

# Iniciando con FORTRAN

Luisa Fernanda Orci Fernandez.

17 de Febrero del 2015

## 1. Introducción

Esta actividad está conformada por una serie de ejercicios breves en Fortran, que nos sirven para calcular áreas, volúmenes, así como algunos ejemplos de como resolver funciones.

## 2. Calcular el área de un círculo.

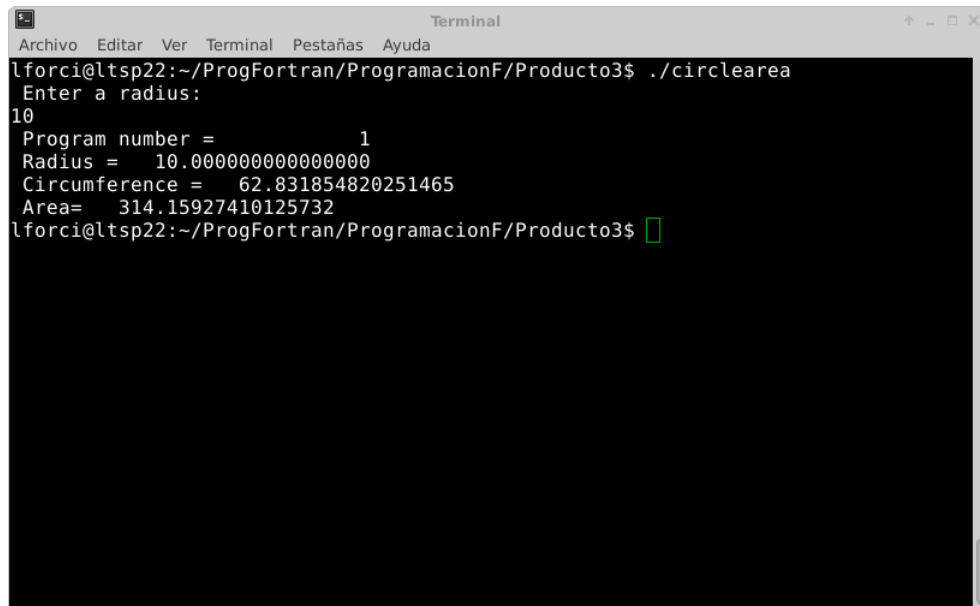
Con este programa se puede calcular facilmente el área de un círculo. El programa está diseñado para que el usuario de un radio y a partir de eso dar el resultado del área.

A continuación, el código utilizado para calcular el área, seguido del resultado al compilarlo.

```
! 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, assing Real
  Integer :: model_n = 1 ! Declare, assing 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
```



```
Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
lforci@ltsp22:~/ProgFortran/ProgramacionF/Producto3$ ./circlearea
Enter a radius:
10
Program number =          1
Radius = 10.000000000000000
Circumference = 62.831854820251465
Area= 314.15927410125732
lforci@ltsp22:~/ProgFortran/ProgramacionF/Producto3$
```

### 3. Calcular el volumen de un tanque esférico de radio $r$ y altura $h$ .

Este programa sirve para calcular el volumen de un tanque que no está completamente lleno. Al usuario se le pide que proporcione un radio y la altura hasta donde está lleno el tanque.

A continuación, el código utilizado para calcular el volumen, seguido del resultado al compilarlo.

```
! Volumen.f90: Calculates a spherical cap
!-----

Program Volumen
  Implicit None
  ! Declarar mis variables, NO CALCULAR NADA
  Real *8 :: radio, circum, PI, vol, h
  Integer :: model_n = 2

  ! EPEZAR A realizar calculos y demas
  PI= 4.0 * atan(1.0)
  print *, 'Dame un radio'
  read *, radio
  print *, 'Dame una altura'
  read *, h
```

```

circum= 2.0 * PI * radio
vol= (PI*(h*h))*(radio-(h/3))
print *, 'Program number = ' , model_n
print *, 'Radio =' , radio
print *, 'Altura=' , h
print *, 'Circunferencia =' , circum
print *, 'Volumen =' , vol

```

End

```

File Edit View Search Terminal Help
[luisa@leah Producto3]$ ./Volumen
Dame un radio
5
Dame una altura
10
Program number =      2
Radio =  5.000000000000000
Altura=  10.000000000000000
Circunferencia =  31.415927410125732
Volumen =  523.59879016876221
[luisa@leah Producto3]$

```

#### 4. Determinar la precisión de la máquina.

Este programa sirve para determinar la precisión de la máquina. A continuación, el código y un ejemplo de lo que sucede al compilar.

```

! 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

```

```

lforci@ltsp22:~/ProgFortran/ProgramacionF/Producto3$ ./Limits
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  1.52587891E-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  2.38418579E-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

```

## 5. Funciones matemáticas especiales.

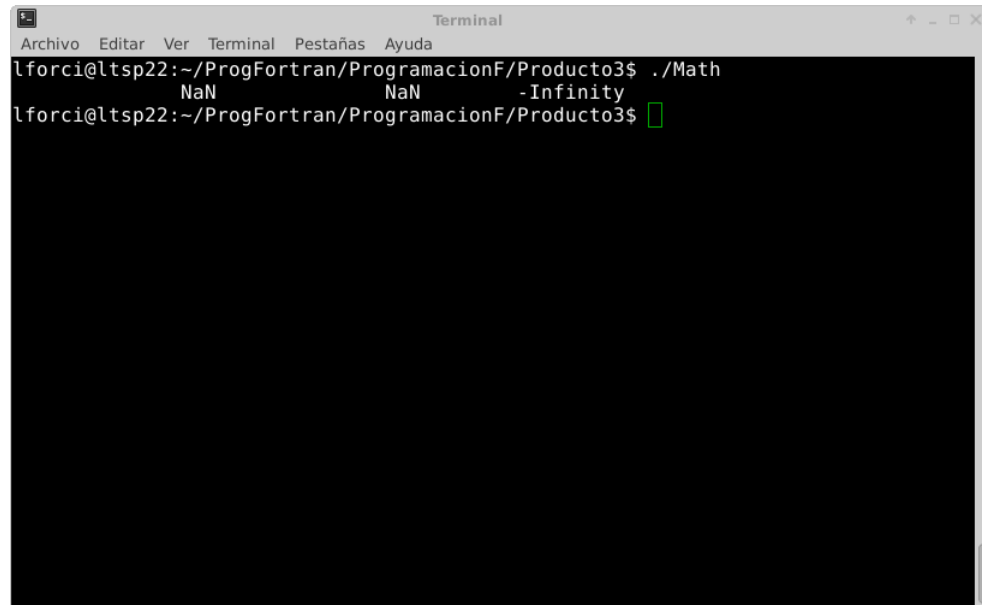
Fortran maneja las funciones especiales y las trigonométricas, pero no maneja la relación entre ellas.

A continuación un ejemplo para calcular la raíz cuadrada de -1, el arcoseno de 2 y el log10 de 0.

```
! Math . f90: demo some Fortran math functions
! -----

Program Math_test                ! Begin main program
  Real *8 :: x=-1.0, y=2.0, z=0  ! Declare variables x, y, z
  v = SQRT (x)                  ! Call the Square root function
  w = ASIN (y)                  ! Call the Arcsine function
  f = LOG10 (z)                 ! Call the Common logarithm function
  print *, v, w, f              ! Print x, y, z

End Program Math_test            ! End main program
```



```
Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
lforci@ltsp22:~/ProgFortran/ProgramacionF/Producto3$ ./Math
NaN NaN -Infinity
lforci@ltsp22:~/ProgFortran/ProgramacionF/Producto3$
```

## 6. Calcular Funciones.

En Fortran también nos sirve para calcular funciones.

A continuación un ejemplo de como calcular la funcion  $f(x) = 1 + \sin(x)$ .

```
! Function . f90: Program calls a simple function
! -----
```

```

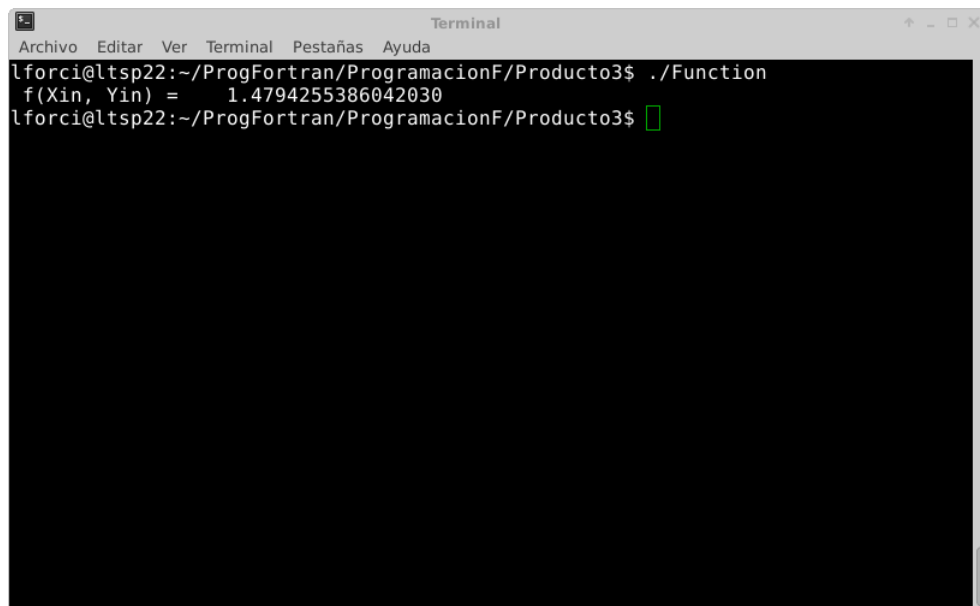
Real*8 Function f(x,y)
  Implicit None
  Real*8 :: x, y
  f = 1.0 + sin(x*y)
End Function f

!

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

End program Main

```



```

Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
lforci@ltsp22:~/ProgFortran/ProgramacionF/Producto3$ ./Function
f(Xin, Yin) = 1.4794255386042030
lforci@ltsp22:~/ProgFortran/ProgramacionF/Producto3$

```

## 7. Subrutinas.

En Fortran también podemos manejar subrutinas, estas sirven para encapsular y reutilizar funciones específicas.

A continuación un programa que contiene un ejemplo de subrutina.

```

! Subroutine . f90: Demonstrates the call for a simple subroutine
! -----
Subroutine g(x, y, ans1, ans2)

```

```

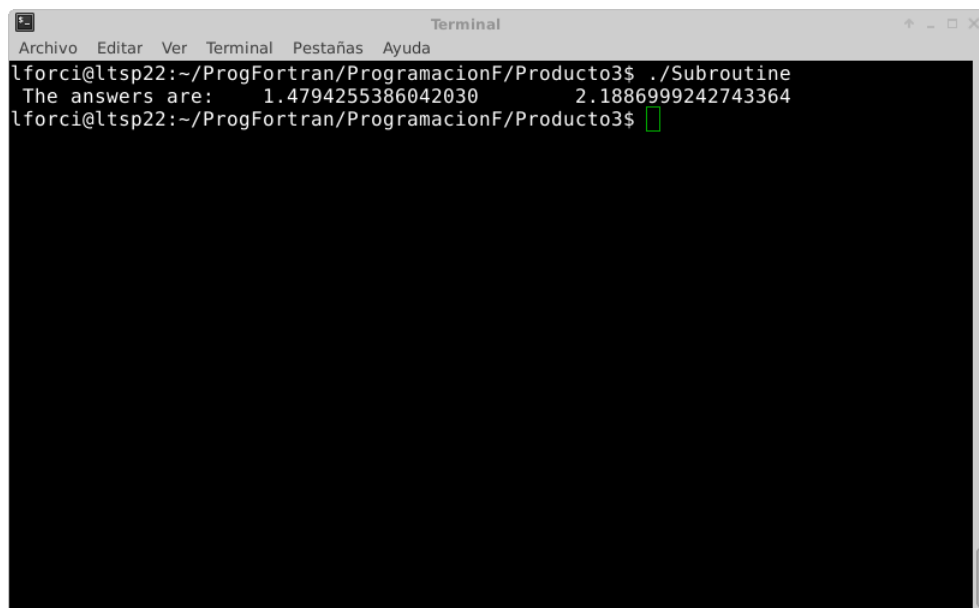
Implicit None
Real(8) :: x, y, ans1, ans2 ! Declare variables
ans1= sin(x*y) + 1.          ! Use sine intrinsic func.
ans2= ans1**2
End Subroutine g

!

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

```



```

Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
lforci@ltsp22:~/ProgFortran/ProgramacionF/Producto3$ ./Subroutine
The answers are: 1.4794255386042030 2.1886999242743364
lforci@ltsp22:~/ProgFortran/ProgramacionF/Producto3$

```