**Міністерство освіти і науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра обчислювальної техніки**

**Лабораторна робота № 6**

з дисципліни

«Системне програмування»

на тему

“Обчислення арифметичних виразів і трансцендентних функцій.

Використання співпроцесора іх87”

Виконав:

Перевірив:

студент групи ІП-93 доцент

Марченко Максим Олегович Павлов В.Г.

номер залікової книжки: 9316

Варіант 13

Київ 2020

Мета роботи : Вивчення команд Асемблера для арифметики з плаваючою комою і здобуття навичок виконання розрахунків з елементами масивів.

Для виконання арифметичних операцій з нецілими числами, використовується стек st, що максимально може містити 8 елементів (st(0) - st(7)). Для того, щоб записати певну змінну в стек, використовується команда fld <змінна>,що ставить її у початок стеку, на st(0).

Арифметичні операції з плаваючою комою:

1. Додавання

Дія додавання виконується командою fadd <arg1>, і додає значення першого аргументу і st(0). Результат дії записується в st(0)

1. Віднімання

Дія віднімання виконується командою fsub <arg1>, і виконує дію <arg1> - st(0). Результат дії записується в st(0). Для того, щоб виконати st(0) - <arg1>, використовується схожа команда fsubr

1. Множення

Дія множення виконується командою: fmul <arg> виконує множення <arg> та того, що знаходиться в st(0). Результат записується y st(0)

1. Ділення

Дія ділення виконується командою fdiv <arg> і виконує операцію ділення числа y st(0) на arg. Результат ділення записується в st(0).

Трансцендентні функції

Fрsin, fрcos - вичисляють, відповідно, синус та косинус від st(0). Результат записується в st(0).

Fptan - вичисляє тангенс кута st(0). Результат записується в st(1), а st(0) = 1.Зроблено це було задля того, щоб була можливою співпраця з старішими співпроцесорами.

Fpatan - вичисляє арктангенс кута st(1)/st(0), і результат записує в st(0).

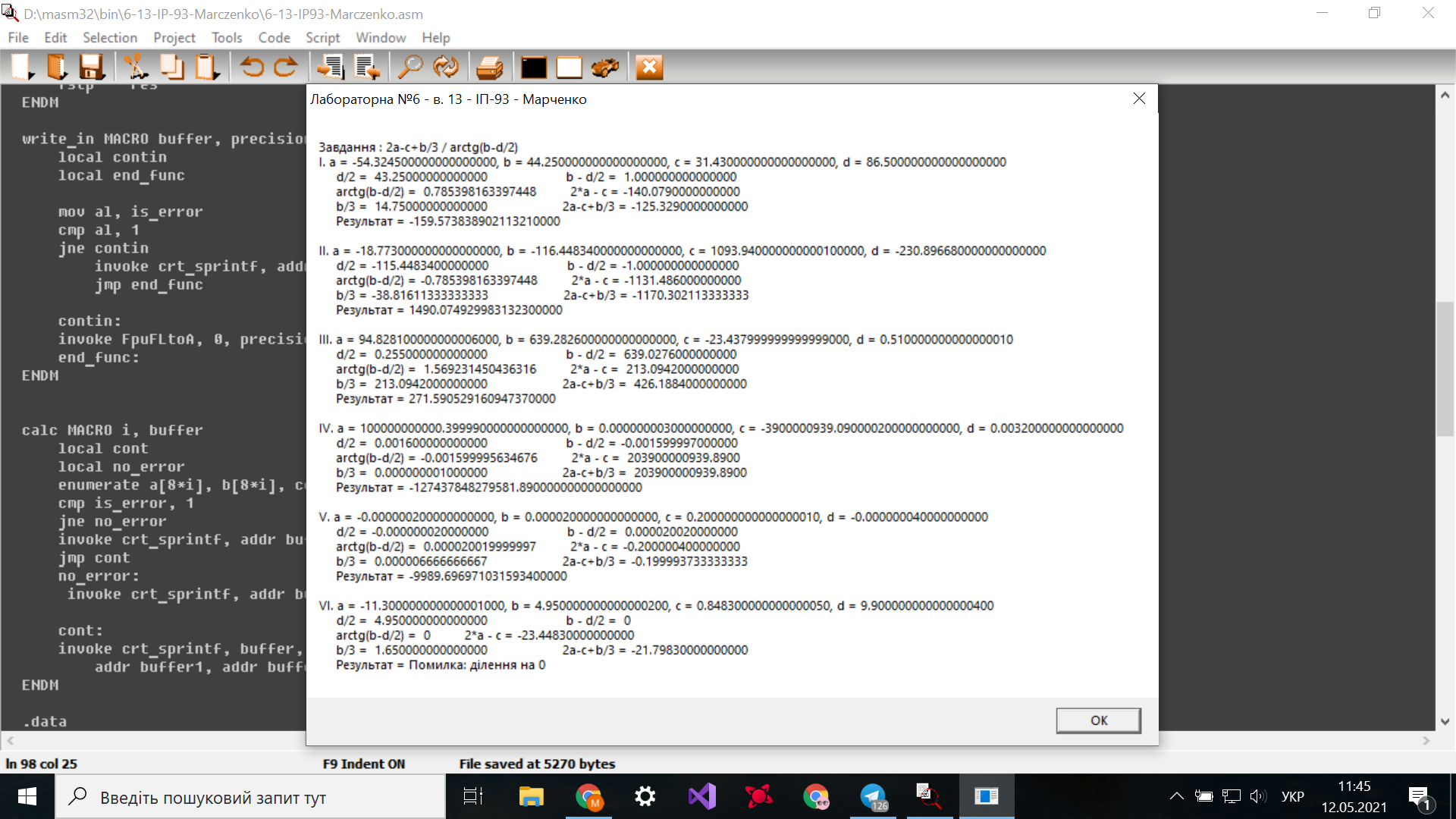
F2XM1 - вичисляє функцію ST(0)=2^ST(0)-1, і записує в st(0).

FYL2X - вичисляє функцію st(1) \* log2(st(0)), і записує в st(0).

Задля отримання інших тригонометричних та логарифмічних функцій необхідно використовувати композицію існуючих функцій та констант.

Константами є: числа 0 та 1, пі, e , log2 (10), log2 (е).

Скріншот виконання програми



На скріншоті зображено виконання програми з 6 наборами вхідних даних, 5 з яких - коректні, 1 - спричинює ділення на 0.

Для знаходження помилки ділення на 0 знаменник виразу порівнюється з нулем перед фінальною дією ділення чисельник на знаменник. Якщо знаменник рівний нулю, то ця дія не виконується, замість її результату виводиться повідомлення про помилку.

Контрольні розрахунки

Розрахунки проводитимуться в тому ж порядку, як і в програмі. Черговість дій:

1. d/2
2. b-d/2
3. arctg(b-d/2)
4. 2a-c
5. b/3
6. 2a-c+b/3
7. (2a-c+b/3) / arctg(b-d/2)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | 1) | 2) | 3) | 4) | 5) | 6) | 7) |
| -54,3245 | 44,25 | 31,43 | 86,5 | 43,25 | 1 | 0,7853981634 | -140,079 | 14,75 | -125,329 | -159,5738389 |
| -18,73 | -116,44834 | 1093,94 | -230,89668 | -115,44834 | -1 | -0,7853981634 | -1131,4 | -38,8161133 | -1170,216113 | 1489,965431 |
| 94,8281 | 639,2826 | -23,438 | 0,51 | 0,255 | 639,0276 | 1,56923145 | 213,0942 | 213,0942 | 426,1884 | 271,5905292 |
| 100000000000 | 0,000000003 | -3900000939 | 0,0032 | 0,0016 | -0,001599997 | -0,001599995635 | 203900000940 | 0 | 203900000940 | -127437848279582 |
| -0,0000002 | 0,00002 | 0,2 | -0,00000004 | -0,00000002 | 0,00002002 | 0,00002002 | -0,2000004 | 0,0000067 | -0,1999937 | -9989,695306 |
| -11,3 | 4,95 | 0,8483 | 9,9 | 4,95 | 0 | 0 | -23,4483 | 1,65 | -21,7983 | #DIV/0! |

Дії для контрольних розрахунків проведені в тій ж послідовності, що й в програмі.

Як видно, результат виконання програми програми та контрольні розрахунки збігаються з точністю до заокруглення для всіх вхідних даних, включаючи випадки помилок, що підтверджує правильність програми.

Висновки

При виконанні роботи я навчився працювати з числами з плаваючою комою, проводити над ними арифметичні операції та застосовувати до них ірраціональні функції. Зокрема, для всіх цих операцій використовується спеціальний стек st, у який і записуються та з якого зчитуються потрібні значення. Функції додавання, віднімання, множення та ділення кожні явно приймають один аргумент, другим аргументом є вершина стеку, тобто st(0), туди ж і записуються результати цих дій. Тригонометричні функції синуса та косинуса неявно приймають в якості аргументу верхівку стеку, куди і записують результат дії функції. Функція тангенсу ж віддає результат виконання у st(1), a st(0) = 1. Функція арктангенсу, логарифму за основою 2 приймають в якості аргументу два значення зі стеку, і результат дії записують у верхівку. Задля отримання інших тригонометричних, а також логарифмічних функцій, використовуються композиції існуючих функцій та констант.

Додаток: Код програми 6-13-IP93-Marczenko

INCLUDE \masm32\include\masm32rt.inc

include \masm32\include\Fpu.inc

includelib \masm32\lib\Fpu.lib

enumerate MACRO a, b, cc, d, res

local fin

local error\_0

mov is\_error, 0

finit

fld d ; st(0) = d

fld constants[0] ; st(0) = 2, st(1) = d

fdiv ; st(0) = d/2

write\_in buffer1, precision

fld b ; st(0) = b, st(1) = d/2

fsubr ; st(0) = b - d/2

write\_in buffer2, precision

fld1 ; st(0) = 1, st(1) = b-d/2

fpatan ; st(0) = arctg(b-d/2)

write\_in buffer3, precision

fld constants[0] ; st(0) = 2, st(1) = arctg(b-d/2)

fld a ; st(0) = a, st(1) = 2, st(2) = arctg(b-d/2)

fmulp st(1), st ; st(0) = 2a, st(1) = arctg(b-d/2)

fld cc ; st(0) = c, st(1) = 2a, st(2) = arctg(b-d/2)

fsub ; st(0) = 2a-c, st(1) = arctg(b-d/2)

write\_in buffer4, precision

fld b ; st(0) = b, st(1) = 2a-c, st(2) = arctg(b-d/2)

fld constants[8] ; st(0) = 3, st(1) = b, st(2) = 2a-c, st(3) = arctg(b-d/2)

fdiv ; st(0) = b/3, st(1) = 2a-c, st(2) = arctg(b-d/2)

write\_in buffer5, precision

fadd ; st(0) = 2a-c+b/3, st(1) = arctg(b-d/2)

write\_in buffer6, precision

fxch st(1) ; st(0) = arctg(b-d/2), st(1) = st(0) = 2a-c+b/3

ftst

fstsw AX

SAHF

JZ error\_0

fdiv ; st(0) = 2a-c+b/3 / arctg(b-d/2)

JMP fin

error\_0:

mov is\_error, 1

fin:

fstp res

ENDM

write\_in MACRO buffer, precision2

local contin

local end\_func

mov al, is\_error

cmp al, 1

jne contin

invoke crt\_sprintf, addr buffer, addr error\_message

jmp end\_func

contin:

invoke FpuFLtoA, 0, precision2, addr buffer, SRC1\_FPU or SRC2\_DIMM

end\_func:

ENDM

calc MACRO i, buffer

local cont

local no\_error

enumerate a[8\*i], b[8\*i], cc[8\*i], d[8\*i], res[8\*i]

cmp is\_error, 1

jne no\_error

invoke crt\_sprintf, addr buffer7, addr error\_message

jmp cont

no\_error:

invoke crt\_sprintf, addr buffer7, addr results, res[8\*i]

cont:

invoke crt\_sprintf, buffer, addr out\_message, a[8\*i], b[8\*i], cc[8\*i], d[8\*i],

addr buffer1, addr buffer2, addr buffer3, addr buffer4, addr buffer5, addr buffer6, addr buffer7

ENDM

.data

header DB "Лабораторна №6 - в. 13 - ІП-93 - Марченко", 0

template DB "Завдання : 2a-c+b/3 / arctg(b-d/2)", 10,

"I. %s", 10,

"II. %s", 10,

"III. %s", 10,

"IV. %s", 10,

"V. %s", 10,

"VI. %s", 0

out\_message DB "a = %.18f, b = %.18f, c = %.18f, d = %.18f", 10,

" d/2 = %s"," b - d/2 = %s", 10,

" arctg(b-d/2) = %s"," 2\*a - c = %s", 10,

" b/3 = %s"," 2a-c+b/3 = %s", 10,

" Результат = %s",10, 0

number DB " DQ = %.25f", 0

results DB "%.18f", 0

error\_message DB "Помилка: ділення на 0", 0

out\_buffer DB 4096 DUP (0)

buff\_ex1 DB 1024 DUP (0)

buff\_ex2 DB 1024 DUP (0)

buff\_ex3 DB 1024 DUP (0)

buff\_ex4 DB 1024 DUP (0)

buff\_ex5 DB 1024 DUP (0)

buff\_ex6 DB 1024 DUP (0)

buff\_size = $ - buff\_ex6

buffer1 DB 0128 DUP (' '), 0

buffer2 DB 0128 DUP (' '), 0

buffer3 DB 0128 DUP (' '), 0

buffer4 DB 0128 DUP (' '), 0

buffer5 DB 0128 DUP (' '), 0

buffer6 DB 0128 DUP (' '), 0

buffer7 DB 0128 DUP (0)

current\_buff DD 0

res DQ 6 DUP (0)

a DQ -54.3245, -18.773, 94.8281, 100000000000.4, -0.0000002, -11.3

b DQ 44.25, -116.44834, 639.2826 , 0.000000003 , 0.00002, 4.95

cc DQ 31.43, 1093.94, -23.438, -3900000939.09 , 0.2, 0.8483

d DQ 86.5, -230.89668, 0.51 , 0.0032 , -0.00000004, 9.9

constants DQ 2.0, 3.0

temp DQ 0.0

is\_error DB 0

precision DD 15 ;; кількість цифр після коми

.code

start:

mov esi, 0

mov current\_buff, offset buff\_ex1

cycle:

calc esi, current\_buff

add current\_buff, buff\_size

inc esi

cmp esi, 6

JB cycle

invoke crt\_sprintf, addr out\_buffer, addr template,

addr buff\_ex1,

addr buff\_ex2,

addr buff\_ex3,

addr buff\_ex4,

addr buff\_ex5,

addr buff\_ex6

invoke MessageBox, 0, addr out\_buffer, addr header, MB\_OK

invoke ExitProcess, 0

END start