

# Iniciando con Fortran

Valenzuela Coronado Paulina

Licenciatura en Física

## 1. Actividad 3

Esta actividad se baso en realizar una serie de programas que solucionaban problemas matemáticos de una forma más sencilla y rápida para el usuario. Estos programas se realizaron usando el lenguaje Fortran.

## 2. A continuacion se presentan distintos códigos con los que trabajamos:

- **Área de un círculo:** En este programa se pide al usuario el radio del círculo con el cual se realizaran los cálculos de la circunferencia y área del círculo.

```
! Area . f90 : Calculates the area of a circle, sample program
```

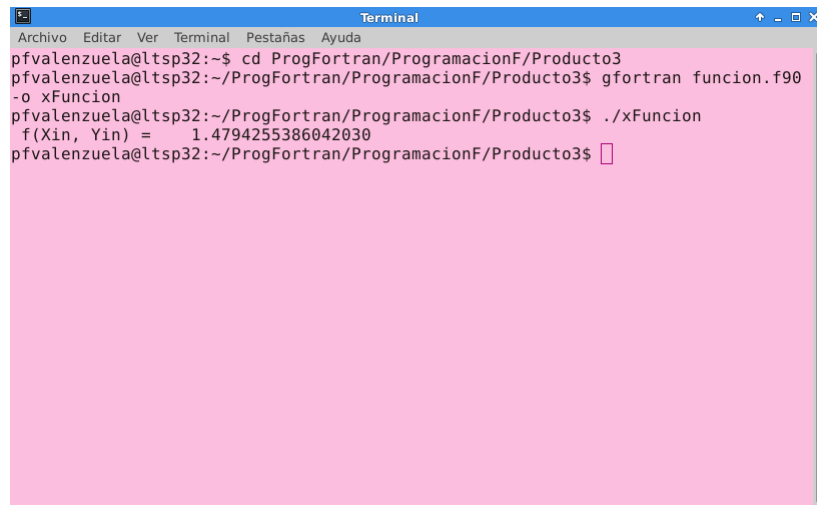
```
Program areadelcirculo ! Begin main program
Implicit None ! Declare all variables
Real *8 :: radius , circum , area ! Declare Reals
Real *8 :: PI = 4.0 * atan(1.0) ! Declare , assign Real
Integer :: model_n = 1 ! Declare , assign Ints
print * , 'Enter a radius:' ! Talk to user
read * , radius ! Read into radius
circum = 2.00 * PI * radius ! Calc circumference
area = radius * radius * PI ! Calc area
```



```
Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
pfvalenzuela@ltsp32:~$ cd ProgFortran/ProgramacionF/Producto3
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ gfortran area.f90 -o xArea
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ ./xArea
Enter a radius:
3
Program number =          1
Radius = 3.0000000000000000
Circumference = 18.849556446075439
Area = 28.274334669113159
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$
```

```
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 areadelcirculo ! End main program code
```

- **Función:** Este programa calcula el valor de una función  $f(x, y) = 1 + \sin(x \text{ y})$  definida por el usuario.

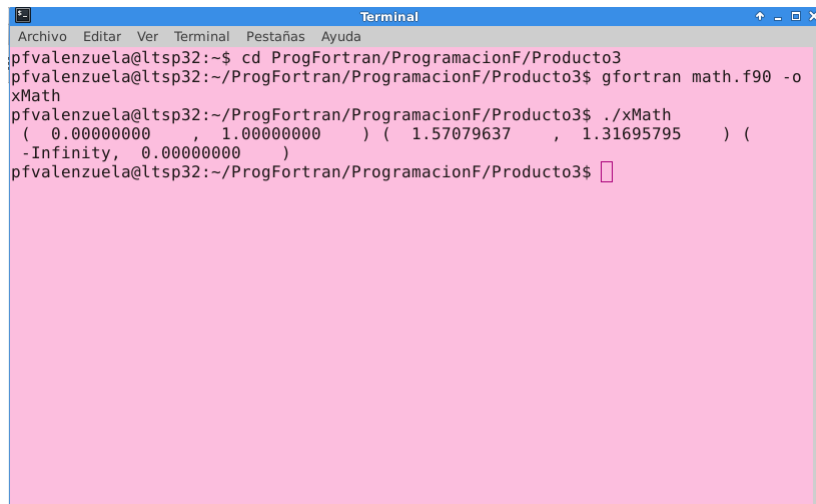


```
Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
pfvalenzuela@ltsp32:~$ cd ProgFortran/ProgramacionF/Producto3
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ gfortran funcion.f90 -o xFuncion
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ ./xFuncion
f(Xin, Yin) = 1.4794255386042030
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$
```

```
! 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
```

- **Math:** El lenguaje Fortran maneja funciones trigonométricas y las especiales. Lo que se realizó fue evaluar diferentes funciones e imprimir un resultado.



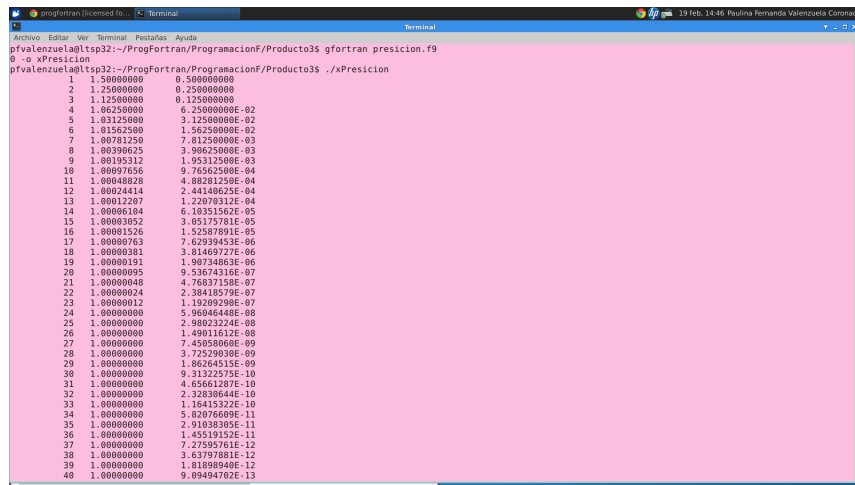
```
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda
pfvalenzuela@ltsp32:~$ cd ProgFortran/ProgramacionF/Producto3
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ gfortran math.f90 -o
xMath
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ ./xMath
( 0.00000000 , 1.00000000 ) ( 1.57079637 , 1.31695795 ) (
-Infinity, 0.00000000 )
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$
```

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

Program Math! Begin main program

```
Complex *8 :: x=- 1.0 , y=2.0, z=0 ! Declare variables x, y, z
x = sqrt (x)
y = asin (y) ! Call the asine function
z = log (z) ! Call the log function
print * , x, y, z ! Print x, y, z
End Program Math ! End main program
```

- Precisión: Este programa determina la precisión de la máquina.



```
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 4.76837150E-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 7.45058060E-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 1.81898948E-12
40 1.00000000 9.09494702E-13
```

! 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

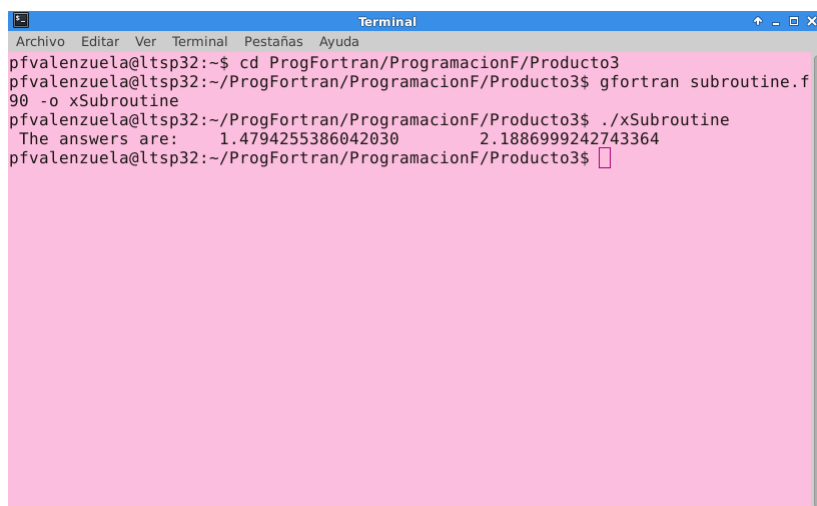
! incremented from 1 to n ( since n = 60) :

```

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

```

- **Subroutine:** Fortran además de funciones, también se manejan subrutinas. El siguiente programa contiene un ejemplo de una subrutina.



```

Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
pfvalenzuela@ltsp32:~$ cd ProgFortran/ProgramacionF/Producto3
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ gfortran subroutine.f
90 -o xSubroutine
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ ./xSubroutine
The answers are: 1.4794255386042030 2.1886999242743364
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$

```

```

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

```

```

Program Main
  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

- **Volumen:** Este programa calcula el volumen de líquido en un tanque esférico de radio R, en el caso en que el nivel de líquido se encuentre a una altura H medida desde el fondo del tanque.



```

Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
pfvalenzuela@ltsp32:~$ cd ProgFortran/ProgramacionF/Producto3
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ gfortran volumen.f90
-o xVolumen
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ ./xVolumen
Enter a Radius:
3
Enter a Height:
6
Program number =      1
Radius =    3.0000000000000000
Height =    6.0000000000000000
Volume =   112.98424279385955
pfvalenzuela@ltsp32:~/ProgFortran/ProgramacionF/Producto3$ 

```

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

Program volumendelaesfera! Begin main program

Implicit None ! Declare all variables

Real \*8 :: radius , height , newradius, volume ! 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

print \* , 'Enter a Height:' ! Talk to user

read \* , height ! Take the value of h

newradius = 3.00 \* radius - height

volume = 0.333 \* PI \* height \* height \* newradius ! Calc Volume

```
print * , 'Program number =' , model_n ! Print program number
print * , 'Radius =' , radius ! Print radius
print * , 'Height =' , height ! Print height
print * , 'Volume =' , volume ! Print volume
End Program volumendelaesfera ! End main program code
```