

Reporte Actividad 3

Andrés Rodríguez

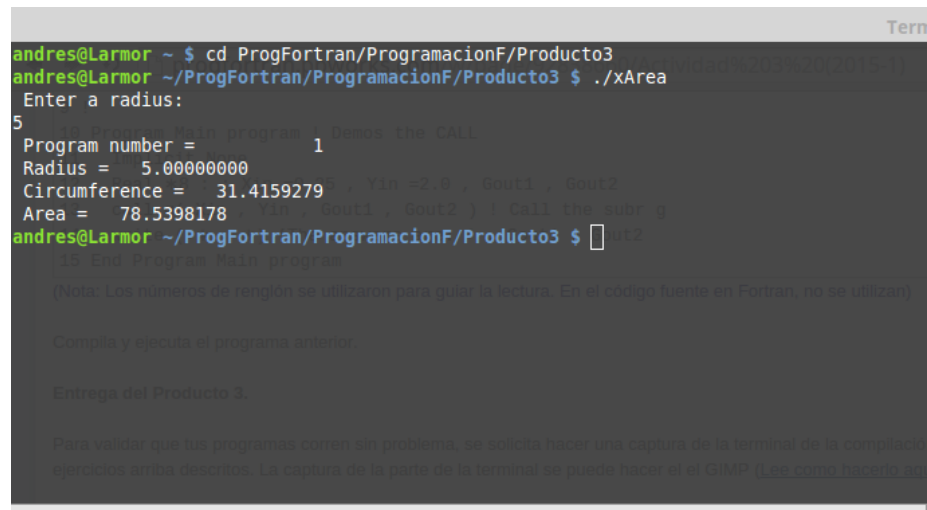
Introducción

El objetivo en esta práctica es introducirnos al lenguaje de programación Fortran. El presente reporte detalla las actividades realizadas con este propósito, se describe brevemente cada una, se añade una imagen de la ejecución y se muestra el código correspondiente.

Actividades

Area

Calcular el área de un círculo, especificando el radio durante la ejecución.



```
andres@Larmor ~ $ cd ProgFortran/ProgramacionF/Producto3
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ ./xArea
Enter a radius:
5
14 Program Main program : Demos the CALL
Program number = 1
Radius = 5.00000000
Circumference = 31.4159279
Area = 78.5398178
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $
```

Program Circlearea

Implicit None

Real :: radius , circum , area

```

Real :: PI = 4.0 * atan(1.0)
Integer :: model_n = 1

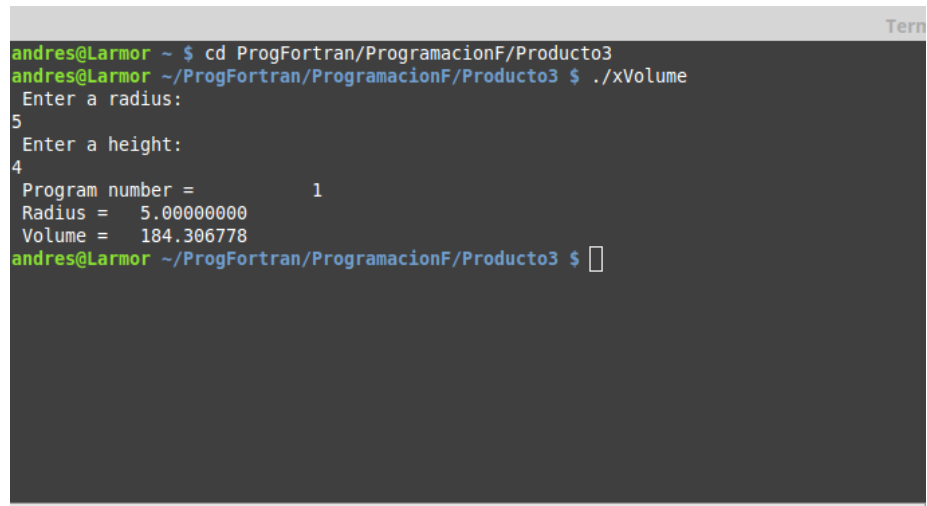
print *, 'Enter a radius:'
read *, radius
circum = 2.0 * PI * radius
area = radius * radius * PI
print *, 'Program number =', model_n
print *, 'Radius =', radius
print *, 'Circumference =', circum
print *, 'Area =', area

End Program Circlearea

```

Volume

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 .



```

andres@Larmor ~ $ cd ProgFortran/ProgramacionF/Producto3
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ ./xVolume
Enter a radius:
5
Enter a height:
4
Program number =          1
Radius =    5.00000000
Volume =   184.306778
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ 

```

```

Program Volume

Implicit NONE

Real :: radius ,h, vol
Real :: PI = 4.0 * atan(1.0)
Integer :: model_n = 1

```

```
print *, 'Enter a radius:'  
read *, radius  
  
print *, 'Enter a height:'  
read *, h  
  
vol = PI *h*h*(3*radius - h ) / 3  
  
print * , 'Program number =' , model_n  
print * , 'Radius =' , radius  
print * , 'Volume =' , vol  
  
End Program Volume
```

Limits

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

```

Terminado
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ ./xLimits
 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
39  1.0000000000018190  1.818989403545856E-012
40  1.0000000000009095  9.094947017729282E-013
41  1.0000000000004547  4.547473508864641E-013

```

```

Program Limits
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 Limits
```

Limits 4

Se modifica el programa anterior cambiado la precisión a sencilla.

```
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ ./xLimits4
```

1	1.50000000	0.500000000
2	1.25000000	0.250000000
3	1.12500000	0.125000000
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

```
Program Limits
```

```
Implicit None
```

```
Integer :: i, n
```

```
Real*4 :: epsilon_m, one
```

```
n=60
```

```

epsilon_m = 1.0
one = 1.0

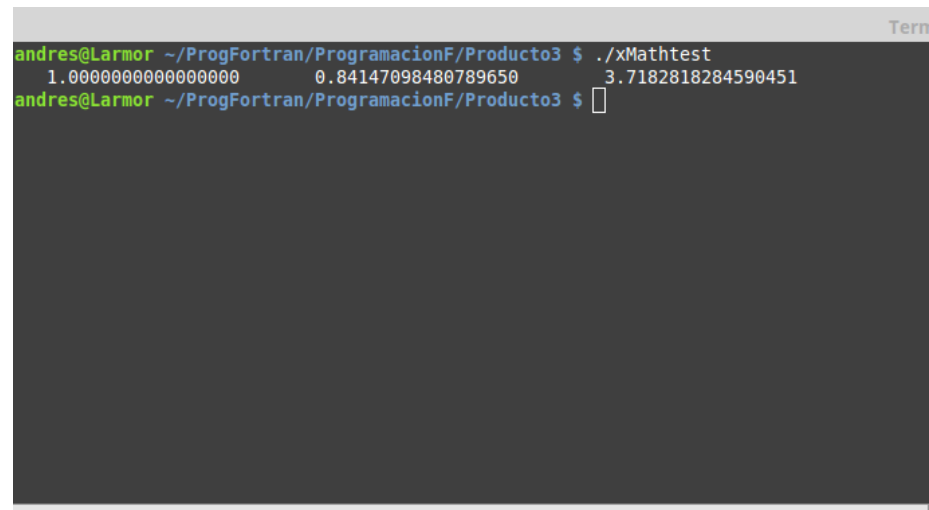
do i = 1, n, 1
    epsilon_m = epsilon_m / 2.0
    one = 1.0 + epsilon_m
    print *, i, one, epsilon_m
end do

End Program Limits

```

Mathtest

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



```

andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ ./xMathtest
1.0000000000000000    0.84147098480789650    3.7182818284590451
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $

```

```

Program Mathtest

Real *8 :: x=1.0 , y, z
y=sin(x)
z=exp(x) + 1.0

Print * , x, y, z

```

```
End Program Mathtest
```

Mathtest2

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



```
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ ./xMathtest2
( 0.00000000 , 1.00000000 ) ( 1.57079637 , 1.31695795 )
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $

-Infinity
```

```
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)

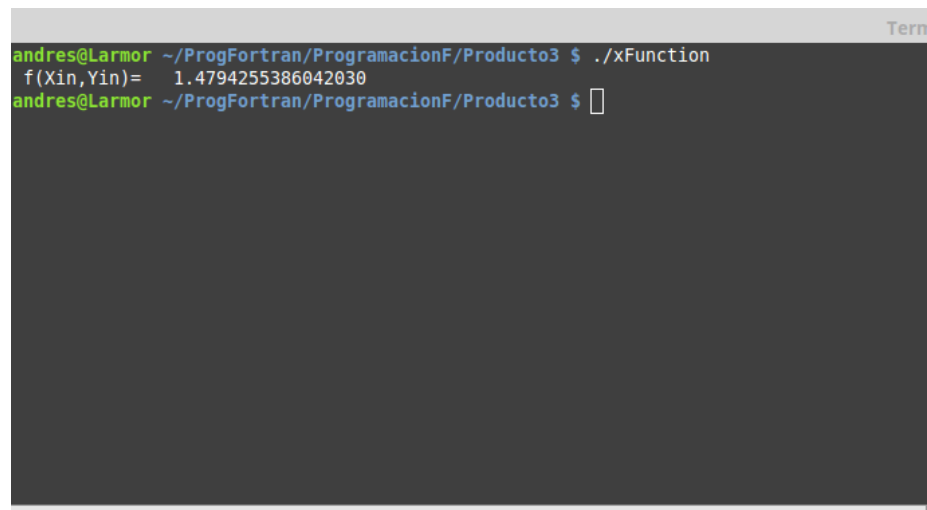
Print * , Z, y, w

End Program Mathtest2

```

Function

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



```

andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ ./xFunction
f(Xin,Yin)= 1.4794255386042030
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ 

```

```

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
```

Subroutine

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.

```
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $ ./xSubroutine
The answers are:  1.4794255386042030    2.1886999242743364
andres@Larmor ~/ProgFortran/ProgramacionF/Producto3 $
```

```
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
```