

НИУ ВШЭ

Факультет компьютерных наук

Программная инженерия

Программа для вычисления значения биномиальной функции
 $(1+x)^m$ с помощью степенного ряда

Поляков Лев Алексеевич, БПИ199

Содержание

1. Текст задания
2. Применяемые расчетные методы
3. Тестовые примеры
4. Список использованных источников
5. Приложения

Текст задания

Вариант 17. Разработать программу, вычисляющую с помощью степенного ряда с точностью не хуже 0,05% значение биномиальной функции $(1+x)^m$ для заданного параметра m и x (использовать FPU)

Применяемые расчетные методы

Функция может быть представлена рядом Тейлора:

$$(1+x)^m = 1 + \sum_{k=1}^{\infty} \binom{m}{k} x^k, \text{ для всех } |x| < 1 \text{ и всех } m \in \mathbb{R}, \text{ где } \binom{m}{k} = \prod_{l=1}^k \frac{m-l+1}{l}$$

$$\text{или } 1 + \frac{m}{1!} x^1 + \frac{m(m-1)}{2!} x^2 + \frac{m(m-1)(m-2)}{3!} x^3 + \dots$$

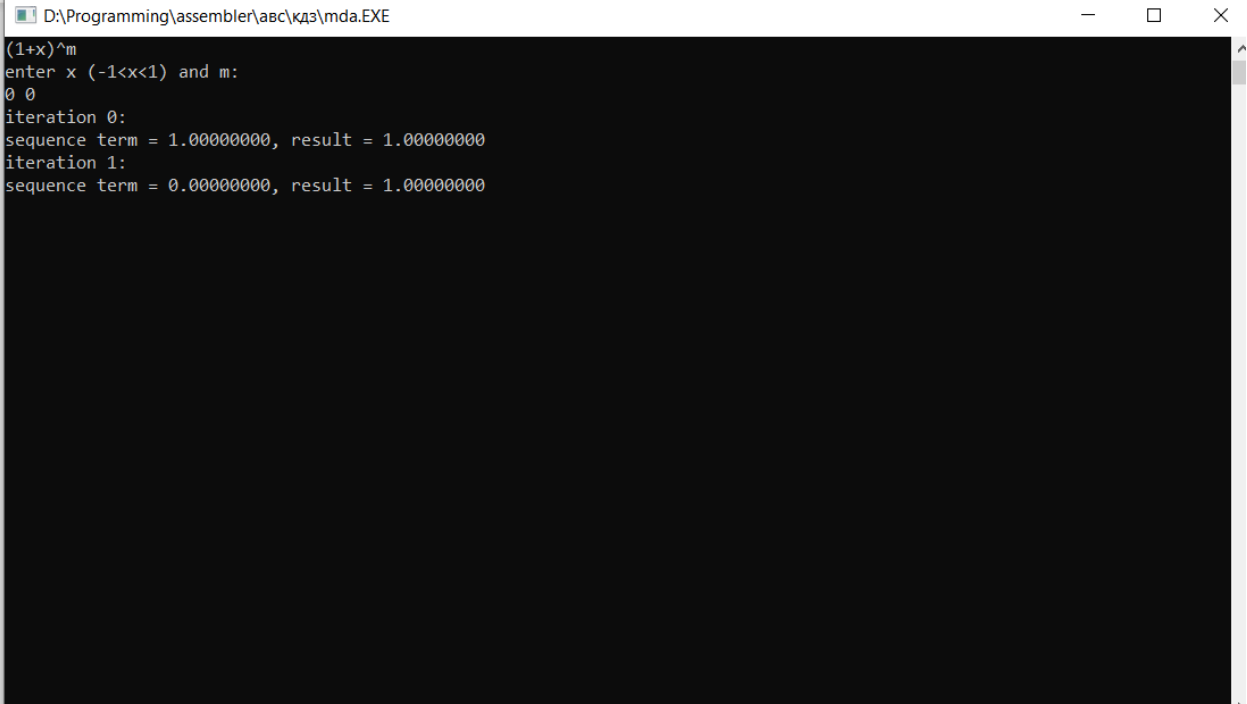
Заметим, что при $m \in \mathbb{N}$ формула превращается в обычный бином Ньютона, и в какой-то момент очередной член ряда станет равным 0. В остальных случаях же это неверно, и ряд с какого-то момента станет знакопеременным. В таком случае, чтобы ряд сходиллся, по признаку Лейбница надо чтобы последовательность из модулей членов ряда монотонно убывала. Найдем отношение модуля $k+1$ -ого члена к модулю k -ого. Это будет $\left| \frac{m-k}{k+1} x \right|$.

Переходя к пределу при $k \rightarrow \infty$, дробь $\frac{m-k}{k+1}$ будет стремиться к -1, и так как мы потребовали $|x| < 1$, то отношение будет меньше 1. Значит последовательность монотонно убывающая, и ряд сходится. Тогда при больших k остаток ряда можно условно считать бесконечно убывающей геометрической прогрессией со знаменателем $-x$. Как оценить остаток ряда? Сумма прогрессии будет равна $b_1/(1-q) = b_1/(1+x)$, где b_1 это очередной член ряда. Это сопоставить с ϵ довольно сложно, однако мною было замечено, что если взять b_1 примерно как ϵ^2 , то погрешность становится незначительной. Тогда достаточно бежать в цикле, пока очередной член ряда по модулю больше ϵ^2

Тестовые примеры

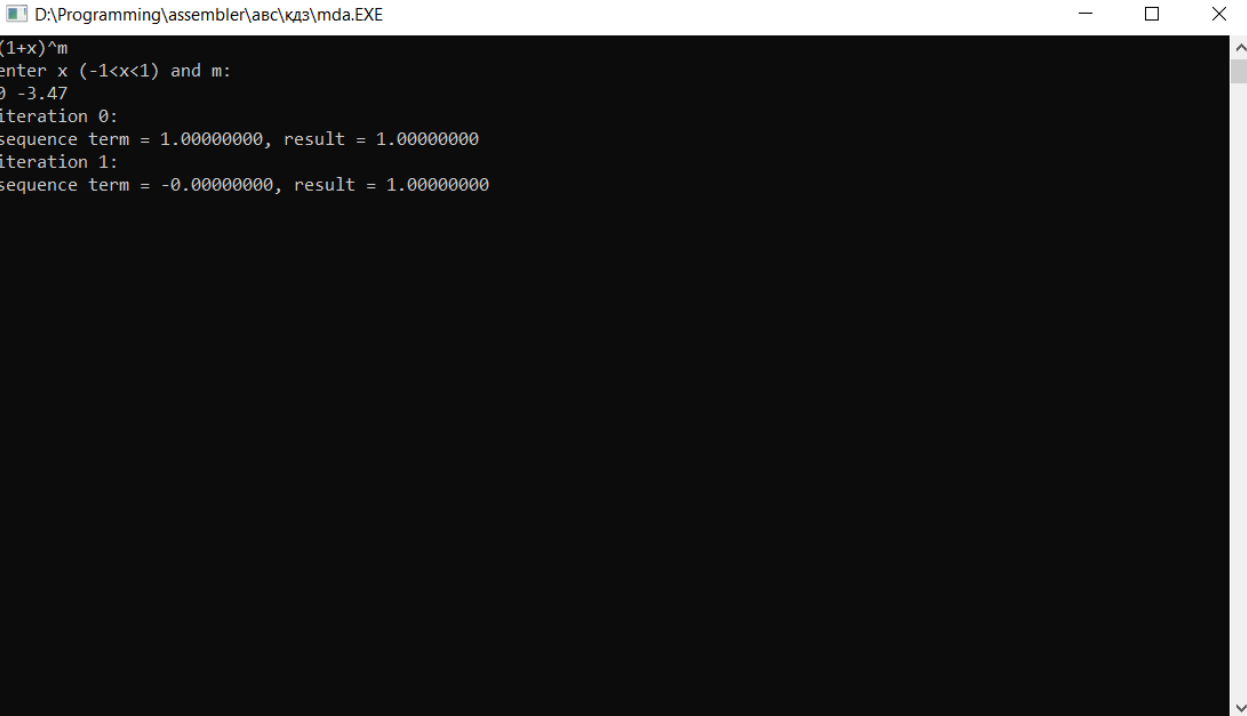
Результаты сравнивались с результатами калькулятора на Android

Результат калькулятора = 1



```
D:\Programming\assembler\abc\кдз\mda.EXE
(1+x)^m
enter x (-1<x<1) and m:
0 0
iteration 0:
sequence term = 1.00000000, result = 1.00000000
iteration 1:
sequence term = 0.00000000, result = 1.00000000
```

Результат калькулятора = 1



```
D:\Programming\assembler\abc\кдз\mda.EXE
(1+x)^m
enter x (-1<x<1) and m:
0 -3.47
iteration 0:
sequence term = 1.00000000, result = 1.00000000
iteration 1:
sequence term = -0.00000000, result = 1.00000000
```

Результат калькулятора = 5,159780352

```
D:\Programming\assembler\авс\кдз\mda.EXE
(1+x)^m
enter x (-1<x<1) and m:
0.2 9
iteration 0:
sequence term = 1.00000000, result = 1.00000000
iteration 1:
sequence term = 1.80000000, result = 2.80000000
iteration 2:
sequence term = 1.44000000, result = 4.24000000
iteration 3:
sequence term = 0.67200000, result = 4.91200000
iteration 4:
sequence term = 0.20160000, result = 5.11360000
iteration 5:
sequence term = 0.04032000, result = 5.15392000
iteration 6:
sequence term = 0.00537600, result = 5.15929600
iteration 7:
sequence term = 0.00046080, result = 5.15975680
iteration 8:
sequence term = 0.00002304, result = 5.15977984
iteration 9:
sequence term = 0.00000051, result = 5.15978035
iteration 10:
sequence term = 0.00000000, result = 5.15978035
```

Результат калькулятора = 0,0004985847562...

```
D:\Programming\assembler\авс\кдз\mda.EXE
(1+x)^m
enter x (-1<x<1) and m:
-0.46 12.34
iteration 0:
sequence term = 1.00000000, result = 1.00000000
iteration 1:
sequence term = -5.67640000, result = -4.67640000
iteration 2:
sequence term = 14.80518648, result = 10.12878648
iteration 3:
sequence term = -23.47312966, result = -13.34434318
iteration 4:
sequence term = 25.21248857, result = 11.86814539
iteration 5:
sequence term = -19.34503823, result = -7.47689284
iteration 6:
sequence term = 10.88609784, result = 3.40920501
iteration 7:
sequence term = -4.53545939, result = -1.12625439
iteration 8:
sequence term = 1.39261281, result = 0.26635842
iteration 9:
sequence term = -0.30891247, result = -0.04255405
iteration 10:
sequence term = 0.04746131, result = 0.00490726
iteration 11:
sequence term = -0.00464431, result = 0.00026296
iteration 12:
sequence term = 0.00023856, result = 0.00050152
iteration 13:
sequence term = -0.00000287, result = 0.00049865
iteration 14:
sequence term = -0.00000006, result = 0.00049859
```

Результат калькулятора = 6,7274999493256...

```
D:\Programming\assembler\авс\кдз\mda.EXE
(1+x)^m
enter x (-1<x<1) and m:
0.1 20
iteration 0:
sequence term = 1.00000000, result = 1.00000000
iteration 1:
sequence term = 2.00000000, result = 3.00000000
iteration 2:
sequence term = 1.90000000, result = 4.90000000
iteration 3:
sequence term = 1.14000000, result = 6.04000000
iteration 4:
sequence term = 0.48450000, result = 6.52450000
iteration 5:
sequence term = 0.15504000, result = 6.67954000
iteration 6:
sequence term = 0.03876000, result = 6.71830000
iteration 7:
sequence term = 0.00775200, result = 6.72605200
iteration 8:
sequence term = 0.00125970, result = 6.72731170
iteration 9:
sequence term = 0.00016796, result = 6.72747966
iteration 10:
sequence term = 0.00001848, result = 6.72749814
iteration 11:
sequence term = 0.00000168, result = 6.72749982
iteration 12:
sequence term = 0.00000013, result = 6.72749994
```

Список использованных источников

<http://natalia.appmat.ru/c&c++/assembler.html>

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

<https://programmersforum.ru/showthread.php?t=227444&page=1>

https://ru.wikipedia.org/wiki/%D0%A0%D1%8F%D0%B4_%D0%A2%D0%B5%D0%B9%D0%BB%D0%BE%D1%80%D0%B0#%D0%A0%D1%8F%D0%B4%D1%8B_%D0%9C%D0%B0%D0%BA%D0%BB%D0%BE%D1%80%D0%B5%D0%BD%D0%B0_%D0%BD%D0%B5%D0%BA%D0%BE%D1%82%D0%BE%D1%80%D1%8B%D1%85_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B

Приложения

Код программы:

format PE console

entry Start

include 'win32a.inc'

section '.data' data readable writable

enterStr db "(1+x)^m",10,"enter x (-1<x<1) and m:",10,0

ffStr db "%lf %lf",0

ffnlStr db "sequence term = %.8lf, result = %.8lf",10,0

inlStr db "iteration %d:",10,0

nl db 10,0

; номер итерации

k dd 0

; вводимый x

x dq ?

; вводимый m

m dq ?

; текущий член ряда

term dq ?

; текущая сумма ряда

sum dq ?

; константа 1

one dq 1.0

; погрешность

epsilon dq 0.0005

; квадрат погрешности

epsilonSq dq ?

section '.code' code readable executable

; макроинструкция для вывода чисел с плавающей запятой

macro PrintFloat str, [args] {

```

reverse
    push dword [args+4]
    push dword [args]
common
    push str
    call [printf]
}

```

Start:

```

    ; finit

    ; ввод x и m
    cinvoke printf, enterStr
    cinvoke scanf, ffStr, x, m

    ; term = 1, sum = 1
    fld qword [one]
    fst qword [term]
    fstp qword [sum]

    ; epsilonSq = epsilon^2
    fld qword [epsilon]
    fmul qword [epsilon]
    fstp qword [epsilonSq]

    ; главный цикл
MainLoop:
    cmp [k], 1000000 ; максимум итераций
    je PrintResult

    ; вывод текущих term и sum
    cinvoke printf, inlStr, [k]
    PrintFloat ffnlStr, term, sum

    fld qword [m]
    fisub dword [k] ; st0 = m-k

```

```

fild dword [k]

fadd qword [one] ; st0 = k+1


fdivp st1, st0 ; st0 = (m-k)/(k+1)

fmul qword [x] ; st0 = (m-k)/(k+1)*x


fmul qword [term] ; st0 = st0*term

fst qword [term] ; term = (m-k)/(k+1)*x*term


fadd qword [sum] ; st0 = st0+sum

fstp qword [sum] ; sum = sum+term


inc [k] ; k++


; если |term| <= epsilon^2 тогда break
; (term <= epsilon^2 && -term >= epsilon^2)

fld qword [epsilonSq] ; epsilon^2

fld qword [term]

fcomi st1

jnbe Last ; если term > epsilon^2


ffree st0 ; st0 = null

ffree st1 ; st1 = null


fld qword [epsilonSq]

fsub qword [epsilonSq]

fsub qword [epsilonSq] ; -epsilon^2

fld qword [term]

fcomi st1

jb Last ; или если -term < epsilon^2


ffree st0 ; st0 = null

ffree st1 ; st1 = null


jmp PrintResult ; break


Last:

ffree st0 ; st0 = null

```

```
    ffree st1 ; st1 = null  
    jmp MainLoop
```

PrintResult:

```
    ; вывод результата  
    cinvoke printf, inlStr, [k]  
    PrintFloat ffnlStr, term, sum
```

Finish:

```
    cinvoke getch  
    cinvoke ExitProcess, 0
```

section '.idata' import data readable

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

```
import kernel32,\  
    ExitProcess, 'ExitProcess',\  
    HeapCreate, 'HeapCreate',\  
    HeapAlloc, 'HeapAlloc',\  
    HeapFree, 'HeapFree'
```

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