НИУ ВШЭ

Факультет компьютерных наук

Программная инженерия

Программа для вычисления гиперболического косинуса с помощью степенного ряда

Куприхин Дмитрий Алексеевич, БПИ199

1. Оглавление

1. Текст программы

Вариант 14. Разработать программу, вычисляющую с помощью степенного ряда с точностью не хуже 0,05% значение функции гиперболического косинуса ch(x)=(e^x+e^(-x))/2 для заданного параметра x (использовать FPU).

2. Применяемый метод

Значение гиперболического косинуса для аргумента x может быть выражено через ряд Тейлора[1]:

$$\cosh(x) = \sum_{n=0}^{\infty} \frac{x^{2n}}{(2n)!}$$

Список литературы

1. Abramowitz, M., & Stegun, I. A. (1972). *Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables.* New York: Dover Publications.

Приложение 1. Текст файла hw3Macro.inc

```
; Макрос для вывода чисел с плавующей запятой.
macro PrintFloat string, [args] {
 reverse
  push dword[args + 4]
  push dword[args]
 common
  push string
  call [printf]
}

; Макрос, инкрементирующий значение переменной.
macro Inc value {
 mov ecx, [value]
 inc ecx
 mov [value], ecx
}

; Макрос, копирующий значение одного числа с плавующей
запятой в другое.
macro CopyFloat to, from {
 mov eax, dword[from]
 mov ebx, dword[from + 4]
 mov dword[to], eax
 mov dword[to + 4], ebx
}
```

Приложение 2. Текст файла hw3.asm

```
; Программа для вычисления гиперболического косинуса (вариант 14) с помощью
; степенного ряда с точностью не хуже 0.05%.
; cosh(x) = 1 + (x^2 / 2!) + (x^4 / 4!) + ...
; Автор: Куприхин Дима, БПИ199

format PE Console
entry start

include 'win32a.inc'
include 'hw3Macro.inc'

section '.data' data readable writable

        enterStr db 'This program calculates cosh(x). Enter x', 10, 0
        numberStr db '%lf', 0
        iterationStr db 'Iteration %d: ', 0
        currentValuesStr db 'current term = %lf, sum = %lf', 10, 0
        resultStr db 'Result: cosh(%lf) = %lf', 10, 0
        ; Вводимый пользователем аргумент вычисляемой функции
        x dq ?
        ; Квадрат введеного значения.
        sqrX dq ?
        ; Переменная, хранящая значение текущего члена ряда.
        term dq 1.0
        ; Переменная, в которой вычисляется значение следущего члена ряда.
        nextTerm dq ?
```

```
        ; Счетчик значения факториала.
        factorialCounter dd 1
        ; Счетчик итерации.
        iterationCounter dd 1
        ; Переменная, хранящая сумму ряда.
        sum dq 1.0
        ; Переменная, хранящая точность, с которыми
производятся вычисления.
        epsilon dq 0.0005

section '.code' code readable executable

        start:
                ; Вводим аргумент функции.
                invoke printf, enterStr
                invoke scanf, numberStr, x

                FINIT
                ; Считаем x^2, так как каждый следующий член
ряда в
                ; числителе увеличивается на x^2
                fld [x]
                fmul [x]
                fstp [sqrX]

        mainLoop:

                ; Выводим промежуточный результат.
                invoke printf,
iterationStr,[iterationCounter]
                PrintFloat currentValuesStr, term, sum
```

```
                Inc iterationCounter

                ; Вычисляем следующий член:
                ; nextTerm = term; nextTerm *= x^2;
                fld [term]
                fmul [sqrX]
                ; nextTerm /= factorialCounter++;
                fidiv [factorialCounter]
                Inc factorialCounter
                ; nextTerm /= factorialCounter++;
                fidiv [factorialCounter]
                Inc factorialCounter

                ; Прибавляем к сумме вычисленный член. sum
+= nextTerm;
                fstp [nextTerm]
                fld [sum]
                fadd [nextTerm]
                fstp [sum]

                ; Проверяем на остановку алгоритма.
                ; Если новый член меньше 0.05% от суммы, то
заканчиваем
                ; вычисления и выводим результат.
                fld [sum]
                fmul [epsilon]
                fld [nextTerm]
                fcomp
                fstsw ax
                sahf
```

```
                jb outputResult

                ; term = nextTerm;
                CopyFloat term, nextTerm
                jmp mainLoop

        outputResult:
                PrintFloat resultStr, x, sum

        finish:
                push 0
                invoke scanf, numberStr, x
                call ExitProcess

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'
```