Программа вычисления числа Пи с точностью не хуже 0,1% посредством ряда Нилаканта (с использованием FPU)

Пояснительная записка

Москва 2020 г.

Текст задания: разработать программу вычисления числа Пи с точностью не хуже 0,1% посредством ряда Нилаканта (с использованием FPU).

Программа написана на последней версии **flat assembler**.

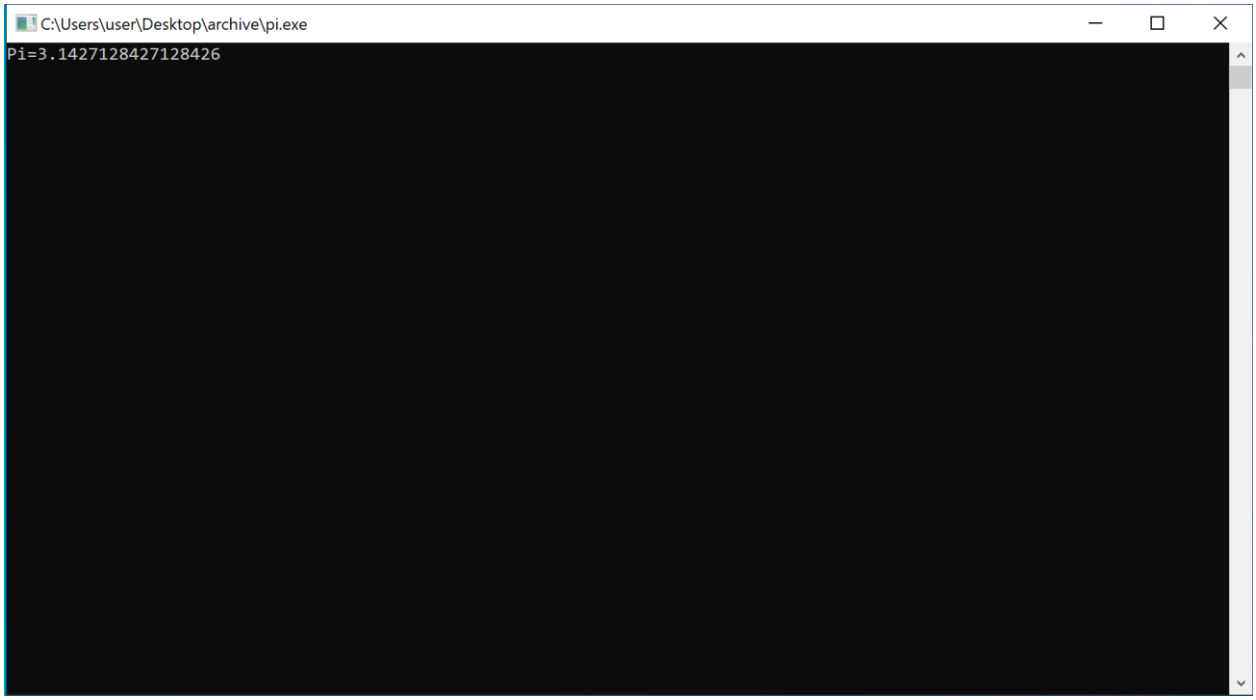
В предоставленной программе для вычисления числа Пи используется **ряд Нилаканта**.

Ряд Нилаканта позволяет получить искомый результат по формуле: *π = 3 + 4/(2*3*4) — 4/(4*5*6) + 4/(6*7*8) — 4/(8*9*10) + 4/(10*11*12) — (4/(12*13*14)*

Текст программы:

```
format pe console 4.0

include 'win32a.inc'

  .start:
        finit
; pi = 3 + 4/(2*3*4) - 4/(4*5*6) + 4/(6*7*8) - 4/(8*9*10) + 4/(10*11*12) - (4/(12*13*14)

        xor     eax,eax
        fld     qword [_err]    ; ST0=0.001
        fldl
        fld         ST0             ; ST0=1         ST1=1         ST2=0.001
        fadd    ST1,ST0         ; ST0=1         ST1=2         ST0=0.001
        fld     ST1             ; ST0=2         ST1=1         ST2=2         ST3=0.001
        fadd    ST2,ST0         ; ST0=2         ST1=1         ST2=4         ST3=0.001
        fadd    ST1,ST0         ; ST0=2         ST1=3         ST2=4         ST3=0.001
        fldl                    ; ST0=1         ST1=2         ST2=3         ST3=4         ST4=0.001
; ST0 - 1
; ST1 - первый делитель (ПД)
; ST2 - значение предыдущей итерации (ЗПИ)
; ST3 - 4
; ST4 - эталонная погрешность (ЭП)
  .next:
        fld     ST3
        fld     ST2             ; ST0=ПД        ST1=4         ST2(0)=1      ST3(1)=ПД     ST4(2)=ЗПИ    ST5(3)=4      ST6(4)=ЭП
        fdiv    ST1,ST0
        fadd    ST0,ST2
        fdiv    ST1,ST0
        fadd    ST0,ST2
        fdiv    ST1,ST0
        fxch    ST3             ; Заменяем значение первого делителя для следующей итерации на значение последнего делителя для текущей итерации
        fstp    ST0             ; ST0 - результата деления 4 на очередную группу цифр
                                ;               ST1(0)=1      ST2(1)=ПД     ST3(2)=ЗПИ    ST4(3)=4      ST5(4)=ЭП
        fld     ST3             ; ST0 - ЗПИ     ST1 - результата деления 4 на очередную группу цифр
                                ;               ST2(0)=1      ST3(1)=ПД     ST4(2)=ЗПИ    ST5(3)=4      ST6(4)=ЭП
        inc     eax
        jz      .end            ; защита от слишком долгого выполнения
        test    al,1            ; переход если четное
        jz      @f
        faddp
        jmp     .eoo
  @@:
        fsubrp
  .eoo:




                                ;               ST1(0)=1      ST2(1)=ПД     ST3(2)=ЗПИ    ST4(3)=4      ST5(4)=ЭП
; считаем погрешность
        fld     ST0             ; ST0=ЗТИ       ST1=ЗТИ       ST2(0)=1      ST3(1)=ПД     ST4(2)=ЗПИ    ST5(3)=4      ST6(4)=ЭП
        fsub    ST0,ST4
        fdiv    ST0,ST4
        fabs                    ; ST0=Погрешность
                                ;               ST1=ЗТИ       ST2(0)=1      ST3(1)=ПД     ST4(2)=ЗПИ    ST5(3)=4      ST6(4)=ЭП
        fcomp   ST6             ; ST0=ЗТИ       ST1(0)=1      ST2(1)=ПД     ST3(2)=ЗПИ    ST4(3)=4      ST5(4)=ЭП
        fnstsw  word [_fsw]     ; Сохраняем регистр sw, в нем выставлены флаги по результатам сравнения.
        fxch    ST3
        fstp    ST0             ; ST0=1         ST1=ПД        ST2=ЗПИ       ST3=4         ST4=ЭП
        and     byte [_fsw+1],45h
        jnz     @f              ; Погрешность меньше целевой

        jmp     .next
  .end:
        fstp    ST0                             ; Очищаем стек FPU от лишних значений
        fstp    ST0                             ; Очищаем стек FPU от лишних значений
  @@:
        fstp    ST0                             ; Очищаем стек FPU от лишних значений
        fstp    ST0                             ; Очищаем стек FPU от лишних значений
        fstp    qword [_pi]
        cinvoke printf, _format, dword [_pi], dword [_pi+4]
        fstp    ST0                             ; Очищаем стек FPU от лишних значений
        fstp    ST0                             ; Очищаем стек FPU от лишних значений
  .exit:
        cinvoke getchar                 ; Ждем ввода чего-нибудь
        invoke ExitProcess,0
_fsw    dw      ?
_err    dq      0.001
_pi     dq      3.0
_message db     1024 dup (0)
_format db      'Pi=%.16f',13,10,0

data import

 library kernel32,'KERNEL32.DLL',\
         msvcrt,'msvcrt.DLL'

 import kernel32,\
         ExitProcess,'ExitProcess'

 import msvcrt,\
        printf,'printf',\
        getchar,'getchar'

end data
```

Тест программы:



Список используемых источников:

http://flatassembler.narod.ru/fasm.htm

https://newtonov.ru/chemu-ravno-chislo-pi/

Москва 2020 г.