Evaluación 2

Eduardo Hndz
Universidad De Sonora
Lic. en Física

December 1, 2017

1 actividad1

en esta actividad se nos proporcionó un código que calculaba el valor de la exponencial, así como su aproximación mediante la serie de Maclaurin.

El programa solo daba los primeros 20 términos necesarios para calcular el valor de e a la primera potencia.

el código muestra es como sigue

```
! ----------- Begin ------------
!taylor.f90
program taylor

    implicit none
real (kind=8) :: x, exp_true, y
    real (kind=8), external :: exptaylor
    integer :: n

    n = 20               ! number of terms to use
    x = 1.0
    exp_true = exp(x)
    y = exptaylor(x,n)   ! uses function below
    print *, "x = ",x
    print *, "exp_true  = ",exp_true
    print *, "exptaylor = ",y
    print *, "error     = ",y - exp_true

end program taylor

!==========================
function exptaylor(x,n)
!==========================
    implicit none
```

```
    ! function arguments:
    real (kind=8), intent(in) :: x
    integer, intent(in) :: n
    real (kind=8) :: exptaylor

    ! local variables:
    real (kind=8) :: term, partial_sum
    integer :: j

    term = 1.
    partial_sum = term

    do j=1,n
        ! j'th term is  x**j / j!  which is the previous term times x/j:
        term = term*x/j
        ! add this term to the partial sum:
        partial_sum = partial_sum + term
        enddo
     exptaylor = partial_sum  ! this is the value returned
end function exptaylor
! --------  End --------------
```

los datos que arrojaba este programa son como sigue

```
 x =    1.0000000000000000
 exp_true  =    2.7182818284590451
 exptaylor =    2.7182818284590455
 error     =    4.4408920985006262E-016
```

2 actividad2

a continuacion se muestra mi intento fallido de la actividad2, en la cual debíamos realizar una subturina para calcular las aproximaciones del polinomio de Taylor mediante series de Maclauri y compararlas en Gnuplot

```
subroutine expD(x,x1,n)
  real(kind=8), intent(in)::x
  real(kind=8),dimension(100), intent(out)::x1
  integer, intent(in)::n
  !variables
  real(kind=8):: term,partial_sum,fi
  integer::i

  term=1.
  partial_sum=term
  do i=1,n
```

```
     fi=float(i)
     term=term*x/fi
  end do

end subroutine expD


program Taylor
  implicit none
  real(kind=8) :: x,term,partial_sum,exp_true
  real(kind=8),dimension(100):: x1
  integer ::i,j,n
  open(unit=1, file='taylor.dat',status='unknown')

do j=1,15,2
     x=float(j)
     call  expD(x,x1,n)
     exp_true=exp(x)

  print*, "x=", x1
  print*, "exp_true = " , exp_true
  print*, "error=", x1-exp_true
  write(1,*) x1 , exp_true
   end do

end program Taylor
```