

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Департамент программной инженерии

Пояснительная записка

Исполнитель:
студент группы **БПИ199**
Шарипов Сардор Уткирович

Москва 2020

1. Текст задания

Разработать программу, вычисляющую с помощью степенного ряда с точностью не хуже 0,05% значение функции гиперболического тангенса $th(x) = (e^x - e^{-x})/(e^x + e^{-x})$ для заданного параметра x с точностью 0.05 % (использовать FPU)

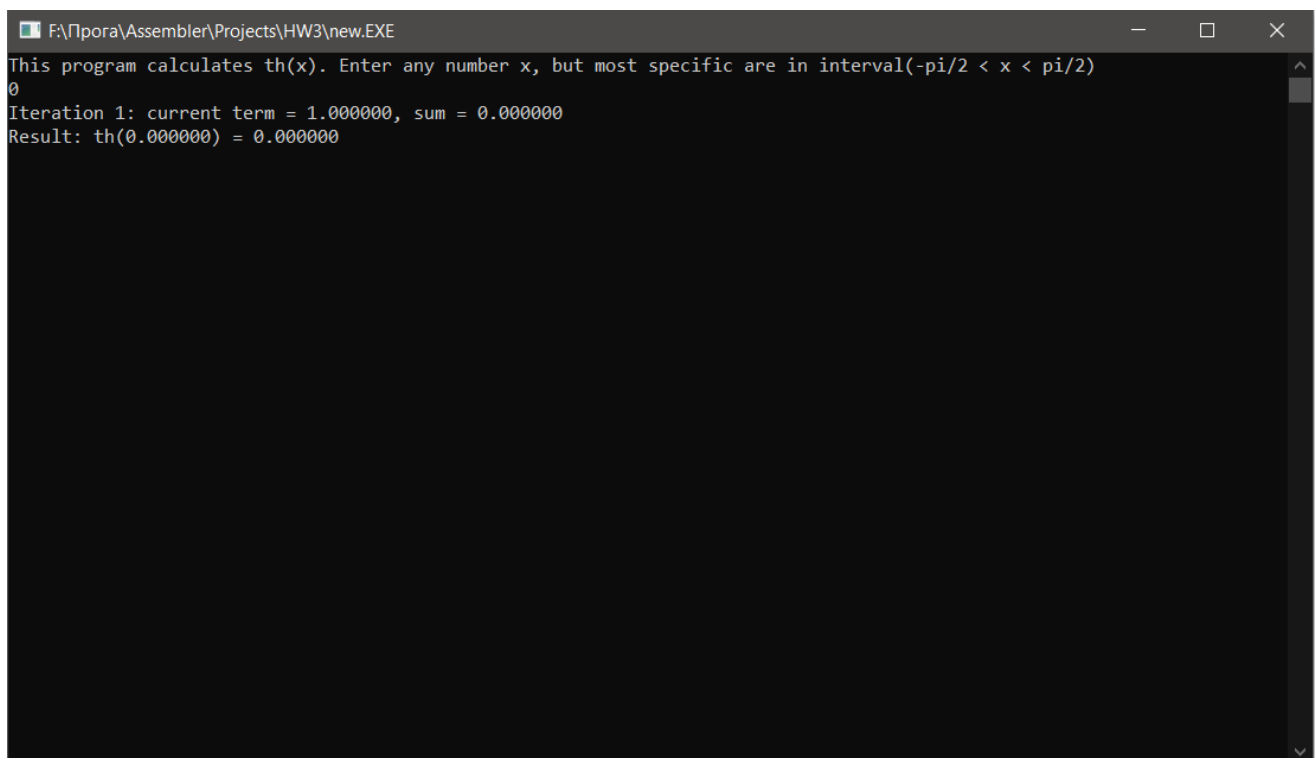
2. Описание алгоритма

- 1) Пользователь вводит число $x < \left|\frac{\pi}{2}\right|$, проверяется на корректность.
- 2) Вычисляем $th(x) = \sum_{n=1}^{\infty} \frac{2^{2n}(2^{2n}-1)B_{2n}x^{2n-1}}{(2n)!}$ где B_{2n} – числа Бернулли.
- 3) $B_{2n} = \frac{2(-1)^{n+1}(2k)!}{(2\pi)^{2n}}\zeta(2k)$ где $\zeta(2k) = \sum_{n=1}^{\infty} \frac{1}{n^k}$
- 4) Проверяется на точность.
- 5) Выводится результат.

3. Испытания

3.1. Особые точки

$X = 0$, на калькуляторе – 0, программа:



```
F:\Прора\Assembler\Projects\HW3\new.EXE
This program calculates th(x). Enter any number x, but most specific are in interval(-pi/2 < x < pi/2)
0
Iteration 1: current term = 1.000000, sum = 0.000000
Result: th(0.000000) = 0.000000
```

$X \geq \pi/2$, на калькуляторе – 1, программа:

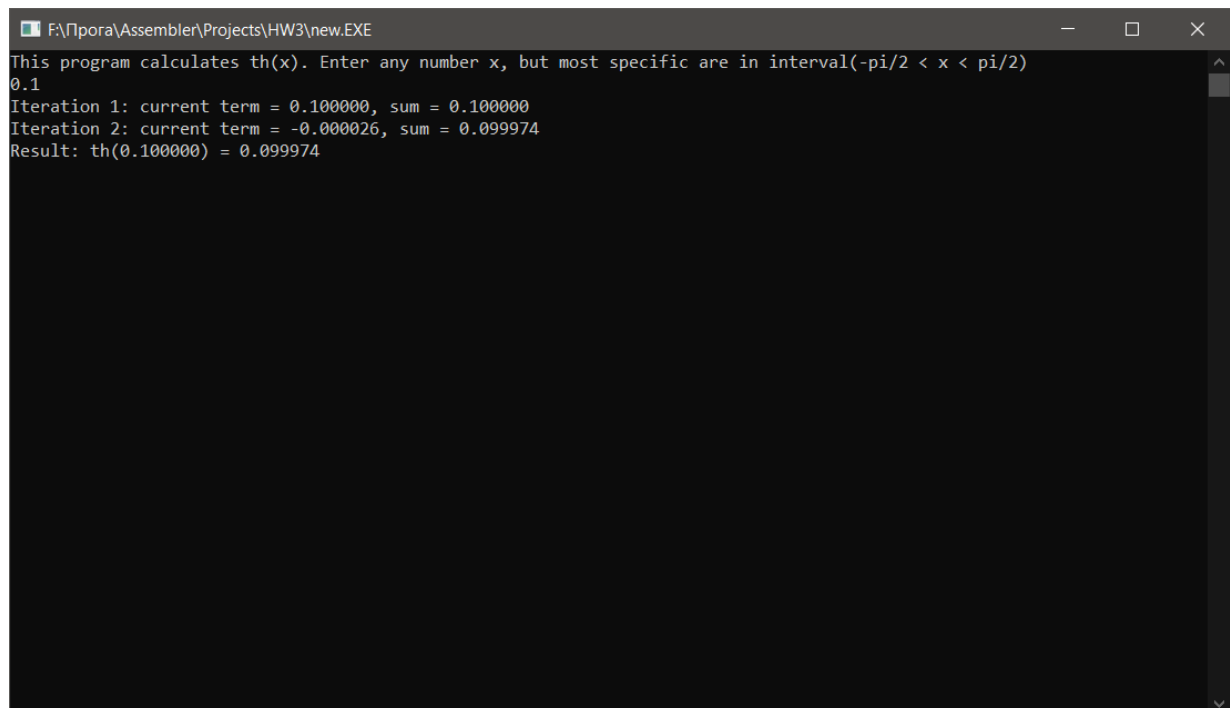
```
F:\Ппора\Assembler\Projects\HW3\new.EXE
This program calculates th(x). Enter any number x, but most specific are in interval(-pi/2 < x < pi/2)
2
Iteration 1: current term = 1.000000, sum = 1.000000
Result: th(2.000000) = 1.000000
```

$X \leq \pi/2$, на калькуляторе – (-1), программа:

```
F:\Ппора\Assembler\Projects\HW3\new.EXE
This program calculates th(x). Enter any number x, but most specific are in interval(-pi/2 < x < pi/2)
-10
Iteration 1: current term = -1.000000, sum = -1.000000
Result: th(-10.000000) = -1.000000
```

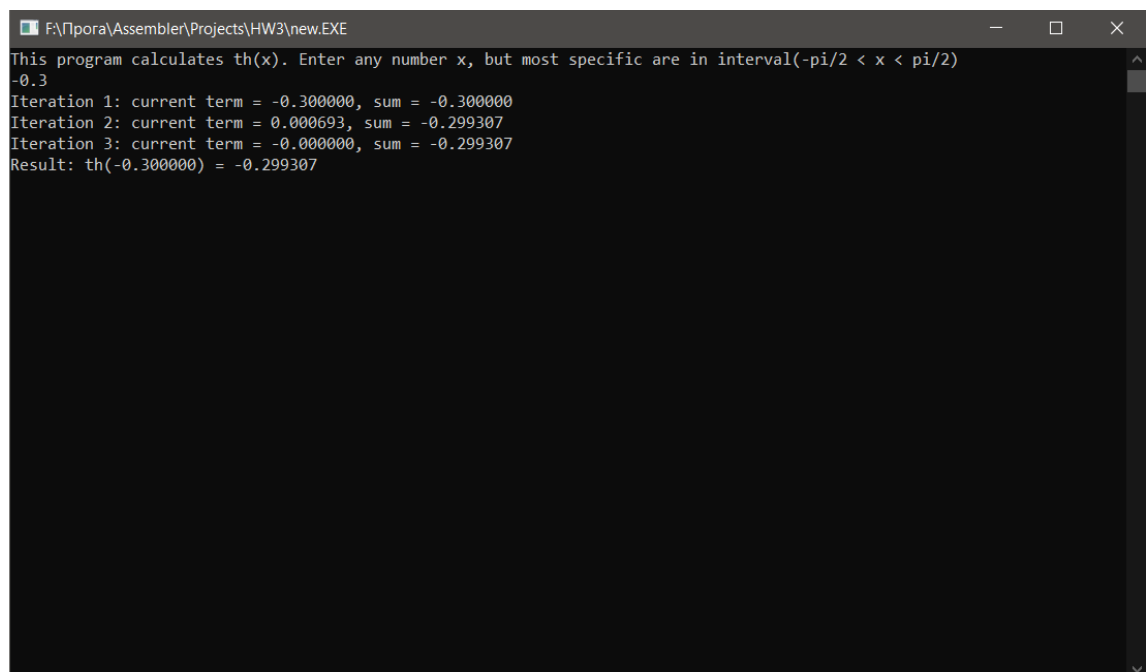
3.2. На промежутке $-\pi/2 < x < \pi/2$

$X = 0.1$, на калькуляторе - 0.099668, программа:



```
F:\Ппора\Assembler\Projects\HW3\new.EXE
This program calculates th(x). Enter any number x, but most specific are in interval(-pi/2 < x < pi/2)
0.1
Iteration 1: current term = 0.100000, sum = 0.100000
Iteration 2: current term = -0.000026, sum = 0.099974
Result: th(0.100000) = 0.099974
```

$X = -0.3$, на калькуляторе – (-0.291313), программа:



```
F:\Ппора\Assembler\Projects\HW3\new.EXE
This program calculates th(x). Enter any number x, but most specific are in interval(-pi/2 < x < pi/2)
-0.3
Iteration 1: current term = -0.300000, sum = -0.300000
Iteration 2: current term = 0.000693, sum = -0.299307
Iteration 3: current term = -0.000000, sum = -0.299307
Result: th(-0.300000) = -0.299307
```

$X = 0.5$, на калькуляторе - 0.493117, программа:

```
F:\Ппора\Assembler\Projects\HW3\new.EXE
This program calculates th(x). Enter any number x, but most specific are in interval(-pi/2 < x < pi/2)
0.5
Iteration 1: current term = 0.500000, sum = 0.500000
Iteration 2: current term = -0.003208, sum = 0.496792
Result: th(0.500000) = 0.496792
```

$X = -0.7$, на калькуляторе – (-0.684368), программа:

```
F:\Ппора\Assembler\Projects\HW3\new.EXE
This program calculates th(x). Enter any number x, but most specific are in interval(-pi/2 < x < pi/2)
-0.7
Iteration 1: current term = -0.700000, sum = -0.700000
Iteration 2: current term = 0.008803, sum = -0.691197
Iteration 3: current term = -0.000000, sum = -0.691197
Result: th(-0.700000) = -0.691197
```

4. Источники

https://en.wikipedia.org/wiki/Bernoulli_number Числа Бернулли

https://en.wikipedia.org/wiki/Riemann_zeta_function#The_functional_equation Зета функция Римана

5. Текст программы

5.1. MacroHW.INC

```
6. ; Макрос для вывода чисел с плавающей запятой.
7. macro PrintFloat string, [args] {
8.     reverse
9.     push dword[args + 4]
10.    push dword[args]
11. common
12.    push string
13.    call [printf]
14.}
15.
16.; Макрос, инкрементирующий значение переменной word.
17.macro Inc value {
18.    mov cx, [value]
19.    inc cx
20.    mov [value], cx
21.}
22.
23.; Макрос, инкрементирующий значение переменной dword.
24.macro Inc2 value {
25.    mov ecx, [value]
26.    inc ecx
27.    mov [value], ecx
28.}
29.
30.; Макрос, копирующий значение одного числа с плавающей запятой в другое.
31.macro CopyFloat to, from {
32.    mov eax, dword[from]
33.    mov ebx, dword[from + 4]
34.    mov dword[to], eax
35.    mov dword[to + 4], ebx
36.}
37.; Макрос для подсчета чисел Бернулли B_2n
38.;  $B_{2n} = (-1)^{(n+1)} * 2 / (2\pi)^{2n} * E(2n)$ 
39.;  $E(2n)$  = зета функция Эйлера
40.;  $E(2k) = \text{Sum } (1/n)^{2k}$ 
41.; здесь в коде n как 2n в формуле
42.macro bernoulli result,n, one,minOne, two, inRes, inSum , pi, temp{
43.
44.    mov cx, [n]
45.    fld [two]
46.    b:
47.        fdiv [two]
48.    loop b ;  $2/2^{2n}$ 
```

```

49.
50. mov cx, [n]
51. c:
52.     fdiv [pi]
53.     loop c ; 2/(2pi)^2n
54.
55. fstp [result]
56.
57. mov cx, [n]
58. d:
59.     mov bx, cx
60.
61.     fld [one]
62.     mov cx, [n]
63.     e:
64.         fdiv [temp]
65.         loop e
66.         fstp [inSum] ; 1/n^2k
67.
68.         fld [temp]
69.         fsub [one]
70.         fstp [temp]
71.
72.         fld [inSum]
73.         fadd [inRes]
74.         fstp [inSum] ; Сумма этих чисел
75.         mov cx, bx
76.     loop d
77.     fld [result]
78.     fmul [inSum]
79.     fstp [result]
80. ; Смотря на n берем знак числа бернулли
81.     mov ax, [n]
82.     fild [n]
83.     fidiv word[two]
84.     fiadd word[one]
85.     fistp [n]
86.     cmp ax, [n]
87.     jpo odd
88.     mov [n], ax
89.
90. odd:
91.     fld [result]
92.     fchs
93.     fstp [result]
94. ; Это я темп на место возвращаю
95.     mov cx, [n]
96.     fld [temp]
97.     f:
98.         fadd [one]
99.         loop f

```

```
100.     fstp [temp]
101.     }
```

5.2. Main Program

```
format PE Console
entry start

include 'F:\Ппора\Assembler\INCLUDE\WIN32A.INC'
include 'F:\Ппора\Assembler\Projects\HW3\MacroHW.INC'

section '.data' data readable writable

    enterStr db 'This program calculates th(x). Enter any number x, but most specific
are in interval(-pi/2 < x < pi/2)', 10, 0
    numberStr db '%lf', 0
    debugStr db '%lf', 10, 0
    iterationStr db 'Iteration %d: ', 0
    currentValuesStr db 'current term = %lf, sum = %lf', 10, 0
    resultStr db 'Result: th(%lf) = %lf', 10, 0

    x dq ? ; вводимое число
    sqrX dq ? ; квадрат x

    term dq 1.0 ; текущий член ряда

    nextTerm dq ? ; последующий член ряда

    factorialCounter dw 4

    iterationCounter dd 1
    subpow dd 2 ; 2^2n
    subpow2 dd 1 ; 2^2n-1

    sum dq 0.0 ; сумма ряда
    berRes dq ? ; число Бернулли
    one dq 1.0
    minOne dq -1.0
    two dq 2.0
    inRes dq 0.0 ; промежуточные
    inSum dq 0.0 ; переменные
    pi dq 3.141592
    temp dq 4.0
    epsilon dq 0.0005

section '.code' code readable executable

start:

    FINIT
```



```

        call  inputNumber
        call  mainLoop
        call  outputResult

finish:
        push  0
        invoke getch
        call  ExitProcess

zeroF:
        fld  [term]
        fstp [nextTerm]
        call outputResult
        call finish
        ret

minPiF:
        fld  [one]
        fchs
        fst  [nextTerm]
        fstp [sum]
        call outputResult
        call finish
        ret

plusPiF:
        fld  [one]
        fst  [nextTerm]
        fstp [sum]
        call outputResult
        call finish
        ret

inputNumber:

        invoke printf, enterStr
        invoke scanf, numberStr, x
        fld  [sum]
        fld  [x]
        ; производим сравнения
        fcomp
        fstsw ax
        sahf
        jz  zeroF ;если 0 то функция равен 0

        fld  [pi]
        fdiv [two]
        fld  [x]

```

```

fcompp
fstsw ax
sahf
jae plusPiF ;если больше pi/2 то 1
clc

```

```

fld [pi]
fdiv [two]
fchs
fld [x]
fcompp
fstsw ax
sahf
jbe minPiF ;если меньше pi/2 то -1

```

```

fld [x]
fmul [x]
fstp [sqrX]; находим x^2

```

```

fld [term]
fmul [x]
fstp [term]; первый член ряда равен x

```

```

fld [term]
fstp [sum]; сумма тоже x

```

mainLoop:

```

invoke printf, iterationStr,[iterationCounter]
PrintFloat currentValuesStr, term, sum
Inc2 iterationCounter

```

```

fld [subpow]
fmul [subpow]
fstp [subpow]; получим квадрат (2^2n)в каждой итерации

```

```

fld [subpow]
fstp [subpow2]; получим (2^2n-1)
Inc2 subpow2

```

```

fld [term]
fmul [sqrX]
fstp [nextTerm] ; получим 2^2n-1 в каждой итерации

```

```

bernuolli berRes,factorialCounter, one,minOne,two,inRes, inSum,pi,temp
fld [nextTerm]
fmul [berRes] ; умножаем
fmul [subpow] ; полученные
fmul [subpow2] ; члены
fstp [nextTerm] ;

```

```

        fld [temp]
        fadd [two]      ; это как factorial iterator только qword
        fstp [temp]

        fld [sum]
        fadd [nextTerm] ; суммируем
        fstp [sum]

        fld [sum]
        fabs
        fmul [epsilon] ; сравниваем с эпсилон
        fld [nextTerm]
        fcompp
        fstsw ax
        sahf
        jb outputResult

        Inc factorialCounter
        Inc factorialCounter

        CopyFloat term, nextTerm
        jmp mainLoop
ret
outputResult:
        invoke printf, iterationStr,[iterationCounter] ;результат программы
        PrintFloat currentValuesStr, nextTerm, sum
        PrintFloat resultStr, x, sum
        call finish
ret

```

```

section '.idata' import data readable

```

```

library kernel, 'kernel32.dll',\
        msvcrt, 'msvcrt.dll'

```

```

import kernel,\
        ExitProcess, 'ExitProcess',\
        GetProcessHeap, 'GetProcessHeap',\
        HeapAlloc, 'HeapAlloc'

```

```

import msvcrt,\
        printf, 'printf',\
        scanf, 'scanf',\
        getch, '_getch'

```