

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

Факультет компьютерных наук  
Департамент программной инженерии

**ПРОГРАММА, ВЫЧИСЛЯЮЩАЯ С ПОМОЩЬЮ СТЕПЕННОГО РЯДА С  
ТОЧНОСТЬЮ НЕ ХУЖЕ 0,1% ЗНАЧЕНИЕ ФУНКЦИИ  $e^x$  ДЛЯ ЗАДАННОГО  
ПАРАМЕТРА  $x$**

**Пояснительная записка**

Исполнитель  
студент группы БПИ193

\_\_\_\_\_/С. М. Курбанова /  
« \_\_\_\_ » \_\_\_\_\_ 2020 г.

**Москва 2020**

## **Оглавление**

Текст задания	3
Применяемые расчётные методы	4
Обоснование области допустимых значений входных параметров	5
Тестовые примеры	6
Список использованной литературы	7
ПРИЛОЖЕНИЕ	8

### Текст задания

Разработать программу, вычисляющую с помощью степенного ряда с точностью не хуже 0,1% значение функции  $e^x$  для заданного параметра  $x$  (использовать FPU).

## Применяемые расчётные методы

Для нахождения значения функции  $f(x) = e^x$  с точностью  $\varepsilon = 0.001$  была использована формула:

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

Формула нахождения каждого последующего члена:

$$x_n = x_{n-1} \times \frac{x}{n}$$

Для нахождения результата искомой функции в алгоритме к переменной result прибавляются новые члены до тех пор, пока значение последующего члена  $x_n$  превосходит или равно по модулю  $\varepsilon$ , то есть пока результат вычислений не начнёт увеличиваться на значение меньше  $\varepsilon$ .

```
CalcExp:
    mov [tmpStack], esp

expLoop:
; check if result is infinity
    fstsw ax
    and ax, 1000b
    cmp ax, 0
    jg inf

    inc [i]          ; i++

    ; x_n *= x / i
    fld [x]          ; st0 = x
    fdiv [i]          ; st0 = x / i
    fmul [x_n]        ; x_n *= st0
    fstp [x_n]        ; x_n = st0

    ; result += x_n
    fld [result]      ; st0 = result
    fadd [x_n]         ; st0 += x_n
    fstp [result]     ; result = st0

    fld [x_n]         ; st0 = x_n
    fabs             ; st0 = |x_n|
    fcomp [eps]       ; compare |x_n| with 0.001
    fstsw ax
    sahf
    ja expLoop        ; continue loop if result isn't accurate enough

    mov esp, [tmpStack]
    ret
```

Рисунок 1. Фрагмент кода, соответствующий вышеописанному алгоритму.

Промежуточные вычисления располагаются в переменной result, в ней же хранится конечный результат вычислений.

Исходные данные располагаются в переменной x.

### **Обоснование области допустимых значений входных параметров**

Минимальное допустимое значение входного параметра  $x$  равняется -6, т. к. использованная в расчетах точность  $\varepsilon = 0.001$  превосходит значение функции  $e^x$  при  $x < -6$ .

Ограничение на максимальное допустимое значение входного параметра не предусмотрено, однако при переполнении в консоль выводится соответствующее сообщение (см. рисунок 3).

## Тестовые примеры

### 1. Работа программы при вводе некорректных данных.



```
C:\Users\Sabiyat\Documents\GitHub\fas\microproject01\exp.EXE
Enter x: -7
The input value is too small.
```

Рисунок 2. Ввод слишком маленького значения  $x$ .



```
C:\Users\Sabiyat\Documents\GitHub\fas\microproject01\exp.EXE
Enter x: 710
The value of exp(x) is too big.
```

Рисунок 3. Ввод слишком большого значения  $x$ .



```
C:\Users\Sabiyat\Documents\GitHub\fas\microproject01\exp.EXE
Enter x: fff
The input value must be a number.
```

Рисунок 4. Ввод строкового значения  $x$ .

### 2. Работа программы при вводе корректных данных.



```
C:\Users\Sabiyat\Documents\GitHub\fas\microproject01\exp.EXE
Enter x: -6
exp(-6) = 0.002386
```

Рисунок 5. Ввод корректного целого отрицательного значения  $x$ .



```
C:\Users\Sabiyat\Documents\GitHub\fas\microproject01\exp.EXE
Enter x: -1.5
exp(-1.5) = 0.223222
```

Рисунок 6. Ввод корректного вещественного отрицательного значения  $x$ .



```
C:\Users\Sabiyat\Documents\GitHub\fas\microproject01\exp.EXE
Enter x: 20
exp(20) = 485165195.409585
```

Рисунок 7. Ввод корректного целого положительного значения  $x$ .



```
C:\Users\Sabiyat\Documents\GitHub\fas\microproject01\exp.EXE
Enter x: 15.5
exp(15.5) = 5389698.476123
```

Рисунок 8. Ввод корректного вещественного положительного значения  $x$ .

### Список использованной литературы

1. Инструкции FPU // [Электронный ресурс] Режим доступа: <http://flatassembler.narod.ru/fasm.htm#2-1-13>, свободный;
2. Требования к оформлению // [Электронный ресурс] Режим доступа: <http://softcraft.ru/edu/comparch/tasks/mp01/>, свободный;
3. Примеры программ // [Электронный ресурс] Режим доступа: <http://softcraft.ru/edu/comparch/practice/asm86/05-fpu/>, свободный.

## ПРИЛОЖЕНИЕ

### Текст программы

```
format PE console
entry start

include 'win32a.inc'

section '.data' data readable writable

    strInput      db 'Enter x: ', 0
    strOutOfBounds db 'The input value is too small.',
10, 0
    strWrongFormat db 'The input value must be a
number.', 10, 0
    strInfRes      db 'The value of exp(x) is too big.',
10, 0
    fmt            db '%lf', 0
    fmtfp          db '%lf', 13, 10, 0
    outputStr      db 'exp(%g) = %lf', 10, 0

    x              dq ?
    maxL           dd -6.0
    result         dq 1.0      ; result of calculation
exp(x)
    x_n            dq 1.0
    tmpStack       dd ?
    i              dd 0
    eps            dq 0.001

    NULL = 0

section '.code' readable executable

start:
    ; Gets x and checks the value
    call Input

    ; Calcs exp(x)
    call CalcExp

    ; Print result
    invoke printf, outputStr, dword [x], dword
[x + 4],\
                                dword [result],
dword [result + 4]

finish:
    call [getch]
    push NULL
    call [ExitProcess]
```



```

inf:
    push strInfRes
    call [printf]
    jmp finish

outOfBounds:
    push strOutOfBounds
    call [printf]
    jmp finish

wrongFormat:
    push strWrongFormat
    call [printf]
    jmp finish

;-----

Input:
    mov [tmpStack], esp

    invoke printf, strInput
    invoke scanf, fmt, x

    cmp eax, 1
    jne wrongFormat ; jump to wrongFormat if
input value isn't a number

    finit
    fld [x]          ; st0 = x
    fcomp [maxL]     ; compare x with left border
-6.0
    fstsw ax
    sahf
    jnb outOfBounds ; jump to outOfBounds if x
less than -6.0

    mov esp, [tmpStack]
    ret

;-----

CalcExp:
    mov [tmpStack], esp

expLoop:
; check if result is infinity
    fstsw ax
    and ax, 1000b
    cmp ax, 0
    jg inf

    inc [i]          ; i++

```

```

        ; x_n *= x / i
        fld [x]                ; st0 = x
        fdiv [i]               ; st0 = x / i
        fmul [x_n]             ; x_n *= st0
        fstp [x_n]             ; x_n = st0

        ; result += x_n
        fld [result]           ; st0 = result
        fadd [x_n]             ; st0 += x_n
        fstp [result]          ; result = st0

        fld [x_n]              ; st0 = x_n
        fabs                   ; st0 = |x_n|
        fcomp [eps]            ; compare |x_n| with 0.001
        fstsw ax
        sahf
        ja expLoop             ; continue loop if result
isn't accurate enough

        mov esp, [tmpStack]
        ret

;-----

section '.idata' import data readable

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

        import kernel,\
            ExitProcess, 'ExitProcess'

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

```