PRIMERA PRACTICA FORTRAN

Ana Magdalena Sotomayor

18 de febrero de 2015

1. INTRODUCCION

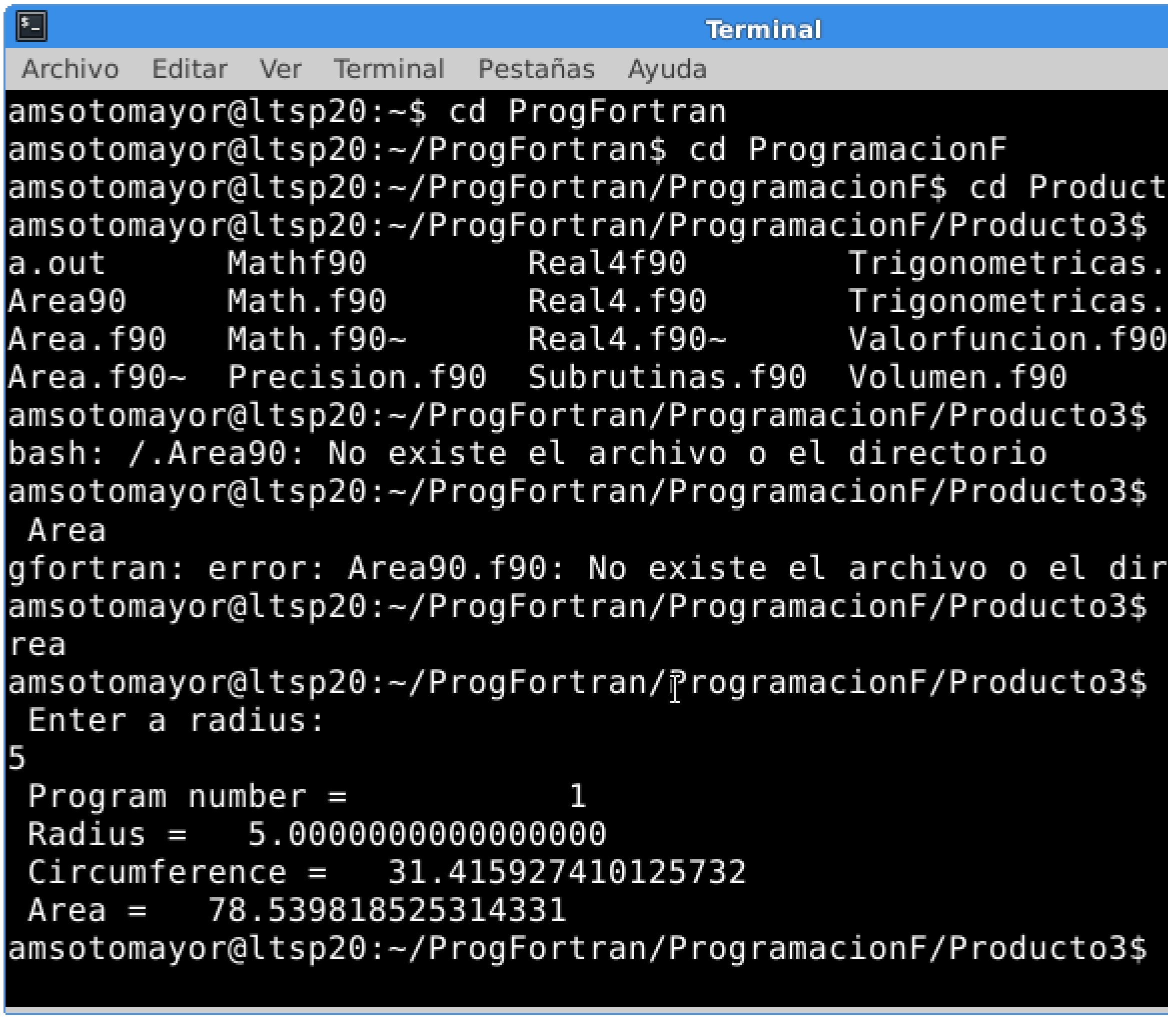
Se codificaron 8 nuevos programas tales que realizan operaciones para obtener el area de un circulo, el volumen de un liquido dentro de una esfera, funciones trigonometricas, Comprobacion de la precision numerica del ordenador en 8 bits y 4 bits y los valores de diferentes funciones matematicas.

2. CODIGOS

2.1. calcular el area de un circulo

```
! Area . f90 : Calculates the area of a circle, sample program
 Program areacirculo ! 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
  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 areacirculo ! End main program code
```

2.2. Imagen de salida



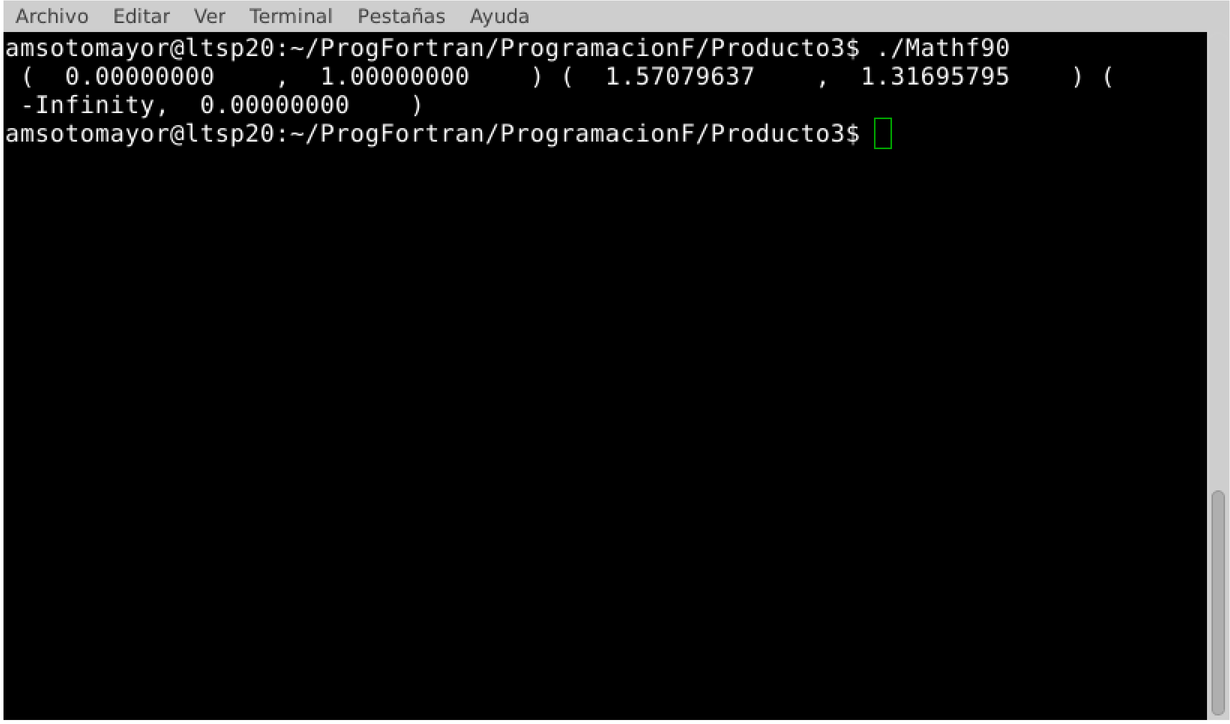
```
amsotomayor@ltsp20:~$ cd ProgFortran
amsotomayor@ltsp20:~/ProgFortran$ cd ProgramacionF
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF$ cd Product
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$
a.out      Mathf90        Real4f90        Trigonometricas.
Area90     Math.f90       Real4.f90       Trigonometricas.
Area.f90   Math.f90~      Real4.f90~      Valorfuncion.f90
Area.f90~  Precision.f90  Subrutinas.f90  Volumen.f90
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$
bash: /.Area90: No existe el archivo o el directorio
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$
 Area
gfortran: error: Area90.f90: No existe el archivo o el dir
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$
rea
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$
 Enter a radius:
5
 Program number =           1
 Radius =   5.0000000000000000
 Circumference =   31.415927410125732
 Area =   78.539818525314331
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$
```

2.3. Calculos Matematicos

```
! Math . f90 : demo some Fortran math functions
 Program Math2! Begin main program

  Complex *8 :: x=- 1.0 , y=2, z=0 ! Declare variables x, y, z
 x = sqrt (x)
 y = asin (y) ! Call the sine function
 z = log (z) ! Call the exponential function
 print * , x, y, z ! Print x, y, z
End Program Math2 ! End main program
```

2.4. Imagen de salida



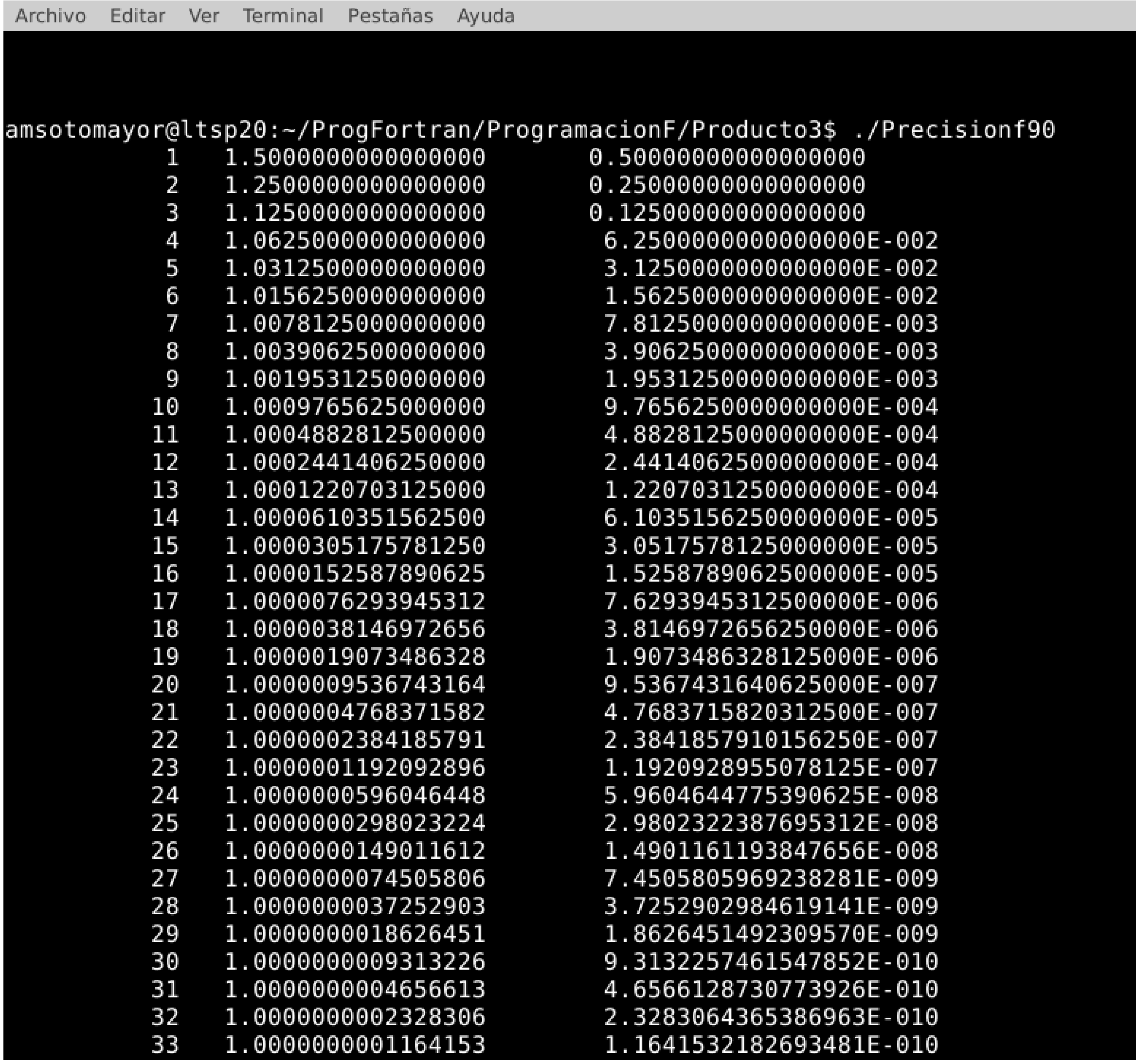
```
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ ./Mathf90
 (  0.00000000    ,  1.00000000    ) (  1.57079637    ,  1.31695795    ) (
 -Infinity,  0.00000000    )
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ 
```

2.5. Precision del Ordenador con numeros Reales en 8 bits

```
! Limits . f90 : Determines machine 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 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
```

2.6. Imagen de salida



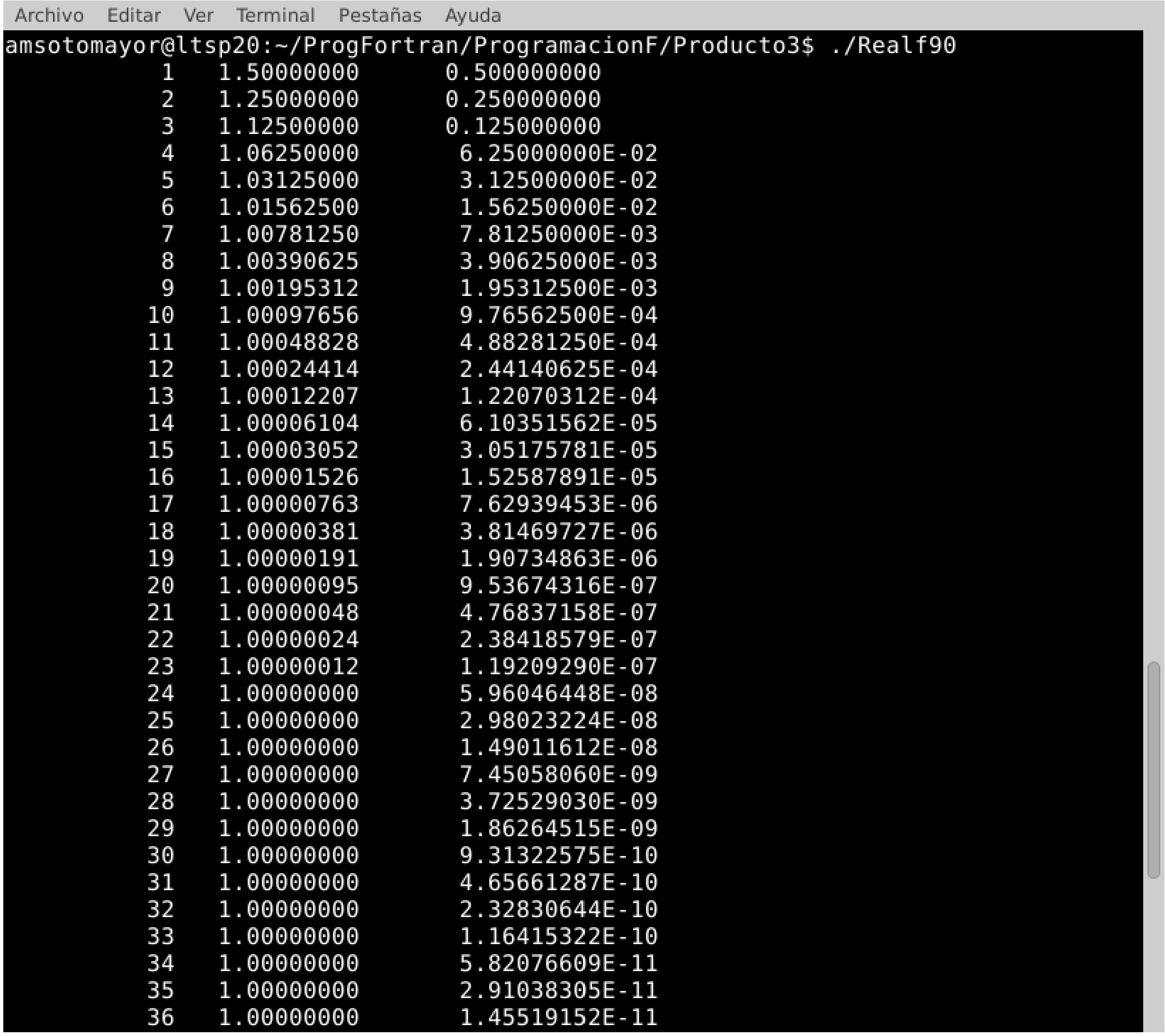
```
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ ./Precisionf90
           1   1.5000000000000000       0.50000000000000000
           2   1.2500000000000000       0.25000000000000000
           3   1.1250000000000000       0.12500000000000000
           4   1.0625000000000000        6.2500000000000000E-002
           5   1.0312500000000000        3.1250000000000000E-002
           6   1.0156250000000000        1.5625000000000000E-002
           7   1.0078125000000000        7.8125000000000000E-003
           8   1.0039062500000000        3.9062500000000000E-003
           9   1.0019531250000000        1.9531250000000000E-003
          10   1.0009765625000000        9.7656250000000000E-004
          11   1.0004882812500000        4.8828125000000000E-004
          12   1.0002441406250000        2.4414062500000000E-004
          13   1.0001220703125000        1.2207031250000000E-004
          14   1.0000610351562500        6.1035156250000000E-005
          15   1.0000305175781250        3.0517578125000000E-005
          16   1.0000152587890625        1.5258789062500000E-005
          17   1.0000076293945312        7.6293945312500000E-006
          18   1.0000038146972656        3.8146972656250000E-006
          19   1.0000019073486328        1.9073486328125000E-006
          20   1.0000009536743164        9.5367431640625000E-007
          21   1.0000004768371582        4.7683715820312500E-007
          22   1.0000002384185791        2.3841857910156250E-007
          23   1.0000001192092896        1.1920928955078125E-007
          24   1.0000000596046448        5.9604644775390625E-008
          25   1.0000000298023224        2.9802322387695312E-008
          26   1.0000000149011612        1.4901161193847656E-008
          27   1.0000000074505806        7.4505805969238281E-009
          28   1.0000000037252903        3.7252902984619141E-009
          29   1.0000000018626451        1.8626451492309570E-009
          30   1.0000000009313226        9.3132257461547852E-010
          31   1.0000000004656613        4.6566128730773926E-010
          32   1.0000000002328306        2.3283064365386963E-010
          33   1.0000000001164153        1.1641532182693481E-010
```

2.7. Programacion de Precision del ordenador en 4 bits

```
! Limits . f90 : Determines machine precision
! 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 docalculate one
   print * , i , one , epsilon_m ! Print values so far
 end do ! End loop when i>n
End Program Real4
```

2.8. Imagen de salida

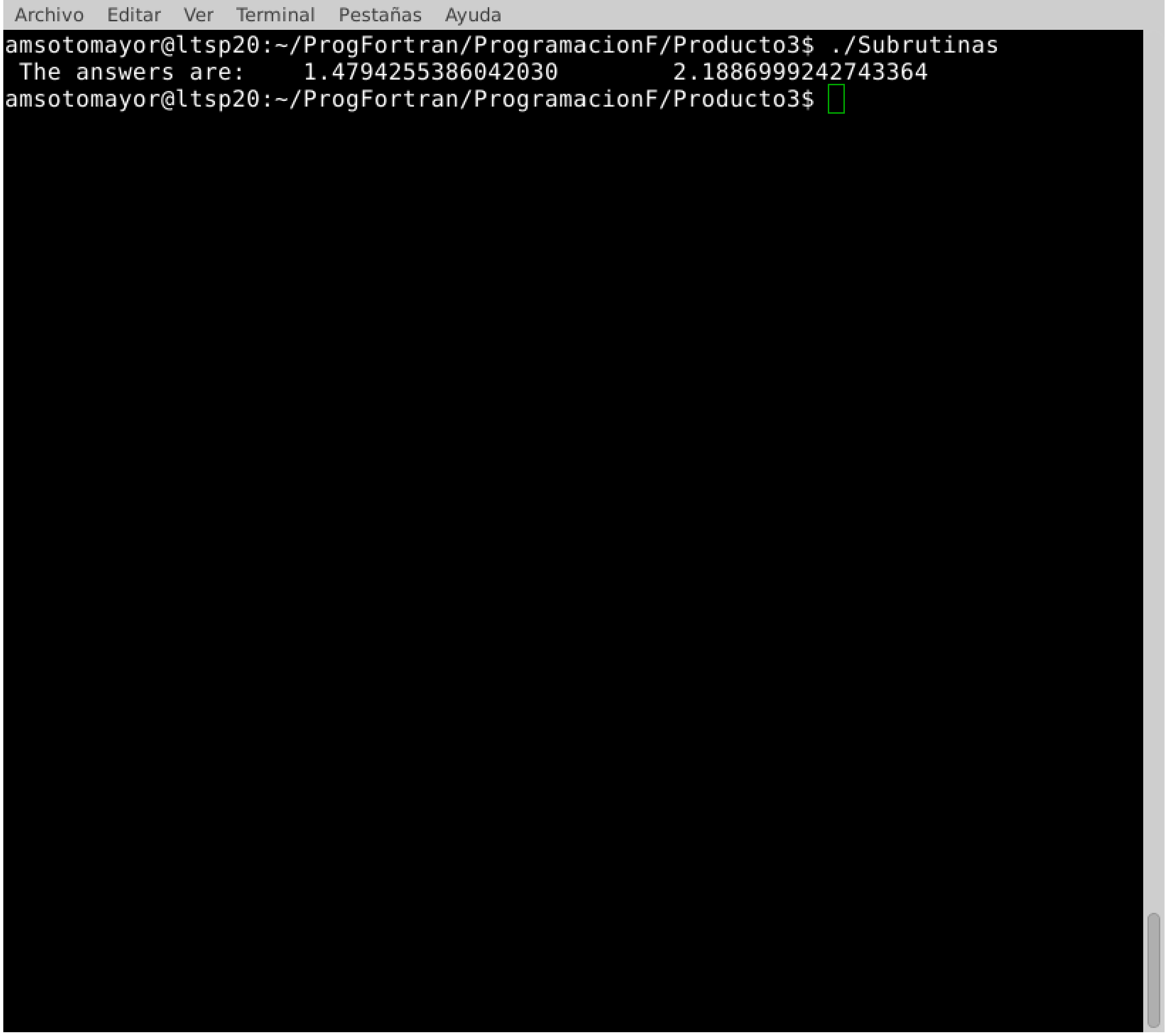


```
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ ./Realf90
           1   1.50000000      0.500000000
           2   1.25000000      0.250000000
           3   1.12500000      0.125000000
           4   1.06250000       6.25000000E-02
           5   1.03125000       3.12500000E-02
           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
```

2.9. Subrutinas

```
! 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 Mainprogram ! 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 Mainprogram
```

2.10. Imagen de salida



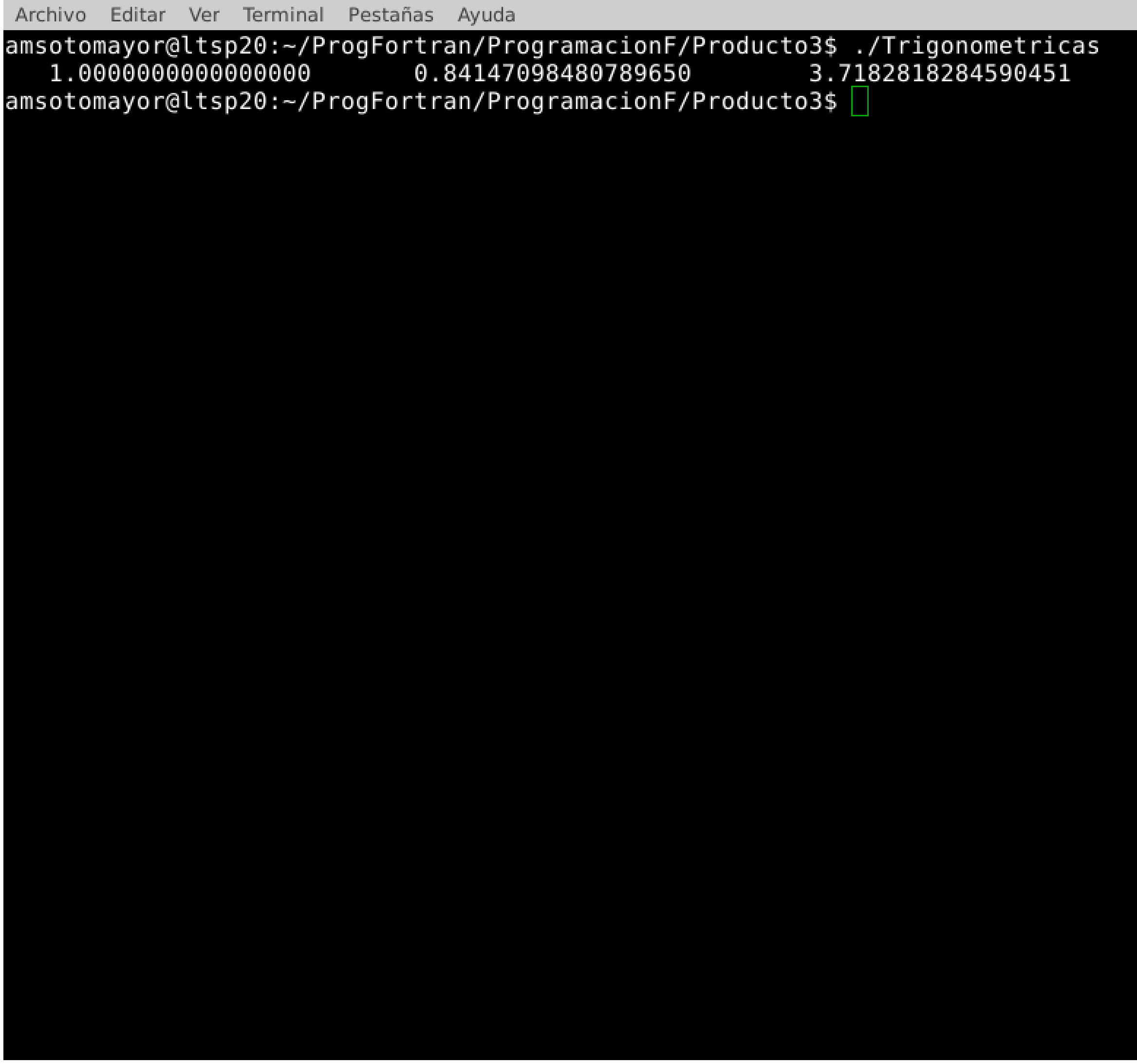
```
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ ./Subrutinas
 The answers are:    1.4794255386042030        2.1886999242743364
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ 
```

2.11. Funciones Trigonometricas

```
! Math . f90 : demo some Fortran math functions
!
Program Mathtest! 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 Mathtest ! End main program
```

2.12. Imagen de salida



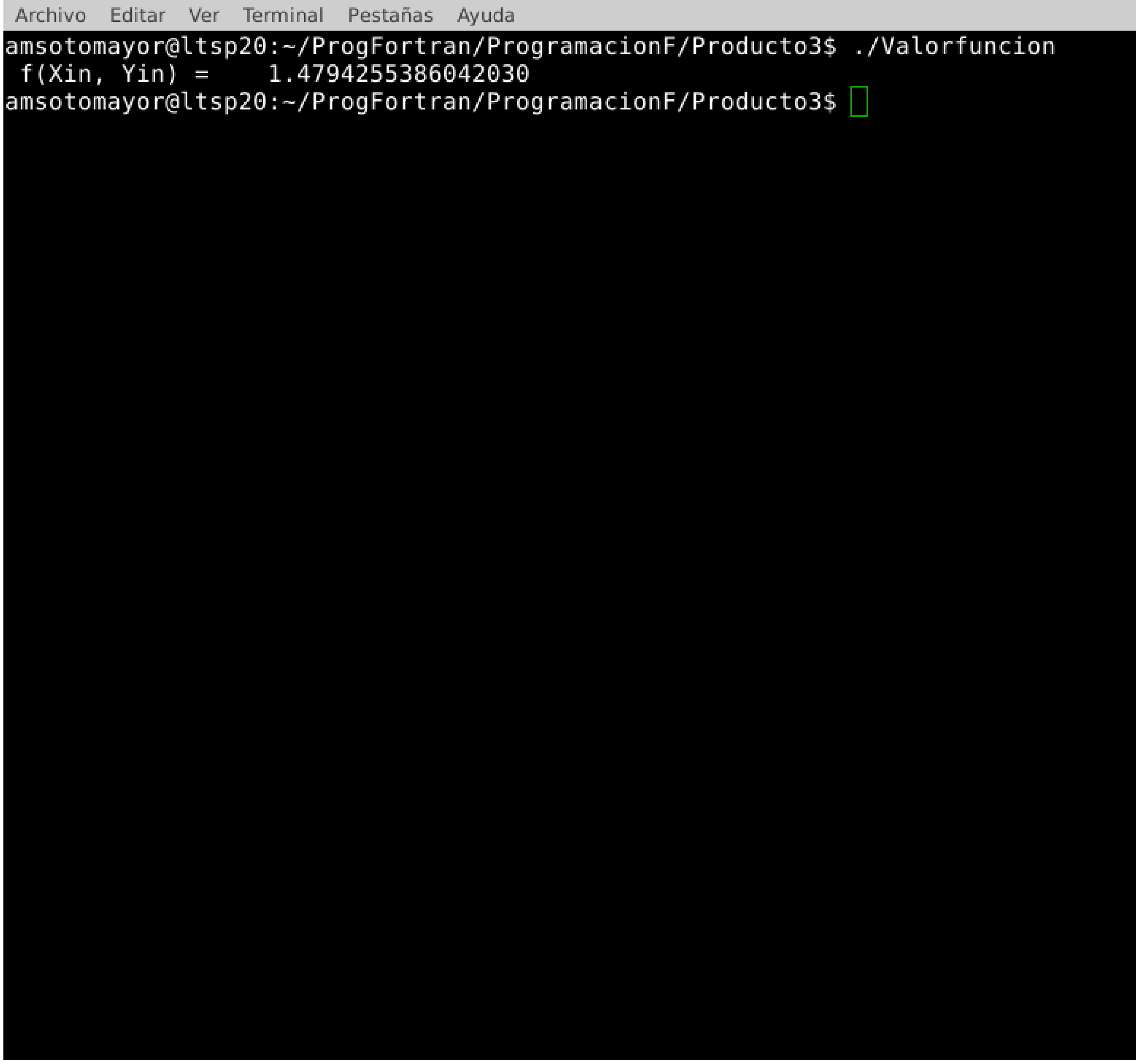
```
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ ./Trigonometricas
   1.0000000000000000       0.84147098480789650        3.7182818284590451
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ 
```

2.13. El Valor de una funcion dada

```
 ! 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
```

2.14. Imagen de salida

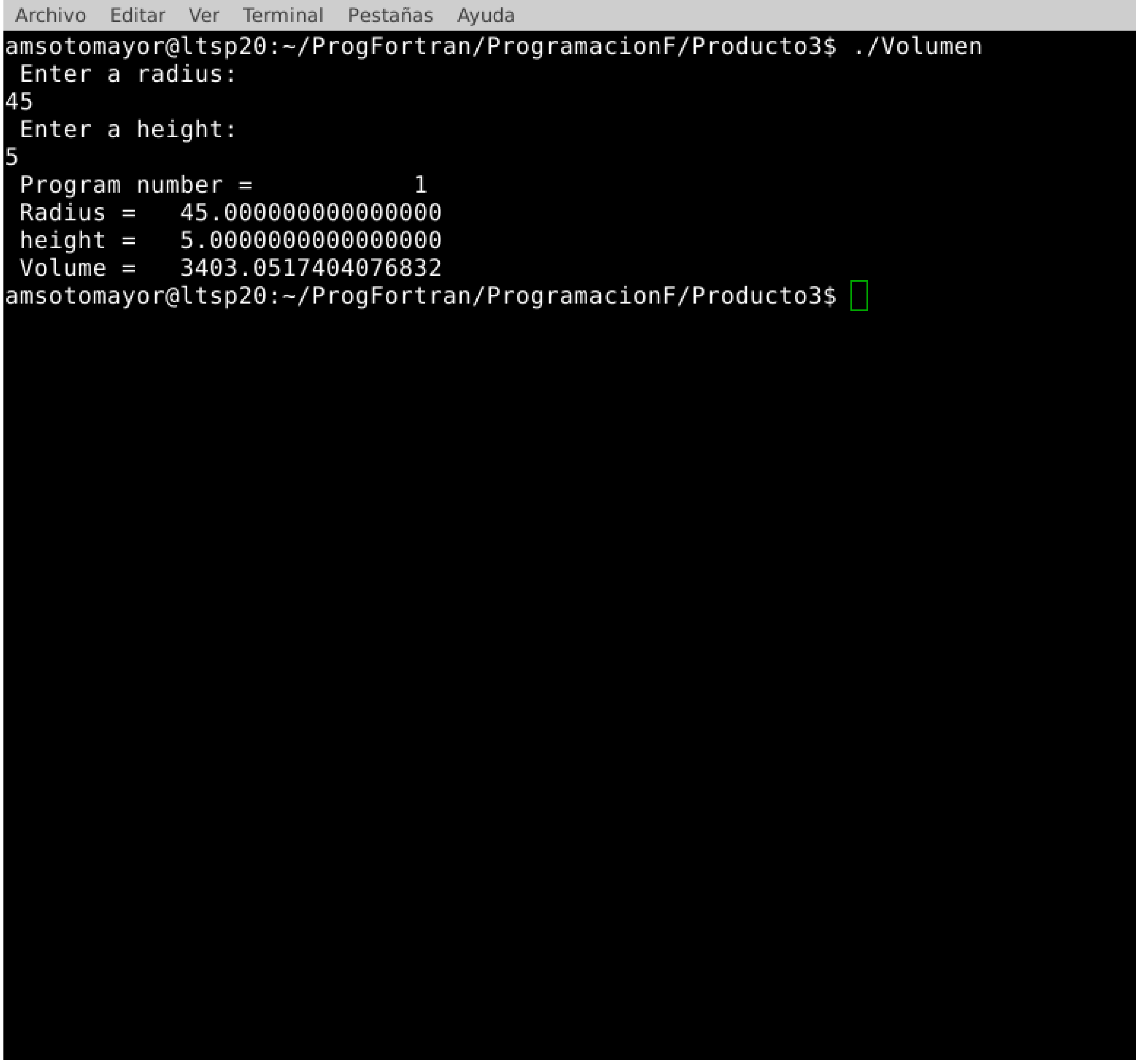


```
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ ./Valorfuncion
 f(Xin, Yin) =    1.4794255386042030
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ 
```

2.15. Volumen de un liquido en una esfera

```
! Area . f90 : Calcular el volumen de liquido en un tanque esferico
 ! -----------------------------------------------
 Program Sphere_volume ! Begin main program
  Implicit None ! Declare all variables
   Real *8 :: radius , height , volume , Newradius ! 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 ! Tomar el valor de la h
  Newradius =  3 * radius - height  ! Calc volume
  volume = 0.3333 * PI * height * height * Newradius
print * , ’Program number =’ , model_n ! Print program number
 print * , ’Radius =’ , radius ! Print radius
 print * , ’height =’ , height ! Print height
 print * , ’Volume =’ , volume ! Print circumference
End Program Sphere_volume ! End main program code
```

2.16. Imagen de salida



```
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ ./Volumen
 Enter a radius:
45
 Enter a height:
5
 Program number =           1
 Radius =   45.000000000000000
 height =   5.0000000000000000
 Volume =   3403.0517404076832
amsotomayor@ltsp20:~/ProgFortran/ProgramacionF/Producto3$ 
```