Асатрян Самвел Ваганович; БПИ 193-1; Вариант 029;

**Микропроект 1**

**Пояснительная записка**

**Текст задания:** Разработать программу, вычисляющую с помощью степенного ряда с точностью не хуже 0,05% значение функции *tan(х)* для заданного параметра *x*(использовать FPU)

**Описание расчётного алгоритма программы:** Алгоритм работы состоится в разложении функций sin(x) и cos(x) в ряд Маклорена, и на каждом шаге взять их деление. Если же это значение отличается от истинного значения меньше чем 0,05%, то программа останавливается и выводится результат.

**Список используемых источников для решения задачи:**

Учебник fasm: <http://flatassembler.narod.ru/fasm.htm#2-1-13>

Система команд сопроцессора: <https://prog-cpp.ru/asm-coprocessor-command/>

Более подробное описание команд: <https://www.club155.ru/x86cmdfpu/>

Материалы с SoftCraft: <http://www.softcraft.ru/edu/comparch/practice/asm86/05-fpu/>

**Описание файлов в папке проекта:**

1. Micro\_1\_Asatryan.docx – исходник для пояснительной записки.
2. Micro\_1\_Asatryan.pdf – пояснительная записка.
3. tanCalc.ASM – файл с исходным кодом программы, пригодным для компиляции и запуска.
4. tanCalc.EXE – готовая скомпилированная программа.
5. positive.png – скриншот работы программы с положительным значением
6. negative.png – скриншот работы программы с отрицательным значением

**Приложение**

Код программы

format PE console

entry Start

include 'win32a.inc'

section '.data' data readable writable

formatin db "%lf",0

formatout db "Program calculated answer is: %lf", 10, 0

formatoutr db "Real Answer is: %lf", 10, 0

ask\_x DB "Enter x: ", 0

it db 0

sinSum dq ? ;Current sum of sin

cosSum dq ? ;Current sum of cos

tanSum dq ?

realTan dq ? ;real calculated tan

;some constants

curIter dd 0

pos dd 1

powIter dd 0

temp dq ?

two dd 2

minusOne dd -1

pi dq ?

x dq 0 ;argument of tg being inputed

wr dq 0.0005 ;difference

nfactorial dq 1

tst dq 4.0

section '.code' code readable executable

;ñåêöèÿ äëÿ êîäà

proc Start

FINIT

push ask\_x

call [printf]

add esp, 4

;reading tg

push x

push formatin

call [scanf]

;makes x to number in interval (-Pi/2, Pi/2)

cmpPi:

FLDPI

FST [pi]

FLD [x]

FXCH

FIDIV [two]

FCOMIP st1

jb minusPi

ja endCmp

minusPi:

FLD [x]

FSUB [pi]

FST [x]

jmp cmpPi

endCmp:

cmpPiL:

FLDPI

FST [pi]

FLD [x]

FXCH

FIDIV [two]

FIMUL [minusOne]

FCOMIP st1

ja plusPi

jb endCmp1

plusPi:

FLD [x]

FADD [pi]

FST [x]

jmp cmpPiL

endCmp1:

;end of that process

;calculating real tan and max difference(wr)

FPTAN

FXCH

FST [realTan]

FABS

FMUL [wr]

FST [wr]

;loop for comparing current tan(x) value to real and reruning nextTan process to move to the next iteration

iter:

FINIT

call nextTan

FLD [tanSum]

FSUB [realTan]

FABS

FLD [wr]

FCOMIP st1

jb iter

invoke printf, formatout, dword[tanSum], dword[tanSum+4]

invoke printf, formatoutr, dword[realTan], dword[realTan+4]

Exit

call [getch]

push 0

call [ExitProcess]

endp

;process to calculate nex sin and add it to sinSum

proc nextSin

mov eax, [curIter]

and ax, 1

mov [pos], 1

jz keepPositive

makeNegative:

mov [pos], -1

keepPositive:

fld [x]

mov esi, 1

pow:

mov eax, [curIter]

add eax, eax

add eax, 1

cmp esi, eax

jge powEnd

fmul [x]

add esi, 1

jge pow

powEnd:

mov ecx, eax

call factorial

mov dword[nfactorial], eax

fimul [pos]

fidiv dword[nfactorial]

fadd [sinSum]

fst [sinSum]

ret

endp

;process to calculate nex cos and add it to cosSum

proc nextCos

mov eax, [curIter]

and ax, 1

mov [pos], 1

jz keepPositive1

makeNegative1:

mov [pos], -1

keepPositive1:

fld [x]

mov esi, 1

pow1:

mov eax, [curIter]

add eax, eax

cmp eax, 0

je resOne

cmp esi, eax

jge powEnd1

fmul [x]

add esi, 1

jge pow1

powEnd1:

mov ecx, eax

call factorial

mov dword[nfactorial], eax

fimul [pos]

fidiv dword[nfactorial]

fadd [cosSum]

fst [cosSum]

ret

resOne:

fdiv [x]

mov ecx, eax

call factorial

mov dword[nfactorial], eax

fimul [pos]

fidiv dword[nfactorial]

fadd [cosSum]

fst [cosSum]

ret

endp

;process for calculating next tan(x) value

proc nextTan

call nextSin

call nextCos

fld [sinSum] ;dividing sin(x) and cos(x)

fdiv [cosSum]

fst [tanSum]

mov eax, [curIter]

add eax, 1

mov [curIter], eax

ret

endp

;process for calculating factorial(n)

proc factorial

mov eax, ecx

loops:

cmp ecx, 0

je retOne

cmp ecx, 1

jle endof

sub ecx, 1

mul ecx

jmp loops

retOne:

mov eax, 1

endof:

ret

endp

section '.idata' import data readable ; èìïîðòèðîâàííûå ìåòîäû

library kernel, 'kernel32.dll',\

msvcrt, 'msvcrt.dll'

import kernel,\

ExitProcess, 'ExitProcess'

import msvcrt,\

printf, 'printf',\

scanf, 'scanf',\

getch, '\_getch'