

Descripción de las prácticas

Rios Quijada Danira

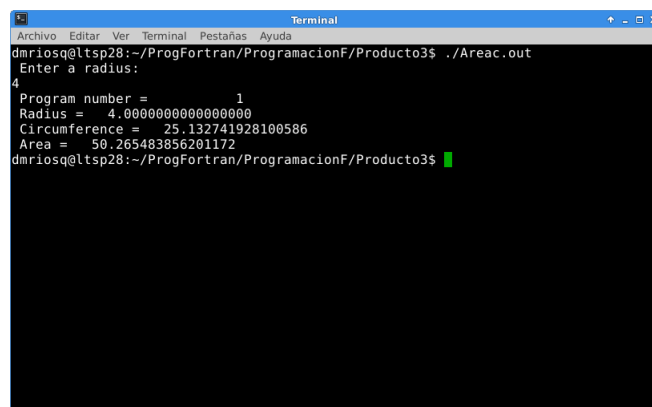
20 de Febrero de 2015

1. Área

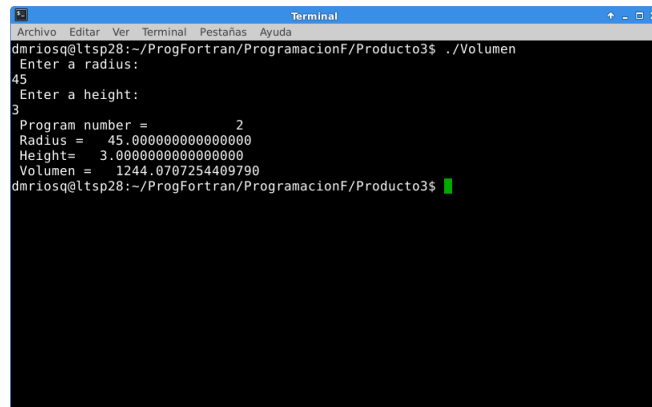
En esta práctica realizamos un programa que calcula el área de un círculo, introduciéndole el radio de dicho círculo.

1.1. Código

```
Program Circle_area
  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 = 2.0 * PI * radius
  area = radius * radius * PI
  print * , 'Program number =', , model_n
  print * , 'Radius =' , radius
  print * , 'Circumference =' , circum
  print * , 'Area =' , area
End Program Circle_area
```



```
Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$ ./Area.out
Enter a radius:
4
Program number = 1
Radius = 4.0000000000000000
Circumference = 25.132741928100586
Area = 50.265483856201172
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$
```



```
Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$ ./Volumen
Enter a radius:
45
Enter a height:
3
Program number = 2
Radius = 45.000000000000000
Height = 3.000000000000000
Volumen = 1244.0707254409790
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$
```

2. Volumen

En esta práctica realizamos un programa que calcula el volumen de un líquido en un recipiente esférico, dependiendo del radio del recipiente y de la altura a la que se encuentre dicho líquido.

2.1. Código

```
Program Volumen_altura
  Implicit None
  Real *8 :: radio , vol , altura
  Real *8 :: PI = 4.0 * atan(1.0)
  Integer :: model_n = 2
  print *, 'Enter a radius:'
  read *, radio
  print *, 'Enter a height:'
  read *, altura
  vol = (PI*(altura*altura))*(radio-(altura/3))
  print *, 'Program number =', model_n
  print *, 'Radius =', radio
  print *, 'Height=', altura
  print *, 'Volumen =', vol
End Program Volumen_altura
```

3. Limites

En esta práctica realizamos un programa que determina la precisión de la maquina en la que se esta ejecutando el programa.

```

dmriosq@lts28:~/ProgFortran/ProgramacionF/Producto3$ ./Limites
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

```

3.1. Código

```

! 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 DOLOOP, 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 doloop
    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

```

4. Math

En esta práctica realizamos un programa que calcula la raíz cuadrada real de -1, la arctan de 2 y el log de 0, los cuales son valores indefinidos o inexistentes en los reales.

4.1. Código

```

! Math . f90 : demo some Fortran math functions
!
Program Math_test ! Begin main program
  Real *8 :: i , p , x = -1.0 , y = 0 , z = 2.0 , w ! Declare variables x, y, z
  i = SQRT (x) ! Call the sine function
  p = LOG (y) ! Call the exponential function
  w= ASIN (z)
  print * , i , p , w ! Print x, y, z
End Program Math_test ! End main program

```

```
Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$ ./Math
NaN
-Infinity
NaN
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$
```

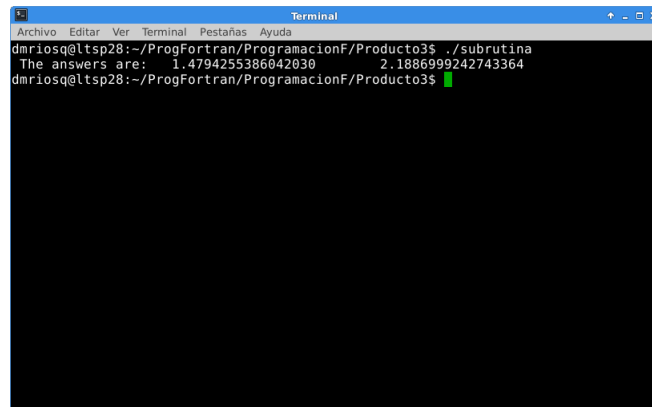
```
Terminal
Archivo Editar Ver Terminal Pestañas Ayuda
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$ ./Function.out
f(Xin, Yin) = 1.4794255386042030
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$
```

5. Funciones

En esta práctica realizamos un programa que calcula el valor de una función de 2 variables.

5.1. Código

```
! 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
```



```
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$ ./subrutina
The answers are: 1.4794255386042030 2.1886999242743364
dmriosq@ltsp28:~/ProgFortran/ProgramacionF/Producto3$
```

6. Subrutina

En esta práctica realizamos una subrutina, el cual es un subproceso, que forma parte de un proceso principal.

6.1. Código

```
! 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
  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
```