

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

Факультет компьютерных наук
Департамент программной инженерии

**КОНСОЛЬНОЕ ПРИЛОЖЕНИЕ, ВЫЧИСЛЯЮЩЕЕ С ПОМОЩЬЮ
СТЕПЕННОГО РЯДА ЗНАЧЕНИЕ ФУНКЦИИ ГИПЕРБОЛИЧЕСКОГО
СИНУСА**

Пояснительная записка

Исполнитель:

Студентка группы БПИ195

_____/Зубарева Н.Д./

«30» октября 2020 г.

Оглавление

1. Текст задания.....	2
2. Применяемые расчетные методы	3
2.1. Теория решения задания	3
2.2. Ввод входных данных	3
2.3. Вывод данных.....	3
3. Тестирование программы.....	4
2.4. Корректные значения	4
2.5. Некорректные значения	5
4. Список литературы.....	6
5. Приложение кода.....	7

1. Текст задания

Разработать программу, вычисляющую с помощью степенного ряда с точностью не хуже 0,1% значение функции гиперболического синуса $\text{sh}(x) = (e^x - e^{-x})/2$ для заданного параметра x (использовать FPU)

2. Применяемые расчетные методы

2.1. Теория решения задания

По условию требуется вычислять значение гиперболического синуса $\text{sh}(x) = (e^x - e^{-x})/2$ с помощью степенного ряда. Согласно [2], гиперболический синус представляется в виде следующего степенного ряда:

$$\sinh x = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \frac{x^7}{7!} + \dots = \sum_{n=0}^{\infty} \frac{x^{2n+1}}{(2n+1)!}$$

Члены степенного ряда вычисляются по очереди, и последний вычисленный прибавляется к сумме. Чтобы добиться требуемой точности, вычисляется и сравнивается с требуемым процентом точности (0.1%) разность последних полученных членов ряда. Вычисление в теории прекращается, когда два последних значения отличаются менее, чем на 0.001 результата суммы. На практике в нашей программе мы будем сравнивать отношение разности и результата с нулем, так как иначе точность будет меньше требуемой из-за сопутствующих погрешностей.

2.2. Ввод входных данных

Аргумент функции считывается из командной строки с помощью функции `scanf`, использующей форматирование “%lf”, чтобы аргумент обрабатывался как число с плавающей точкой.

Проверяется корректность входных данных, в случае неверного ввода выводится сообщение об ошибке и работа программы завершается. В случае корректного ввода вычисляется гиперболический синус, но также при больших по модулю значениях аргумента выводится 1.#INF, так как $\sinh(x)$ стремится к бесконечности при x стремящемся к бесконечности.

2.3. Вывод данных

Конечный результат работы функции выводится с помощью `printf` с форматированием `%.15g`, так как это точность чисел с плавающей точкой в знаках после запятой, поэтому больше выводить нет смысла.

3. Тестирование программы

Программа начинает свою работу с предложения пользователю ввести аргумент вычисления функции (рисунок 1).



Рисунок 1 Начало работы программы

2.4. Корректные значения

Программа осуществляет вычисление при вводе корректных данных, при значениях меньше значения бесконечности для чисел с плавающей точкой она выводит числовые значения гиперболического синуса (рисунок 2, рисунок 3), в ситуациях, когда аргумент x слишком большой и $\sinh(x) = \inf$, выводится `1.#INF` (рисунок 4, рисунок 5).

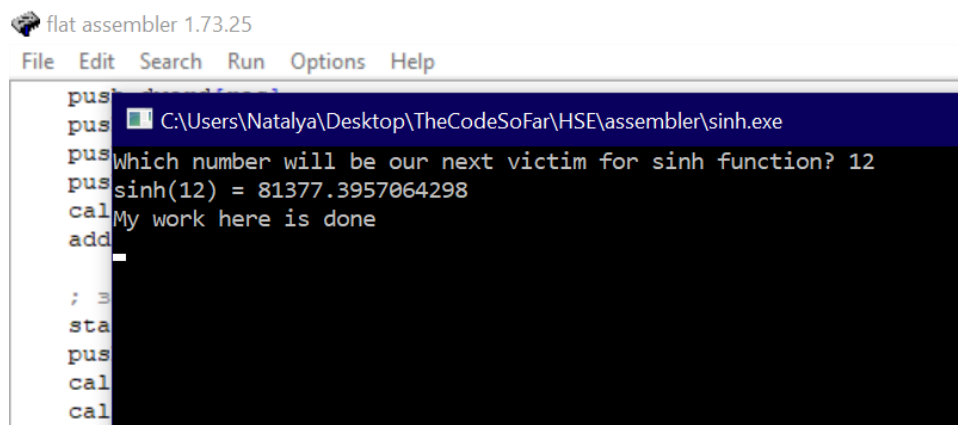


Рисунок 2 Работа программы при $x = 12$

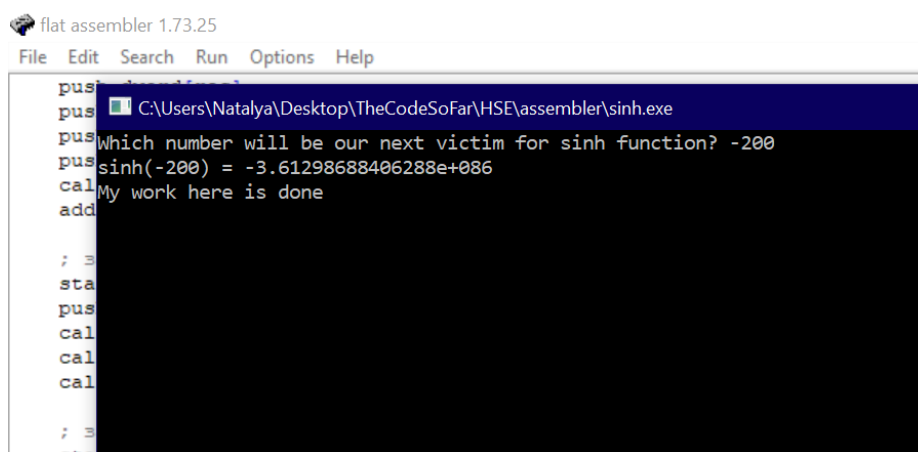


Рисунок 3 Работа программы при $x = -200$

```

flat assembler 1.73.25
File Edit Search Run Options Help

; printf работает только с float64, а push - нет, выводим результат
push dword[res+4]
push dword[res]
push dword[x+4]
push dword[0]
push answer
call [printf]
add esp, 4

; завершаем работу
start_exit:
push exit_code
call [printf]
call [getch]

```

Which number will be our next victim for sinh function? -400000
sinh(-400000) = -1.#INF
My work here is done

Рисунок 4 Работа программы при $x = -400000$

```

flat assembler 1.73.25
File Edit Search Run Options Help

; printf работает только с float64, а push - нет, выводим результат
push dword[res+4]
push dword[res]
push dword[x+4]
push dword[0]
push answer
call [printf]
add esp, 4

; завершаем работу
start_exit:
push exit_code
call [printf]
call [getch]

```

Which number will be our next victim for sinh function? 1000
sinh(1000) = 1.#INF
My work here is done

Рисунок 5 Работа программы при $x = 1000$

2.5. Некорректные значения

Программа также обрабатывает случаи ввода некорректных данных, например не чисел. В такой ситуации в консоль выводится сообщение об ошибке ввода, и программа завершает свою работу (рисунок 6).

```

flat assembler 1.73.25
File Edit Search Run Options Help

format PE console
entry start
include 'win32a.inc'

section '.code' code readable executable
sinh:
    finit
    fldl
    fstp Qword[answer]
    ; заходим в цикл
    mov ecx, 0
    sinh_loop:
        inc ecx
        ; домножаем
        fld Qword[x]

```

Which number will be our next victim for sinh function? ffffffff
Wrong input - not a float
My work here is done

Рисунок 6 Сообщение об ошибке ввода при некорректных данных

4. Список литературы

- [1] Инструкция по составлению пояснительной записки [Электронный ресурс]. //URL: <http://softcraft.ru/edu/comparch/tasks/mp01/> (Дата обращения: 20.10.2020, режим доступа: свободный)
- [2] Статья «Hyperbolic functions» Wikipedia.org //URL: https://en.wikipedia.org/wiki/Hyperbolic_functions (Дата обращения: 20.10.2020, режим доступа: свободный)
- [3] Руководство по синтаксису FASM [Электронный ресурс]. //URL: <http://flatassembler.narod.ru/fasm.htm> (Дата обращения: 20.10.2020, режим доступа: свободный)

5. Приложение кода

```
format PE console
entry start
include 'win32a.inc'

section '.code' code readable executable
sinh:
    finit
    fld1
    fstp Qword[delta];слово на 64 бит

    ; заходим в расчет собственно гиперболического синуса
    mov ecx, 0
sinh_loop:
    inc ecx
    ; домножаем delta на x / i
    fld Qword[x]
    mov [i], ecx
    fidiv dword[i]
    fmul Qword[delta]
    fstp Qword[delta]

    ; пропускаем четные степени i, потому что синус раскладывается
    ; в сумму нечетных степеней
    mov eax, ecx
    and eax, 1
    cmp eax, 0
    je sinh_loop

    ; прибавляем delta к res
    fld Qword[delta]
    fadd Qword[res]
    fstp Qword[res]
```


; сравниваем то, какой процент разницы между последними составляет от результата 0%.

; в условии 0.001, а не 0, но тогда точность будет на самом деле меньше, чем мы хотим

; $\text{abs}(\text{delta} / \text{res}) \leq 0$ (или 0.001 но тогда будет нехорошая точность)

fldz ;делаем 64-битное слово для fraction

fld Qword[delta]

fdiv Qword[res]

fabs

fcompp

fstsw ax

sahf

jbe sinh_end

; на всякий случай делаем лимит на количество итераций,

; чтобы точно избежать бесконечных циклов

cmp ecx, 69420

j1 sinh_loop

sinh_end:

ret

start:

; создаем float64 в scanf, считываем аргумент для синуса

push beginStr

call [printf]

add esp, 4

push x

push scanfFormat

call [scanf]

add esp, 8

cmp eax, 1

jne start_wrongInput ; проверяем на правильность ввода. Если неправильный
- сообщаем и завершаем

start_scanfEnd:

call sinh ; вызываем расчет

```

; printf работает только с float64, а push - нет, выводим результат
push dword[res+4]
push dword[res]
push dword[x+4]
push dword[x]
push answerStr
call [printf]
add esp, 20

```

```

; завершаем программу
start_exit:
push exitStr
call [printf]
call [getch]
call [exit]

```

```

; завершаем программу при некорректном вводе
start_wrongInput:
push errorStr
call [printf]
add esp, 4
jmp start_exit

```

```

section '.data' data readable writable

```

```

scanfFormat: db '%lf',0
answerStr: db 'sinh(%g) = %.15g',10,0 ; выводим 15 знаков, потому что у
даблов точность 15 десятичных знаков
beginStr: db 'Which number will be our next victim for sinh function? ',0
errorStr: db 'Wrong input - not a float',10,0
exitStr: db 'My work here is done',10,0
i: dd 0
x: dq 0
res: dq 0
delta: dq 0

```

```

section '.idata' import code readable

```

```
library msvcrt, 'msvcrt.dll'  
import msvcrt, printf, 'printf', scanf, 'scanf', exit, '_exit', getch,  
'_getch'
```