

Reporte sintético: Actividad 3

Ana Gabriela Carretas Talamante

19 de Febrero de 2015

1. Introducción

En esta tercera práctica hemos iniciado a trabajar en el ambiente de **FORTRAN**, escribiendo algunos comandos y programas muy sencillos. Estos permiten la visualización de el ambiente de trabajo en dicho lenguaje de programación, además de introducirnos un poco.

Se mostrarán en la siguiente sección los programas con los que estuvimos trabajando esta semana.

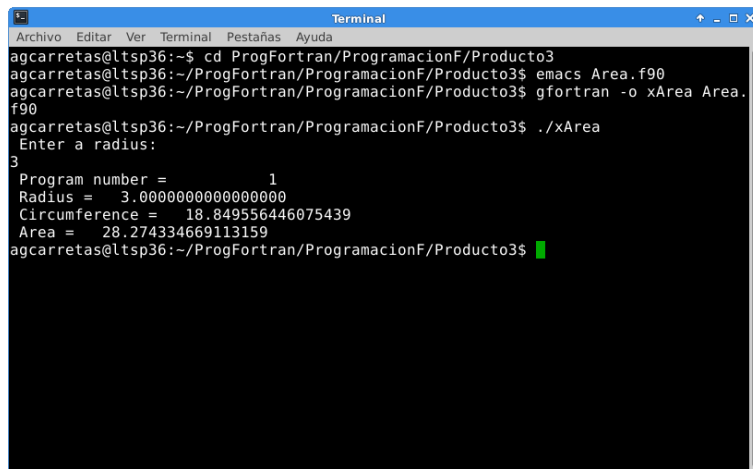
2. Actividades

La estructura de las siguientes subsecciones está conformada por:

- Descripción de la actividad
- Captura de pantalla mostrando su funcionamiento en la terminal
- Código en **FORTRAN**

2.1. Área de un círculo

Este es el primer código que maneja la actividad. La idea es que el programa calcule el área de un círculo, teniendo como entrada la magnitud del radio.

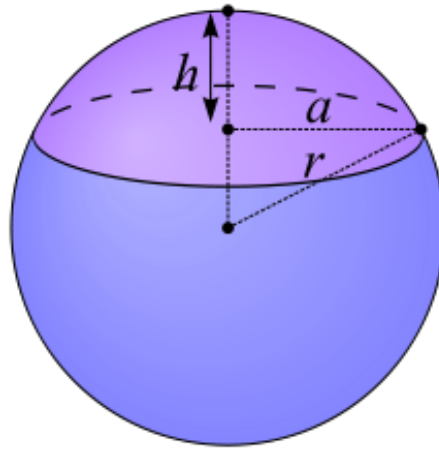


```
Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
agcarretas@ltsp36:~$ cd ProgFortran/ProgramacionF/Producto3
agcarretas@ltsp36:~/ProgFortran/ProgramacionF/Producto3$ emacs Area.f90
agcarretas@ltsp36:~/ProgFortran/ProgramacionF/Producto3$ gfortran -o xArea Area.f90
agcarretas@ltsp36:~/ProgFortran/ProgramacionF/Producto3$ ./xArea
Enter a radius:
3
Program number =      1
Radius =  3.0000000000000000
Circumference =  18.849556446075439
Area =  28.274334669113159
agcarretas@ltsp36:~/ProgFortran/ProgramacionF/Producto3$
```

! Area . f90 : Calcula el area de un circulo

```
Program areacirculo
Implicit None
Real *8 :: radius , circum , area
Real *8 :: PI = 4.0 * atan(1.0)
Integer :: model_n = 1
print * , 'Enter a radius:'
read * , radius
circum = 2.00 * PI * radius
area = radius * radius * PI
print * , 'Program number =' , model_n
print * , 'Radius =' , radius
print * , 'Circumference =' , circum
print * , 'Area =' , area
End Program areacirculo
```

2.2. Volumen de un segmento esférico



La imagen superior muestra gráficamente el volumen que nuestro nuevo código debería calcular. Basándonos en el código del programa anterior, nuestra tarea fue la de rediseñarlo, de manera que mostrara el resultado del volumen de un segmento esférico, teniendo como variables al radio y la altura.

```

Terminal
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda

epsilon_m = 1.0
1
Error: Symbol 'epsilon_m' at (1) has no IMPLICIT type
Precision.f90:9.5:

one = 1.0
1
Error: Symbol 'one' at (1) has no IMPLICIT type
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ emacs Precision.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ emacs Volumen.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ gfortran -o xVolumen Vo
lumen.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ ./xVolumen
Enter a radius:
5
Enter a height:
3
Program number = 1
Radius = 5.0000000000000000
Height = 3.0000000000000000
Adjusted radius = 12.000000000000000
Volume = 113.08602706585532
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$

```

! Volumen . f90 : Calcula el volumen de una region esferica

```

Program volumenregion
Implicit None
Real *8 :: radius , radiusx , volume, height
Real *8 :: PI = 4.0 * atan(1.0)
Integer :: model_n = 1
print * , 'Enter a radius:'
read * , radius

```

```

print * , 'Enter a height:'
read * , height
radiusx = 3.00 * radius - height
volume = 0.3333 * PI * height * height * radiusx
print * , 'Program number =' , model_n
print * , 'Radius =' , radius
print * , 'Height =' , height
print * , 'Adjusted radius =' , radiusx
print * , 'Volume =' , volume
End Program volumenregion

```

2.3. Precisión de una máquina

La actividad consistía en determinar la precisión de la computadora en la que se trabaja, utilizando una herramienta de precisión doble, si esta interactuaba durante sesenta ocasiones.

```

1 1.50000000 0.50000000
2 1.25000000 0.25000000
3 1.12500000 0.12500000
4 1.06250000 6.25000000E-02
5 1.03125000 3.12500000E-02
6 1.01562500 1.56250000E-02
7 1.00781250 7.81250000E-03
8 1.00390625 3.90625000E-03
9 1.00195312 1.95312500E-03
10 1.00097656 9.76562500E-04
11 1.00048828 4.88281250E-04
12 1.00024414 2.44140625E-04
13 1.00012207 1.22070312E-04
14 1.00006104 6.10351562E-05
15 1.00003052 3.05175781E-05
16 1.00001526 1.52587991E-05
17 1.00000763 7.62939453E-06
18 1.00000381 3.81469727E-06
19 1.00000191 1.90734863E-06
20 1.00000095 9.53674316E-07
21 1.00000048 4.76837158E-07
22 1.00000024 2.38418598E-07
23 1.00000012 1.19209299E-07
24 1.00000006 5.96046448E-08
25 1.00000003 2.98023224E-08
26 1.00000001 1.49011612E-08
27 1.00000000 7.45058006E-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 9.09494701E-13
41 1.00000000 4.54747251E-13
42 1.00000000 2.27373625E-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

```

! Precision . f90 : Determina la precision de la computadora

```

Program Precision
  Implicit None
  Integer :: i , n
  Real *8 :: epsilon_m , one
  n=60
  epsilon_m = 1.0
  one = 1.0

  do i = 1, n , 1
    epsilon_m = epsilon_m / 2.0
    one = 1.0 + epsilon_m

```

```

    print * , i , one , epsilon_m
end do
End Program Precision

```

2.4. Precisión de una máquina 2.0

Este código fue retomado del anterior, con el simple hecho de cambiarle la precisión a sencilla.

```

1 1.50000000 0.50000000
2 1.25000000 0.25000000
3 1.12500000 0.12500000
4 1.06250000 6.2500000E-02
5 1.03125000 3.1250000E-02
6 1.01562500 1.5625000E-02
7 1.00781250 7.8125000E-03
8 1.00390625 3.9062500E-03
9 1.00195312 1.9531250E-03
10 1.00097656 9.7656250E-04
11 1.00048828 4.8828125E-04
12 1.00024414 2.4414062E-04
13 1.00012207 1.2207031E-04
14 1.00006104 6.1035156E-05
15 1.00003052 3.0517578E-05
16 1.00001526 1.5258789E-05
17 1.00000763 7.6293945E-06
18 1.00000381 3.8146972E-06
19 1.00000191 1.9073486E-06
20 1.00000095 9.5367431E-07
21 1.00000048 4.7683715E-07
22 1.00000024 2.3841857E-07
23 1.00000012 1.1920929E-07
24 1.00000006 5.9604644E-08
25 1.00000003 2.9802322E-08
26 1.00000001 1.4901161E-08
27 1.00000000 7.4505806E-09
28 1.00000000 3.7252903E-09
29 1.00000000 1.8626451E-09
30 1.00000000 9.3132257E-10
31 1.00000000 4.6566128E-10
32 1.00000000 2.3283064E-10
33 1.00000000 1.1641532E-10
34 1.00000000 5.8207660E-11
35 1.00000000 2.9103830E-11
36 1.00000000 1.4551915E-11
37 1.00000000 7.2759576E-12
38 1.00000000 3.6379788E-12
39 1.00000000 1.8189894E-12
40 1.00000000 9.0949470E-13
41 1.00000000 4.5474751E-13
42 1.00000000 2.2737375E-13
43 1.00000000 1.1368638E-13
44 1.00000000 5.6843189E-14
45 1.00000000 2.8421709E-14
46 1.00000000 1.4210854E-14

```

! Precision . f90 : Determina la precision de la computadora

```

Program Precision
  Implicit None
  Integer :: i , n
  Real *8 :: epsilon_m , one
  n=60
  epsilon_m = 1.0
  one = 1.0

  do i = 1, n , 1
    epsilon_m = epsilon_m / 2.0
    one = 1.0 + epsilon_m
    print * , i , one , epsilon_m
  end do
End Program Precision

```

2.5. Funciones matemáticas

El objetivo era el exponer por medio de variables, que **FORTTRAN** tiene ciertas funciones matemáticas predefinidas. Se calcularon ciertos valores senoidales y exponenciales.

```

Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
End Program Math test
1
Error: Syntax error in END PROGRAM statement at (1)
Error: Unexpected end of file in 'Math.f90'
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ emacs Math.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ gfortran -o xMath Math.f90
Math.f90:3.7:

Program Math test
1
Error: Invalid form of PROGRAM statement at (1)
Math.f90:8.16:

End Program Math test
1
Error: Syntax error in END PROGRAM statement at (1)
Error: Unexpected end of file in 'Math.f90'
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ emacs Math.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ gfortran -o xMath Math.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ ./xMath
1.0000000000000000 0.84147098480789650 3.7182818284590451
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$

```

! Math . f90 : demo de algunas funciones matematicas en Fortran

```

Program Mathtest
  Real *8 :: x = 1.0 , y, z
  y = sin (x)
  z = exp (x) + 1.0
  print * , x, y, z
End Program Mathtest

```

2.6. Otro tipo de funciones matemáticas

Continuando con expresiones matemáticas predefinidas, se presentaron como operaciones que manejaban resultados en los complejos. O infinitos.

```

Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
x MathComplex.f90
MathComplex.f90:5.12:

  x = sqrt (-1.0)
1
Error: Argument of SQRT at (1) has a negative value
MathComplex.f90:7.11:

  z = log (0)
1
Error: 'x' argument of 'log' intrinsic at (1) must be REAL or COMPLEX
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ emacs MathComplex.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ gfortran -o xMathComplex MathComplex.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ ./xMathComplex
( 0.00000000 , 1.00000000 ) ( 1.10714877 , 0.00000000 ) (
-Infinity, 0.00000000 )
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ emacs MathComplex.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ gfortran -o xMathComplex MathComplex.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ ./xMathComplex
( 0.00000000 , 1.00000000 ) ( 1.10714877 , 0.00000000 ) (
-Infinity, 0.00000000 )
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$

```

! Mathcomplex . f90 : demo de algunas funciones matematicas en Fortran

```

Program Mathcomplex

```

```

Complex *8 :: x = -1.0 , y = 2.0 , z = 0 , xx , yy , zz
xx = sqrt (x)
yy = atan (y)
zz = log (z)
print * , xx, yy, zz
End Program Mathcomplex

```

2.7. Declarando nuevas funciones

Declaramos una función que denotaba cierta trigonométrica dependiendo de dos variables, que las introducirá el usuario. Después, se incluyó esa función en un programa que calculaba valores influenciados por la anterior función propuesta.

```

Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
Real \xE2\x88\x97 : : Xin =0.25 , Yin =2. , c , f
1
Error: Invalid character in name at (1)
Funcion.f90:12.3:

  c = f ( Xin , Yin )
1
Error: Symbol 'c' at (1) has no IMPLICIT type
Funcion.f90:12.13:

  c = f ( Xin , Yin )
1
Error: Symbol 'xin' at (1) has no IMPLICIT type
Funcion.f90:12.19:

  c = f ( Xin , Yin )
1
Error: Symbol 'yin' at (1) has no IMPLICIT type
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ emacs Funcion.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ gfortran -o xFuncion Funcion.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ ./xFuncion
f(Xin, Yin) = 1.4794255386042030
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$

```

! Funcion . f90 : Creando funciones

```

Real *8 Function f (x,y)
  Implicit None
  Real *8 :: x, y
  f = 1.0 + sin (x*y )
End Function f

```

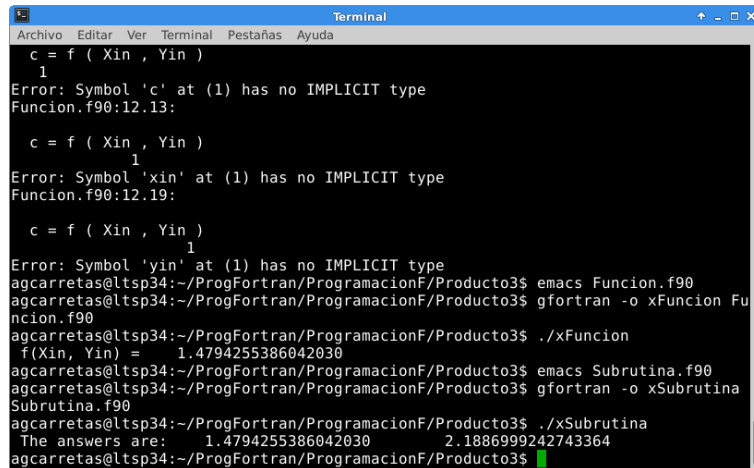
```

Program Main
  Implicit None
  Real *8 :: Xin =0.25 , Yin =2. , c , f
  c = f ( Xin , Yin )
  write ( * , * ) 'f(Xin, Yin) = ' , c
End Program Main

```

2.8. Subrutinas

Las subrutinas son subprogramas que no devuelven ningún resultado, por tanto no tienen tipo. Las subrutinas son una herramienta de FORTRAN para simplificar el trabajo, pues llama a elementos de otras funciones, y les agrega más información cada que son utilizadas.



```
Terminal
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda

c = f ( Xin , Yin )
1
Error: Symbol 'c' at (1) has no IMPLICIT type
Funcion.f90:12.13:

c = f ( Xin , Yin )
1
Error: Symbol 'xin' at (1) has no IMPLICIT type
Funcion.f90:12.19:

c = f ( Xin , Yin )
1
Error: Symbol 'yin' at (1) has no IMPLICIT type
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ emacs Funcion.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ gfortran -o xFuncion Funcion.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ ./xFuncion
f(Xin, Yin) = 1.4794255386042030
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ emacs Subrutina.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ gfortran -o xSubrutina Subrutina.f90
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$ ./xSubrutina
The answers are: 1.4794255386042030 2.1886999242743364
agcarretas@ltsp34:~/ProgFortran/ProgramacionF/Producto3$
```

! Subrutina . f90 : Demuestra la llamada de una subrutina

```
Subroutine g(x, y, ans1 , ans2 )
  Implicit None
  Real (8) :: x , y , ans1 , ans2
  ans1 = sin (x*y) + 1.
  ans2 = ans1**2
End Subroutine g
```

```
Program Mainprogram
  Implicit None
  Real *8 :: Xin =0.25 , Yin =2.0 , Gout1 , Gout2
  call g( Xin , Yin , Gout1 , Gout2 )
  write ( * , *) 'The answers are: ' , Gout1 , Gout2
End Program Mainprogram
```