

Reporte de la actividad 3

Hedwin Aaron Encinas Acosta

25 de Febrero de 2015

Esta es una practica (Actividad 3 215-1) del curso de Fortran en el que veremos algunos ejemplos del lenguaje de programación Fortran. Los ejemplos son sencillos calcularemos el área de un círculo así como su volumen, determinaremos la precisión de la computadora y calcularemos algunos valores con diferentes funciones entre otros.

1. Calcula el area de un círculo

Este es un programa que calculara el el área de un círculo introduciendo el radio como se muestra en la imagen.

```
! Area . f90 : Calculates the area of a circle, sample program
! -----
Program Circle_area ! Begin main program
  Implicit None ! Declare all variables
  Real *8 :: radius , circum , area ! Declare Reals
  Real *8 :: PI = 4.0 * atan(1.0) ! Declare , assign Real
  Integer :: model_n = 1 ! Declare , assign Ints
  print * , 'Enter a radius:' ! Talk to user
  read * , radius ! Read into radius
  circum = 2.0 * PI * radius ! Calc circumference
  area = radius * radius * PI ! Calc area
  print * , 'Program number =' , model_n ! Print program number
  print * , 'Radius =' , radius ! Print radius
  print * , 'Circumference =' , circum ! Print circumference
  print * , 'Area =' , area ! Print area
End Program Circle_area ! End main program code
```

```

hedwin@hedwin-KJ389AA-ABA-SR5433WM: ~/Escritorio/Producto3
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$ ./Areax
Enter a radius:
6
Program number =      1
Radius =  6.0000000000000000
Circumference =  37.699112892150879
Area =  113.09733867645264
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$

```

2. Calcula el volumen de un esfera

En este programa calcularemos el volumen de una parte de una esfera, en este ejemplo se necesitara introducir dos variables en lugar de una.

```

! volumen . f90 : Calculates the area of a circle, sample program
! -----

```

```

Program Circle_area ! Begin main program
  Implicit None ! Declare all variables
  Real *8 :: radius , circum , area, volume, height ! Declare Reals
  Real *8 :: PI = 4.0 * atan(1.0) ! Declare , assign Real
  Real *8 :: f= 1.0/3.0
  Integer :: model_n = 1 ! Declare , assign Ints
  print * , 'Enter a radius:' ! Talk to user
  print * , 'Enter a height:'
  read * , radius ! Read into radius

```

```

read * , height
  volume = f * PI * height * height * ( 3.0 * radius - height )
print * , 'Program number =' , model_n ! Print program number
print * , 'Radius =' , radius ! Print radius
print * , 'height =' , circum ! Print circumference
print * , 'Volume=' , volume
End Program Circle_area ! End main program code

```

```

hedwin@hedwin-KJ389AA-ABA-SR5433WM: ~/Escritorio/Producto3
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$ ./volumen
Enter a radius:
Enter a height:
44
55
Program number =      1
Radius =  44.000000000000000
height =  55.000000000000000
Volume= 243918.50366945553
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$

```

3. Determinar la precision de la maquina

Determinaremos la precisión de la máquina con el siguiente código de doble precisión (*8)

```

! Limits . f90 : Determines machine precision
! -----
Program Limits
Implicit None

```

```

Integer :: i , n
Real *8 :: epsilon_m , one
n=60 ! Establish the number of iterations
! Set initial values :
epsilon_m = 1.0
one = 1.0
! Within a DO-LOOP, calculate each step and print .
! This loop will execute 60 times in a row as i is
! incremented from 1 to n ( since n = 60) :
do i = 1, n , 1 ! Begin the do-loop
    epsilon_m = epsilon_m / 2.0 ! Reduce epsilon m
    one = 1.0 + epsilon_m ! Re-calculate one
    print * , i , one , epsilon_m ! Print values so far
end do ! End loop when i>n
End Program Limits

```

```

hedwin@hedwin-KJ389AA-ABA-SR5433WM: ~/Escritorio/Producto3
21  1.0000004768371582    4.7683715820312500E-007
22  1.0000002384185791    2.3841857910156250E-007
23  1.0000001192092896    1.1920928955078125E-007
24  1.0000000596046448    5.9604644775390625E-008
25  1.0000000298023224    2.9802322387695312E-008
26  1.0000000149011612    1.4901161193847656E-008
27  1.0000000074505806    7.4505805969238281E-009
28  1.0000000037252903    3.7252902984619141E-009
29  1.0000000018626451    1.8626451492309570E-009
30  1.0000000009313226    9.3132257461547852E-010
31  1.0000000004656613    4.6566128730773926E-010
32  1.0000000002328306    2.3283064365386963E-010
33  1.0000000001164153    1.1641532182693481E-010
34  1.0000000000582077    5.8207660913467407E-011
35  1.0000000000291038    2.9103830456733704E-011
36  1.0000000000145519    1.4551915228366852E-011
37  1.0000000000072760    7.2759576141834259E-012
38  1.0000000000036380    3.6379788070917130E-012
39  1.0000000000018190    1.8189894035458565E-012
40  1.0000000000009095    9.0949470177292824E-013
41  1.0000000000004547    4.5474735088646412E-013
42  1.0000000000002274    2.2737367544323206E-013
43  1.0000000000001137    1.1368683772161603E-013
44  1.0000000000000568    5.6843418860808015E-014
45  1.0000000000000284    2.8421709430404007E-014
46  1.0000000000000142    1.4210854715202004E-014
47  1.0000000000000071    7.1054273576010019E-015
48  1.0000000000000036    3.5527136788005009E-015
49  1.0000000000000018    1.7763568394002505E-015
50  1.0000000000000009    8.8817841970012523E-016
51  1.0000000000000004    4.4408920985006262E-016
52  1.0000000000000002    2.2204460492503131E-016
53  1.0000000000000000    1.1102230246251565E-016
54  1.0000000000000000    5.5511151231257827E-017
55  1.0000000000000000    2.7755575615628914E-017
56  1.0000000000000000    1.3877787807814457E-017
57  1.0000000000000000    6.9388939039072284E-018
58  1.0000000000000000    3.4694469519536142E-018
59  1.0000000000000000    1.7347234759768071E-018
60  1.0000000000000000    8.6736173798840355E-019
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$

```

4. Determinar la precision de la maquina

Determinaremos la precisión de la máquina con el siguiente código de doble precisión (*4)

```
! Limits . f90 : Determines machine precision
! -----
Program Limits
Implicit None
Integer :: i , n
Real *4 :: epsilon_m , one
n=60 ! Establish the number of iterations
! Set initial values :
epsilon_m = 1.0
one = 1.0
! Within a DO-LOOP, calculate each step and print .
! This loop will execute 60 times in a row as i is
! incremented from 1 to n ( since n = 60) :
do i = 1, n , 1 ! Begin the do-loop
  epsilon_m = epsilon_m / 2.0 ! Reduce epsilon m
  one = 1.0 + epsilon_m ! Re-calculate one
  print * , i , one , epsilon_m ! Print values so far
end do ! End loop when i>n
End Program Limits
```

```

hedwin@hedwin-KJ389A-A-ABA-SR5433WM: ~/Escritorio/Producto3
21 1.00000048 4.76837158E-07
22 1.00000024 2.38418579E-07
23 1.00000012 1.19209290E-07
24 1.00000000 5.96046448E-08
25 1.00000000 2.98023224E-08
26 1.00000000 1.49011612E-08
27 1.00000000 7.45058060E-09
28 1.00000000 3.72529030E-09
29 1.00000000 1.86264515E-09
30 1.00000000 9.31322575E-10
31 1.00000000 4.65661287E-10
32 1.00000000 2.32830644E-10
33 1.00000000 1.16415322E-10
34 1.00000000 5.82076609E-11
35 1.00000000 2.91038305E-11
36 1.00000000 1.45519152E-11
37 1.00000000 7.27595761E-12
38 1.00000000 3.63797881E-12
39 1.00000000 1.81898940E-12
40 1.00000000 9.09494702E-13
41 1.00000000 4.54747351E-13
42 1.00000000 2.27373675E-13
43 1.00000000 1.13686838E-13
44 1.00000000 5.68434189E-14
45 1.00000000 2.84217094E-14
46 1.00000000 1.42108547E-14
47 1.00000000 7.10542736E-15
48 1.00000000 3.55271368E-15
49 1.00000000 1.77635684E-15
50 1.00000000 8.88178420E-16
51 1.00000000 4.44089210E-16
52 1.00000000 2.22044605E-16
53 1.00000000 1.11022302E-16
54 1.00000000 5.55111512E-17
55 1.00000000 2.77555756E-17
56 1.00000000 1.38777878E-17
57 1.00000000 6.93889390E-18
58 1.00000000 3.46944695E-18
59 1.00000000 1.73472348E-18
60 1.00000000 8.67361738E-19
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$

```

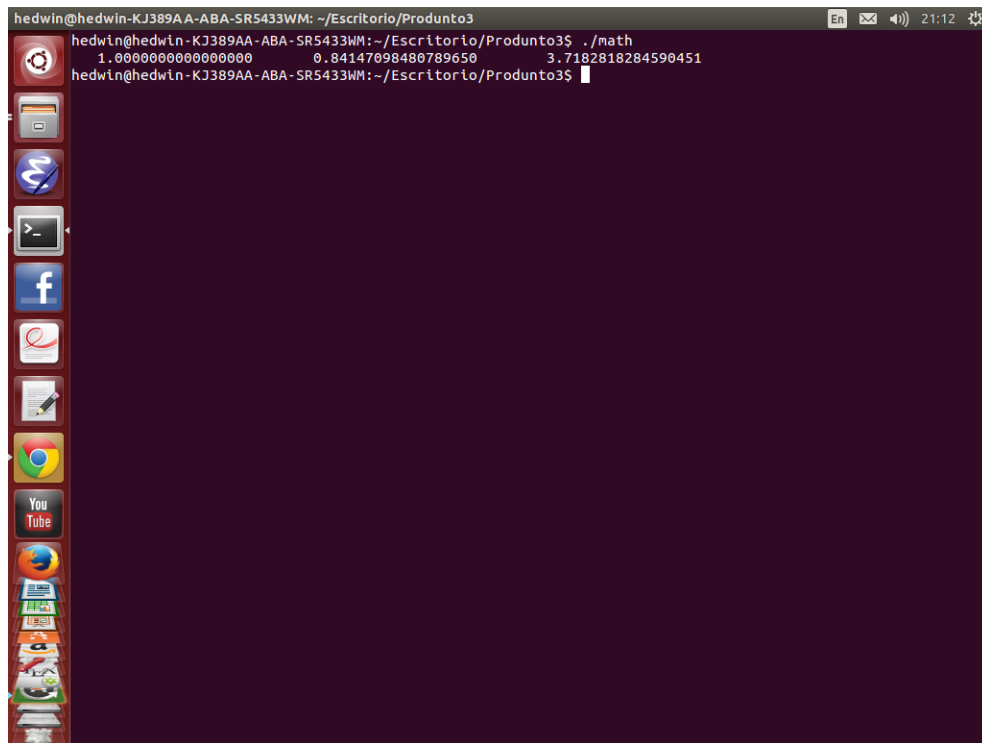
5. Funciones trigonométricas y las especiales

En el siguiente programa se muestran la función seno y la función exponencial

```

! Math . f90 : demo some Fortran math functions
! -----
Program Math test ! Begin main program
  Real *8 :: x = 1.0 , y, z ! Declare variables x, y, z
  y = sin (x) ! Call the sine function
  z = exp (x) + 1.0 ! Call the exponential function
  print * , x, y, z ! Print x, y, z
End Program Math test ! End main program

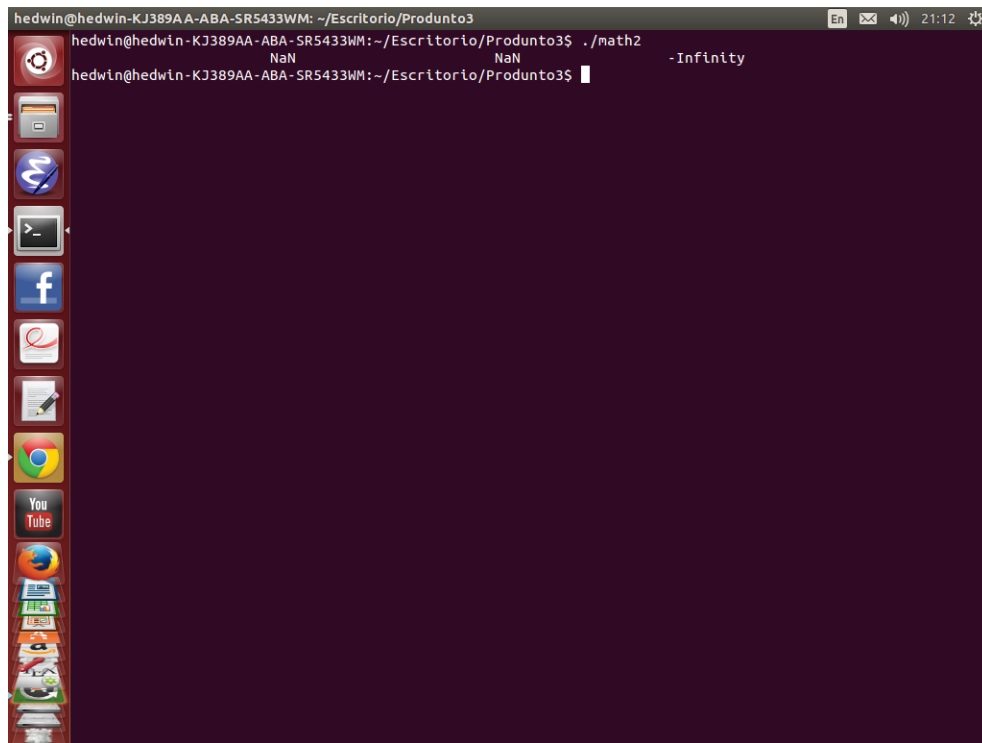
```



6. Funciones las especiales

Este programa es similar al anterior solo que aquí aremos diferente operaciones como raíz cuadrada, arcoseno y logaritmo.

```
! Math . f90 : demo some Fortran math functions
! -----
Program Math ! Begin main program
Real *8 :: x = -1.0 , y = 2.0, z = 0 ! Declare variables x, y, z
x = sqrt (x)
y = asin (y) ! Call the sine function
z = log (z) ! Call the exponential function
print * , x, y, z ! Print x, y, z
End Program Math ! End main program
```



7. Calcular el valor de una funcion

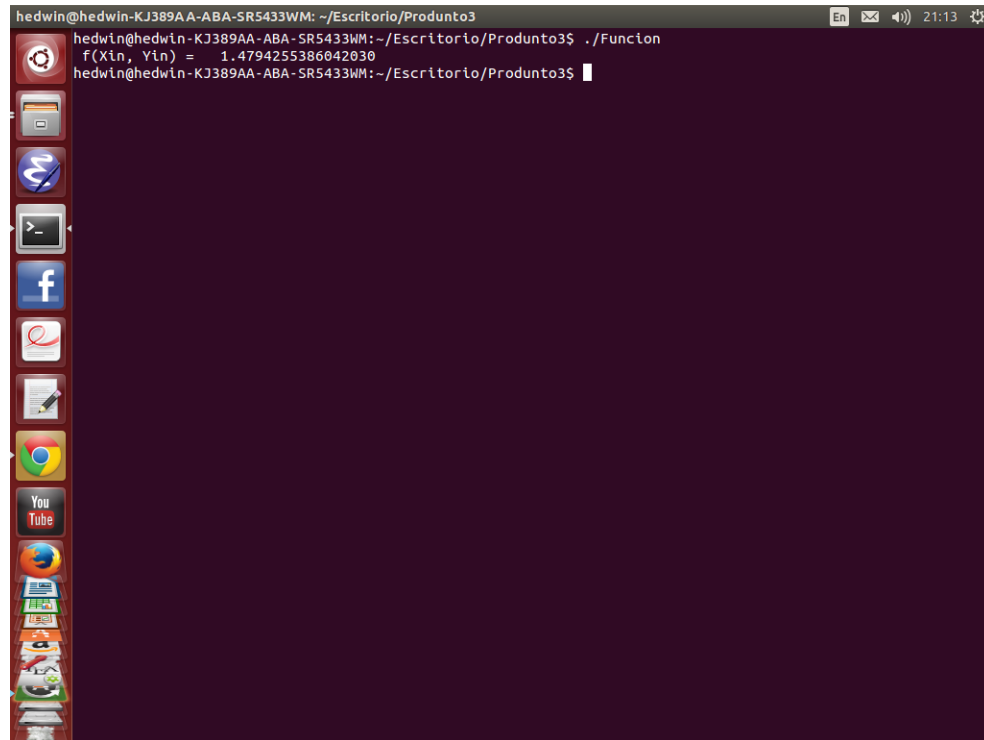
se calculara el valor de la funcion $f(x, y) = 1 + \sin(x \cdot y)$

```

1 ! Function . f90 : Program calls a simple function
2 ! -----
3 Real *8 Function f (x,y)
4   Implicit None
5   Real *8 :: x, y
6   f = 1.0 + sin (x*y )
7 End Function f
8 !
9 Program Main
10  Implicit None
11  Real *8 :: Xin =0.25 , Yin =2. , c , f ! declarations ( also f)
12  c = f ( Xin , Yin )
13  write ( * , * ) 'f(Xin, Yin) = ' , c

```


14 End Program Main



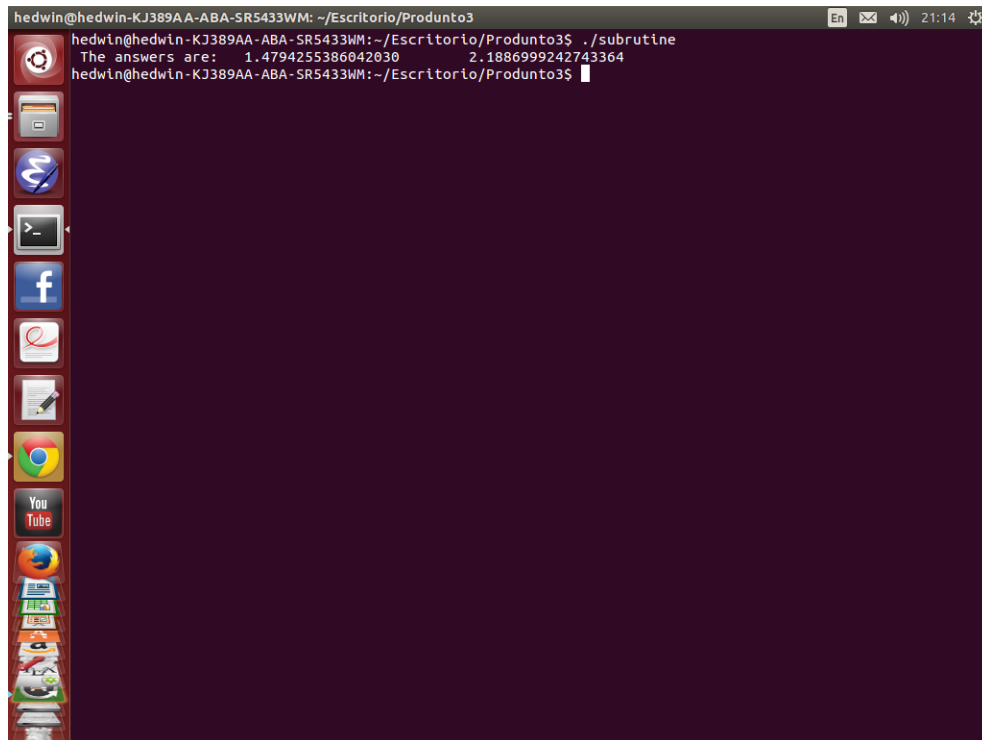
```
hedwin@hedwin-KJ389AA-ABA-SR5433WM: ~/Escritorio/Producto3
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$ ./Funcion
f(Xin, Yin) = 1.4794255386042030
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$
```

8. Subrutinas

Fortran además de funciones, también se manejan subrutinas. El siguiente programa contiene un ejemplo de una subrutin.

```
! Subroutine . f90 : Demonstrates the call for a simple subroutine
! -----
Subroutine g(x, y, ans1 , ans2 )
  Implicit None
  Real (8) :: x , y , ans1 , ans2 ! Declare variables
  ans1 = sin (x*y) + 1. ! Use sine intrinsic func.
  ans2 = ans1**2
End Subroutine g
!
Program Main program ! Demos the CALL
```

```
Implicit None
Real *8 : : Xin =0.25 , Yin =2.0 , Gout1 , Gout2
call g( Xin , Yin , Gout1 , Gout2 ) ! Call the subr g
write ( * , *) 'The answers are: ' , Gout1 , Gout2
End Program Main program
```



The screenshot shows a terminal window with a dark purple background. The title bar at the top reads "hedwin@hedwin-KJ389AA-ABA-SR5433WM: ~/Escritorio/Producto3". The terminal content shows the execution of a Fortran program. The prompt "hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3\$" is followed by the command "./subrutine". The output of the program is displayed on the next line: "The answers are: 1.4794255386042030 2.1886999242743364". The prompt "hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3\$" is shown again on the following line. On the left side of the terminal window, there is a vertical dock containing several application icons, including a file manager, a web browser, and social media icons like Facebook and YouTube.

```
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$ ./subrutine
The answers are: 1.4794255386042030 2.1886999242743364
hedwin@hedwin-KJ389AA-ABA-SR5433WM:~/Escritorio/Producto3$
```