Departamento
de Física

Ciencias Exactas Y Naturales
Programación y Lenguaje Fortran

Compiladores e Interpretadores
Para diferentes lenguajes de programación

Sandra Yulissa Méndez Parra

Universidad de Sonora
Mex.

Fortran

Hermosillo, Sonora 2015

INTRODUCCIÓN:

En lecciones anteriores se ha descrito el concepto de diseo descendente; esta técnica permite desarrollar algoritmos que resuelvan un problema mediante un proceso de refinamiento progresivo, descomponiendo el problema original en subproblemas menores hasta obtener una granularidad suficientemente fina que permita resolver cada subproblema mediante un algoritmo sencillo. Generalmente, por cada nivel del diseo descendente se desarrollo un pseudocódigo de alto nivel que hace uso de acciones no primitivas; si se detecta que alguna de estas acciones no primitivas aparece más de una vez es posible nombrarla y utilizarla de forma repetida. Tales acciones con nombre se denominan subprogramas y pueden ser, a su vez, funciones y subrutinas

La utilizacin de subprogramas proporciona mltiples ventajas:

- Facilitan la modularidad y estructuracin de los algoritmos.
- Facilitan la lectura e inteligibilidad de los algoritmos.
- Permiten una economizacin del esfuerzo del programador al poder escribir cdigo reutilizable en muchas partes de un mismo algoritmo.
- Facilitan la depuracin y mantenimiento de los programas.

FUNCIONES:

Las funciones son subrutinas que pueden tener o no argumentos pero que siempre devuelven un valor de retorno. As pues, las invocaciones a funciones son expresiones de un tipo determinado y deben emplearse igual que cualquier expresin de su tipos; es decir, una llamada a funcin puede formar parte de una expresin aritmtica, lgica o de cadena en funcin de su tipo, puede constituir la parte derecha de una sentencia de asignacin, aparecer en una sentencia de salida o constituir un argumento para otro subprograma. Por otro lado, las llamadas a funciones nunca pueden formar una sentencia aislada ni constituir la parte izquierda de una sentencia de asignacin.

SUNRUTINAS:

En muchas ocasiones puede interesarnos desarrollar un subprograma que no se vea afectado por las limitaciones de las funciones; es decir, puede interesarnos un subprograma que sea capaz de retornar varios valores o ninguno. Para esos casos existen las denominadas subrutinas o procedimientos; las subrutinas son subprogramas que no devuelven ningn resultado, por tanto no tienen tipo, y en los que es lcito emplear los efectos laterales antes mencionados para permitir al programa principal obtener varios valores resultantes de la ejecucin del subprograma. Las subrutinas se diferencian de las funciones fundamentalmente en la sintaxis de la definicin y en la forma de invocarlos; dado que no tienen tipo alguno las subrutinas no pueden formar parte de expresiones ni aparecer en la parte derecha de una sentencia de asignacin, deben aparecer nica y exclusivamente en una sentencia de llamada a procedimiento.

ACTIVIDADES A REALIZAR:

Nos interesa utilzar el concepto de funciones en Fortran, para desarrollar una funcin "Taylor", a la que le demos un par de valores de una funcin y nos

regrese el valor de la Serie de Taylor para un punto.

Escriba una funcin de Taylor en Fortran, que utilice la primera y segunda derivada, por lo que requiere entonces de conocer los valores de una funcin en $f(x-h)$, $f(x)$ y $f(x+h)$.

Utilizando las aproximaciones de diferenciacin numrica para la primera y segunda derivada, calcule el error E(x) como funcin de x, de aproximar con los primeros 3 trminos a una funcin f(x), para los siguientes casos.

1. f(x) = sin(x), alrededor de a=0.
2. f(x) = cos(x), alrededor de a=0.
3. f(x) = tan(x), alrededor de a=0.
4. f(x) = exp(x), alrededor de a=0.

Grafique las funciones error $E(x) = f(x) - Taylor(x, a)$, como funcin de x para cada uno de los casos.

El codigo fuente del programa utilizado

```
program taylorprogram

  IMPLICIT NONE

  INTERFACE

     FUNCTION SerieTaylor(f_0,f_1,f_2,h)
       REAL :: SerieTaylor
       REAL, INTENT(IN) :: f_0, f_1, f_2, h
     END FUNCTION SerieTaylor
   END INTERFACE

  REAL :: x_0, x_1, x_2, h, suma, F
  INTEGER :: i, npts, icaso
  write(*,*) "Ingresa el numero correspondiente para el caso deseado caso seno: 1 ,coseno: 2
  read(*,*) icaso
  write(*,*) icaso
  npts=21

     write(*,*) "Serie de Taylor del Caso Seleccionado Anteriormente"
     h=0.1
```

```
  do i=1, npts
     x_0=float(i-1)*h
     x_1=float(i)*h
     x_2=float(i+1)*h
    suma =SerieTaylor(F(x_0,icaso), F(x_1,icaso), F(x_2,icaso), h)

    write(*,*) i, suma, F(x_1,icaso), (suma-F(x_1,icaso))
    open(unit=12, file='Error.dat')
    write(12,*)i, suma-F(x_1,icaso)
  end do
 close(12)

END PROGRAM taylorprogram

FUNCTION SerieTaylor(f_0,f_1,f_2,h)

  IMPLICIT NONE
  REAL :: SerieTaylor
  REAL, INTENT(IN) :: f_0, f_1, f_2, h
  SerieTaylor=f_1+((f_2-f_0)/(2*h))*h+((f_2-2*f_1+f_0)/(2*h*h))*h*h

  END FUNCTION SerieTaylor

  FUNCTION F(x,icaso)

    IMPLICIT NONE
    REAL :: F
    REAL, INTENT(IN) ::  x
    INTEGER, INTENT(IN) :: icaso
    if (icaso.EQ.1) then
       F=sin(x)
    end if

    if (icaso.EQ.2) then
       F=cos(x)
    end if
    if (icaso.EQ.3) then
       F=tan(x)
    end if
    if (icaso.EQ.4) then
       F=exp(x)
    end if

  END FUNCTION F
```

Las grficas obtenidas apartir del Codigo. Las graficas muestran el error obtenido entre la medicin y la apoximacin.