

# Reporte Actividad 3

Antonio Cota Rodríguez

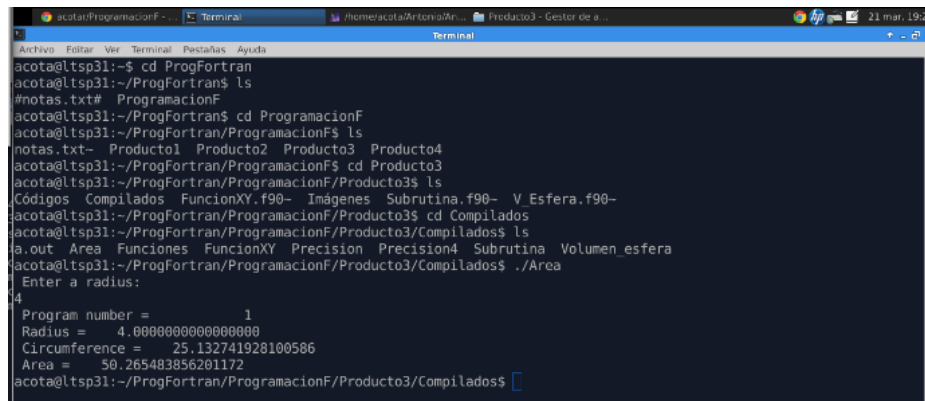
## Introducción

Con la presente práctica su objetivo es introducirnos al lenguaje de programación Fortran. El siguiente reporte explica las actividades realizadas con este propósito, se describe brevemente cada una, se añade una imagen de la ejecución y se muestra su respectivo código.

## Actividades

### Area

Con el siguiente programa podremos calcular el área de un círculo dado su radio.

A screenshot of a terminal window titled 'Terminal'. The user 'acota@ltsp31' is in the directory '~/ProgFortran'. They run 'ls' showing 'notas.txt' and 'ProgramacionF'. Then they run 'cd ProgramacionF' and 'ls' showing 'Producto1', 'Producto2', 'Producto3', and 'Producto4'. They then run 'cd Producto3' and 'ls' showing 'Códigos', 'Compilados', 'FuncionXY.f90', 'Imágenes', 'Subrutina.f90', and 'V\_Esfera.f90'. They run 'cd Compilados' and 'ls' showing 'a.out', 'Area', 'Funciones', 'FuncionXY', 'Precision', 'Precision4', 'Subrutina', and 'Volumen\_esfera'. Finally, they run './Area', which prompts 'Enter a radius:' and they input '4'. The program outputs: 'Program number = 1', 'Radius = 4.0000000000000000', 'Circumference = 25.132741928100586', and 'Area = 50.265483856201172'.

```
acota@ltsp31:~$ cd ProgFortran
acota@ltsp31:~/ProgFortran$ ls
notas.txt  ProgramacionF
acota@ltsp31:~/ProgFortran$ cd ProgramacionF
acota@ltsp31:~/ProgFortran/ProgramacionF$ ls
notas.txt  Producto1  Producto2  Producto3  Producto4
acota@ltsp31:~/ProgFortran/ProgramacionF$ cd Producto3
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3$ ls
Códigos  Compilados  FuncionXY.f90  Imágenes  Subrutina.f90  V_Esfera.f90
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3$ cd Compilados
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$ ls
a.out  Area  Funciones  FuncionXY  Precision  Precision4  Subrutina  Volumen_esfera
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$ ./Area
Enter a radius:
4
Program number =      1
Radius =  4.0000000000000000
Circumference =  25.132741928100586
Area =  50.265483856201172
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$
```

```
! Area.f90 : Calculates the area of the 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
```

```

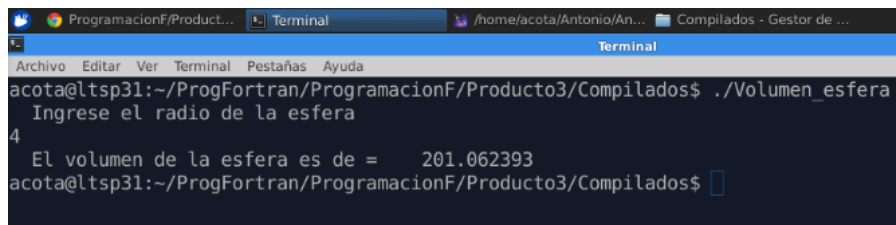
Integer :: model_n = 1 ! Declare , assign Ints
print * , "Enter a radius:" ! Talk to user
read * , radius ! Read into radius
circum = 2.0 * PI * radius ! Calc circumference
area = radius * radius * PI ! Calc area
print * , "Program number = " , model_n ! Print program number
print * , "Radius = " , radius ! Print radius
print * , "Circumference = " , circum ! Print circumference
print * , "Area = " , area ! Print Area

End Program Circle_area ! End main program code

```

## Volumen

Calcular el volumen de un líquido en un tanque esférico cuando el líquido está a una altura  $h$  del fondo del tanque. En la ejecución se define el radio y la altura  $h$ .



```

acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$ ./Volumen_esfera
Ingrese el radio de la esfera
4
El volumen de la esfera es de = 201.062393
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$

```

```

! Objetivo del programa:
! Obtener el volumen de una esfera dado el radio por el usuario

PROGRAM Volumen_Esfera

IMPLICIT NONE

! Declaracion de variables

REAL :: Volumen
REAL :: radio
REAL, PARAMETER :: PI = 3.1416

!-----

! Obtener el valor del radio

Write (*,*) " Ingrese el radio de la esfera "

```

```

READ (*,*) radio

!-----

! Calcular el volumen

Volumen = (4/3) * PI * radio**3

!-----

! Brindar la informacion

WRITE (*,*) " El volumen de la esfera es de = ", Volumen

!-----

END PROGRAM Volumen_Esfera

```

## Precisión

Este programa determina la precisión de la máquina. Se compara repetidamente  $1 + \epsilon_m$  con 1 a medida que  $\epsilon_m$  se vuelve mas pequeño y se muestra como impacta en la precisión del cálculo.

```

ProgramacionFProduct... Terminal
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$ ./Precision
1 1.5000000000000000 0.5000000000000000
2 1.2500000000000000 0.2500000000000000
3 1.1250000000000000 0.1250000000000000
4 1.0625000000000000 6.250000000000000E-002
5 1.0312500000000000 3.125000000000000E-002
6 1.0156250000000000 1.562500000000000E-002
7 1.0078125000000000 7.812500000000000E-003
8 1.0039062500000000 3.906250000000000E-003
9 1.0019531250000000 1.953125000000000E-003
10 1.0009765625000000 9.765625000000000E-004
11 1.0004882812500000 4.882812500000000E-004
12 1.0002441406250000 2.441406250000000E-004
13 1.0001220703125000 1.220703125000000E-004
14 1.0000610351562500 6.103515625000000E-005
15 1.0000305175781250 3.051757812500000E-005
16 1.0000152587890625 1.525878906250000E-005
17 1.0000076293945312 7.629394531250000E-006
18 1.0000038146972656 3.814697265625000E-006
19 1.0000019073486328 1.907348632812500E-006
20 1.0000009536743164 9.536743164062500E-007
21 1.0000004768371582 4.768371582031250E-007
22 1.0000002384185791 2.384185791015625E-007
23 1.0000001192092896 1.192092895507812E-007
24 1.0000000596046448 5.960464477539062E-008
25 1.0000000298023224 2.980232238769531E-008
26 1.0000000149011612 1.490116119384765E-008
27 1.0000000074505806 7.450580596923828E-009
28 1.0000000037252903 3.725290298461914E-009
29 1.0000000018626451 1.862645149230957E-009
30 1.0000000009313226 9.313225746154785E-010
31 1.0000000004656613 4.656612873077392E-010
32 1.0000000002328306 2.328306436538696E-010
33 1.0000000001164153 1.164153218269348E-010
34 1.0000000000582077 5.820766091346740E-011
35 1.0000000000291038 2.910383045673370E-011
36 1.0000000000145519 1.455191522836685E-011
37 1.0000000000072760 7.275957614183425E-012
38 1.0000000000036380 3.637978807091713E-012

```

```

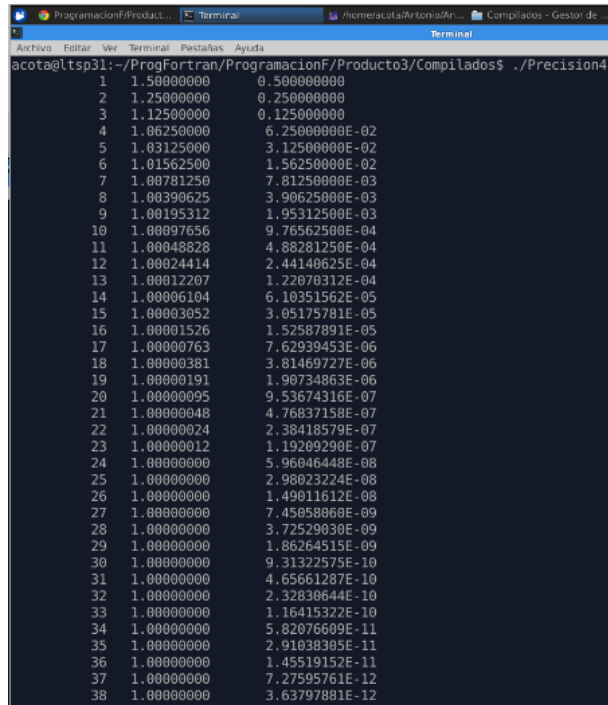
! Limits.f90 : Determines machines precision
!-----

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 form 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 ! Recalculate one
print * , i , one , epsilon_m ! Print values so far
end do ! End loop when i > n
End program Limits

```

## Precisión4

Se modifica el programa anterior cambiando la precisión a sencilla.



```
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$ ./Precision4
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 1.00000006 5.96046448E-08
25 1.00000003 2.98023224E-08
26 1.00000001 1.49011612E-08
27 1.00000000 7.45058860E-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 1.45519157E-11
37 1.00000000 7.27595761E-12
38 1.00000000 3.63797881E-12
```

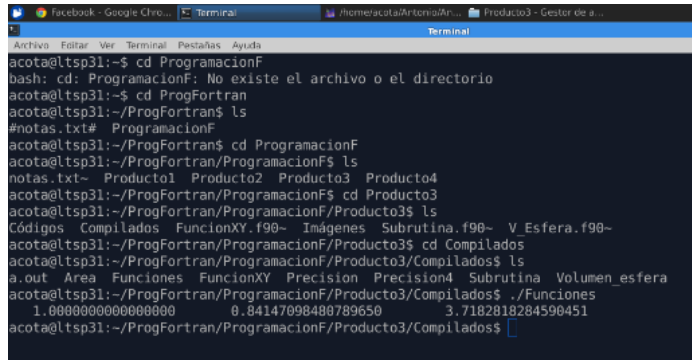
```
! Limits.f90 : Determines machines precision
!-----

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 form 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 ! Recalculate one
print *, i , one , epsilon_m ! Print values so far
end do ! End loop when i > n
```

```
End program Limits
```

## Funciones

Se muestra el uso de algunas funciones matematicas en Fortran y se imprimen en la pantalla de terminal los ejemplos de sus valores.



```
acota@ltsp31:~$ cd ProgramacionF
bash: cd: ProgramacionF: No existe el archivo o el directorio
acota@ltsp31:~$ cd ProgFortran
acota@ltsp31:~/ProgFortran$ ls
#notas.txt# ProgramacionF
acota@ltsp31:~/ProgFortran$ cd ProgramacionF
acota@ltsp31:~/ProgFortran/ProgramacionF$ ls
notas.txt~ Producto1 Producto2 Producto3 Producto4
acota@ltsp31:~/ProgFortran/ProgramacionF$ cd Producto3
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3$ ls
Códigos Compilados FuncionXY.f90~ Imágenes Subrutina.f90~ V.Esfera.f90~
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3$ cd Compilados
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$ ls
a.out Area Funciones FuncionXY Precision Precision4 Subrutina Volumen_esfera
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$ ./Funciones
1.0000000000000000 0.84147098480789650 3.7182818284590451
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$
```

```
! 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
```

## Mathtest2

Se expone el uso de variables complejas para obtener los valores que se pide.

```
Program Mathtest2

COMPLEX, PARAMETER :: MINUS_ONE = -1.0, x= 2.0
COMPLEX :: Z,y
Real *8 :: w, q=1.0
```

```

Z = SQRT(MINUS_ONE)
y=asin(x)
w=log(q-1)

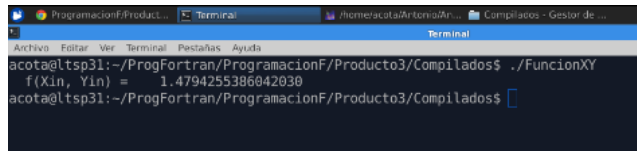
Print * , Z, y, w

End Program Mathtest2

```

## FunciónXY

Aquí se muestra el uso de funciones definidas en el código.



```

acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$ ./FuncionXY
f(Xin, Yin) = 1.4794255386642030
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$

```

```

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=2.0 , c, f

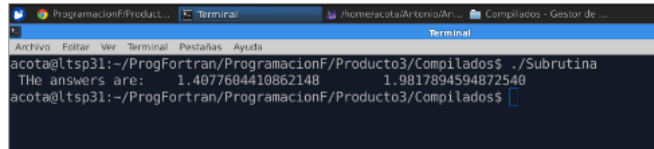
  c=f(Xin,Yin)
  write (*,*) 'f(Xin,Yin)=', c

End Program Main

```

## Subrutina

Y un ejemplo del manejo de subrutinas. En el código, se escribe la rutina a realizar antes de iniciar el programa, la rutina se llama al momento de requerirla especificando los parámetros.



```
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$ ./Subrutina
The answers are: 1.4077604410862148 1.9817894594872540
acota@ltsp31:~/ProgFortran/ProgramacionF/Producto3/Compilados$
```

```
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 Main_program

  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 Main_program
```