KozarezovStrRes

invoke szCatStr, addr KozarezovOutBuf, addr KozarezovTempBuf

KozarezovShowResult:

invoke MessageBox, 0, addr KozarezovOutBuf, addr KozarezovTitle, MB\_OK

inc esi

cmp esi, 5

jl KozarezovLoop

invoke ExitProcess, 0

KozarezovMain:

xor esi, esi

KozarezovLoop:

invoke RtlZeroMemory, addr KozarezovOutBuf, sizeof KozarezovOutBuf

invoke RtlZeroMemory, addr KozarezovTempBuf, sizeof KozarezovTempBuf

invoke RtlZeroMemory, addr KozarezovStrRes, sizeof KozarezovStrRes

invoke RtlZeroMemory, addr KozarezovStrTanVal, sizeof KozarezovStrTanVal

finit

invoke FloatToStr, KozarezovDataA[esi\*8], addr KozarezovStrA

invoke FloatToStr, KozarezovDataB[esi\*8], addr KozarezovStrB

invoke FloatToStr, KozarezovDataC[esi\*8], addr KozarezovStrC

invoke FloatToStr, KozarezovDataD[esi\*8], addr KozarezovStrD

fld KozarezovDataA[esi\*8]

fdiv KozarezovTwo

fstp KozarezovTemp

fld KozarezovTemp

fsub KozarezovPiDiv2

fld KozarezovPi

fxch st(1)

fprem

fstp st(1)

fabs

fld KozarezovEpsilon

fxch st(1)

fcomp st(1)

fstsw ax

sahf

fstp st(0)

jbe KozarezovTanError

fld KozarezovTemp

fptan

fstp st(0)

fstp KozarezovTanVal

fld KozarezovDataB[esi\*8]

fsub KozarezovTanVal

ftst

fstsw ax

sahf

jz KozarezovDivZero

fstp KozarezovDenom

fld KozarezovDataC[esi\*8]

fmul KozarezovFour

fadd KozarezovDataD[esi\*8]

fsub KozarezovOne

fstp KozarezovNumer

fld KozarezovNumer

fdiv KozarezovDenom

fstp KozarezovTemp

invoke FloatToStr, KozarezovTemp, addr KozarezovStrRes

jmp KozarezovBuildOutput

KozarezovForceTanError:

jmp KozarezovTanError

KozarezovForceDivZero:

jmp KozarezovDivZero

end KozarezovMain

**Висновки**

Під час виконання даної лабораторної роботи були поглиблені знання команд Асемблера для арифметики з плаваючою комою. Були здобуті практичні навички виконання розрахунків з елементами масивів. Результатом роботи стала програма на мові Асемблера для обчислення значення арифметичного виразу із застосуванням команд математичного співпроцесора (FPU).

Для перевірки коректності роботи програми було заздалегідь виконано контрольні розрахунки для п'яти тестових випадків:

1. Чисельник і знаменник >0
2. Чисельник <0, а знаменник >0
3. Чисельник >0, а знаменник <0
4. Чисельник і знаменник <0
5. Порушення області визначення тангенса (tg(x) при x≈±π/2)

Порівняння контрольних розрахунків із результатами роботи програми виявило незначні відмінності. Ці відмінності обумовлені властивостями представлення чисел з плаваючою комою в комп'ютері та особливостями їх округлення (зокрема, до 6 знаків після коми при виведенні). Незважаючи на це, загалом результати співпадають, що підтверджує правильність логіки та реалізації програми.