

Fortran- Funciones

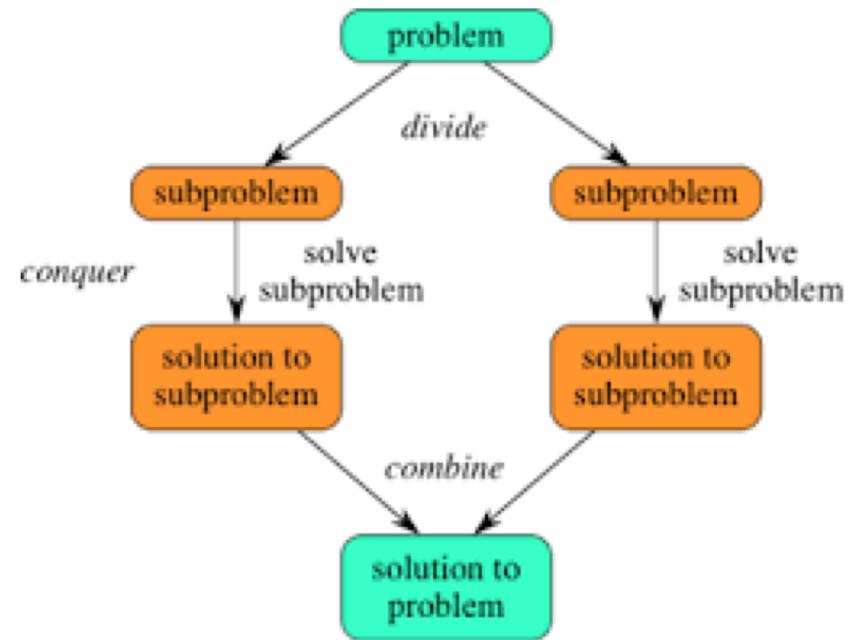
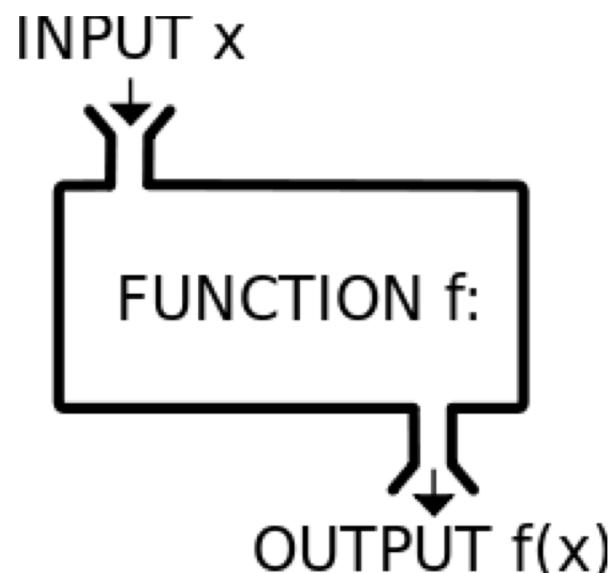
TEORÍA

¿Dónde estamos en el ciclo de vida del software?



Funciones

“Divide y vencerás”



Funciones y subrutinas

Fotran permite tener dentro de un archivo fuente varias funciones, esto con el fin de dividir las tareas y que sea más fácil la depuración, la mejora y el entendimiento del código.

FORTRAN provee dos maneras de implementar subprogramas:

- creando **funciones** o
- creando **subrutinas**.

Estos subprogramas pueden ser de dos tipos:

- ❖ **intrínsecos**
- ❖ **externos**
 - ❖ **De una sola línea**
 - ❖ **De varias líneas**

Subprogramas intrínsecos

Los subprogramas intrínsecos son aquellos que provee el compilador, algunos de los más comunes son:

abs	valor absoluto
min	valor mínimo
max	valor máximo
sqrt	raíz cuadrada
sin	seno
cos	coseno
tan	tangente
atan	arco tangente
exp	exponente (natural)
log	logaritmo (natural)

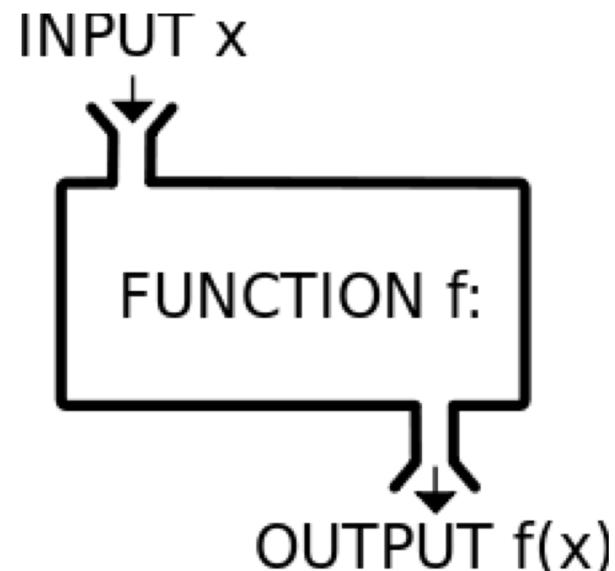
Ejemplo subprogramas intrínsecos

```
1      program intrinsecos
2          real a
3          write(*,*) "Ingresa el valor a calcular:"
4          read(*,*) a
5          write(*,10) "La raiz cuadrada del valor: ", sqrt(a)
6          write(*,10) "El logaritmo natural: ", log(a)
7          10 format (a,f10.2)
8      end
```

Funciones de una sentencia

Son funciones que se pueden expresar con una sola fórmula o expresión. En una sola sentencia.

$$f(x, y, z, \dots) = \text{formula}$$



Subprogramas externos

Por otro lado, los subprogramas externos son aquellos escritos por el usuario (o bien pueden formar parte de una biblioteca desarrollados por terceros).

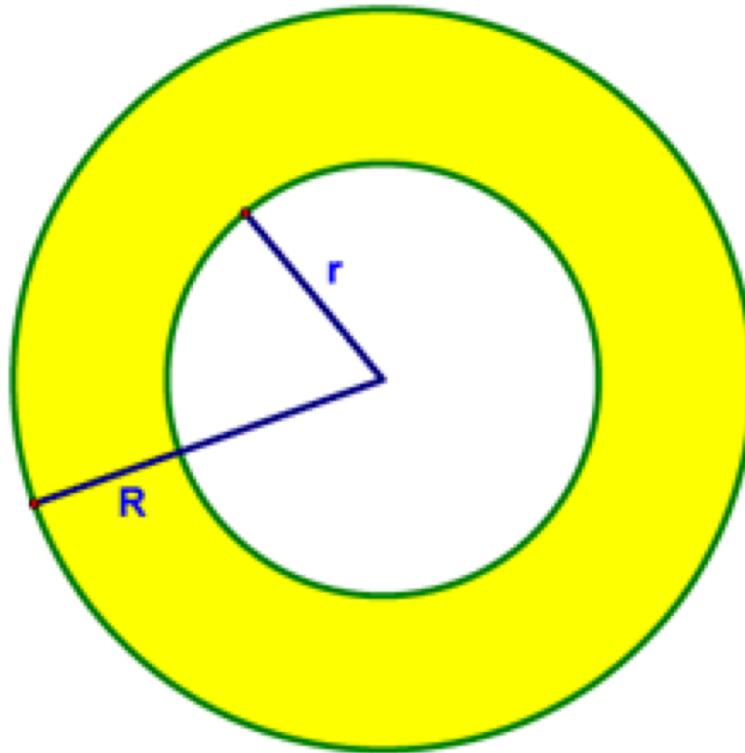


Pueden ser:

- De una sola línea de código
- De varias líneas de código

Ejemplo: Funciones de una sentencia

Problema: Calcular el área de la argolla amarilla



$$\begin{aligned} A_{\text{Annulus}} &= A_{\text{Big Circle}} - A_{\text{Small Circle}} \\ &= \pi R^2 - \pi r^2 \end{aligned}$$

```
1 program annulus  
2 double precision r, area, pi, a, b  
3 parameter (pi = 3.1415926535897932D0)  
4 area(r) = pi * r * r  
5 print *, "Enter the inner and outer radii of  
6 read *, a, b  
7 write (*,10) "The area of the annulus is ",  
8 format (a,f25.15)  
9  
10 end
```

?

Subprogramas externos de varias líneas

Por otro lado cuando el subprograma es más complejo y no puede escribirse en una sola línea de código, utilizaremos funciones completas.

Función (sintaxis)



```
valorRetorno nombre ( parámetros )  
! bloque de código de la función  
END
```

Funciones y subrutinas

Tres elementos (prototipo de función):

1. El nombre de la función: El identificador con el que llamaremos a la función
2. Los parámetros que recibe la función: Los datos de entrada a la función. (opcionales)
3. El valor de retorno de la función: El tipo de dato que regresa la función

valorRetorno nombre (*parámetros*)
! bloque de código de la función

END

Ejemplo: Funciones de varias sentencias

Problema: Calcular el factorial de un número

n	Fórmula	in!
0	Ninguno	1
1	$1! = 1$	1
2	$2! = 1 \cdot 2$	2
3	$3! = 1 \cdot 2 \cdot 3$	6
4	$4! = 1 \cdot 2 \cdot 3 \cdot 4$	24
5	$5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5$	120
6	$6! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6$	720
7	$7! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 \cdot 7$	5040
8	$8! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 \cdot 7 \cdot 8$	40.320

```
1      program demofactorial
2          integer fact, n
3          write(*,*) "Ingresa el valor de n?:"
4          read(*,*) n
5          write(*,*) "El valor de ", n, " factorial es: ", fact(n)
6          end
7          function fact(n)
8              integer fact, n, p
9              p = 1
10             do i = 1, n
11                 p = p * i
12             end do
13             fact = p
14             return
15         end
```

Ejemplo: Funciones de varias sentencias

Problema: Hacer un programa que la **distribución de Poisson** que es una [distribución de probabilidad discreta](#) que expresa, a partir de una frecuencia de ocurrencia media, la probabilidad de que ocurra un determinado número de eventos durante cierto período de tiempo. Concretamente, se especializa en la probabilidad de ocurrencia de sucesos con probabilidades muy pequeñas, o sucesos "raros".

La función de masa o probabilidad de la distribución de Poisson es

$$f(k, \lambda) = \frac{e^{-\lambda} \lambda^k}{k!}$$

donde

- k es el número de ocurrencias del evento o fenómeno (la función nos da la probabilidad de que el evento suceda precisamente k veces).
- λ es un parámetro positivo que representa el número de veces que se espera que ocurra el fenómeno durante un intervalo dado. Por ejemplo, si el suceso estudiado tiene lugar en promedio 4 veces por minuto y estamos interesados en la probabilidad de que ocurra k veces dentro de un intervalo de 10 minutos, usaremos un modelo de distribución de Poisson con $\lambda = 10 \times 4 = 40$.
- e es la base de los logaritmos naturales ($e = 2,71828\dots$)

EJEMPLO:

A man was able to complete 3 files a day on an average. Find the probability that he can complete 5 files the next day.

Solution: Here we know this is a Poisson experiment with following values given:

$\mu = 3$, average number of files completed a day

$x = 5$, the number of files required to be completed next day

On substituting the values in the Poisson distribution formula mentioned above we get the Poisson probability in this case.

We get,

$$P(x, \mu) = \frac{(e^{-\mu})(\mu^x)}{x!}$$

$$\rightarrow P(5, 3) = \frac{(2.71828)^{-3}(3^5)}{5!}$$

= 0.1008 approximately.

Hence the probability for the person to complete 5 files the next day is 0.1008 approximately.

```
1      program poisson_program
2          real poisson, s
3          integer m
4          write(*,*) "Ingresa el valor de m?:"
5          read(*,*) m
6          write(*,*) "Ingresa el valor de s?:"
7          read(*,*) s
8          write(*,*) "La probabilidad de poisson es: ", poisson(m,s)
9      end
10     function fact(n)
11         integer fact, n, p
12         p = 1
13         do i = 1, n
14             p = p * i
15         end do
16         fact = p
17     end
18     function poisson(n,t)
19         real poisson, t
20         integer n, fact
21         poisson = (t ** n) * exp(-t) / fact(n)
22     end
```

Ejemplo: Funciones de varias sentencias
Con arreglos

Problema: Hacer un programa calcule el promedio
de calificaciones de un alumno

$$\bar{x} = \frac{\sum x}{n}$$

```
1      program mean
2      real numbers(100), avg
3      integer m
4      write(*,*) "How many numbers are on your list?"
5      write(*,*) "(no more than 100, please)"
6      read (*,*) m
7      do i =1, m
8          write(*,*) "Enter your next number:"
9          read (*,*) numbers(i)
10     end do
11     write(*,*) "The average is", avg(m,numbers)
12     end
13     function avg(n,list)
14         real avg, list(100), sum
15         integer n
16         sum = 0
17         do i = 1, n
18             sum = sum + list(i)
19         end do
20         avg = sum/n
21     end
```

Ejemplo: SUBRRUTINAS

Problema: Hacer un programa calcule el promedio de calificaciones de un alumno CON SUBRRUTINAS

$$\bar{x} = \frac{\sum x}{n}$$

?

```
1 program average
2 real x, y, z
3 print *, "What are the two numbers you want to
4 read *, x, y
5 call avg(x,y,z)
6 print *, "The average is", z
7 end
8
9 subroutine avg(a,b,c)
10 real a, b, c
11 c = (a + b)/2.
12 end
```

Ejemplo: SUBRRUTINAS

Problema: Hacer multiplicación de dos matrices de 2X2 y 2X1 CON SUBRRUTINAS

$$Ax = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{bmatrix}$$

```
1 subroutine prod(A,x,y)
2 real A(2,2), x(2), y(2)
3 y(1) = A(1,1) * x(1) + A(1,2) * x(2)
4 y(2) = A(2,1) * x(1) + A(2,2) * x(2)
5 end
```

Ejemplo: SUBRRUTINAS

Problema: Sin recibir parámetros, sin devolver valores

```
1 | subroutine bluesky
2 | print *, "The sky is blue."
3 | end
```

The call statement for this subroutine,

call bluesky

EVALUACIÓN PRACTICA 12.

FUNCIONES Y SUBRRUTINAS

Problema 1:

Hacer la sumatoria de los n números naturales, donde n es el número de términos a sumar. La sumatoria de los n términos esta dada por la suma de ese número mas cada uno de los números anteriores hasta llegar a 1

Ejem: n=10

Resultado= $10+9+8+7+6+5+4+3+2+1$

Restricciones: n diferente de cero

Hacer multiplicación de matrices del examen CON FUNCIONES

EVALUACIÓN PRACTICA 12. FUNCIONES Y SUBRRUTINAS

Problema 2:

Programa que calcule la multiplicación de dos matrices como se muestra en el ejemplo. El usuario ingresa los valores de las matrices. Los resultados deben mostrar las matrices ingresadas y la multiplicación. Números enteros

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}_{2 \times 2}; B = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \end{pmatrix}_{2 \times 3}$$
$$A \cdot B = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}_{2 \times 2} \cdot \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \end{pmatrix}_{2 \times 3} = \begin{pmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \end{pmatrix}_{2 \times 3}$$
$$A \cdot B = \begin{pmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \end{pmatrix}_{2 \times 3} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} & a_{11}b_{13} + a_{12}b_{23} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} & a_{21}b_{13} + a_{22}b_{23} \end{pmatrix}_{2 \times 3}$$

Hacer multiplicación de matrices del examen CON SUBRRUTINAS

Referencias:

<http://math.hawaii.edu/wordpress/fortran-4/>

<http://math.hawaii.edu/wordpress/fortran-5/#func>

<http://math.hawaii.edu/wordpress/fortran-6/#call>

Tipos de datos

Integer

Real

Double precision Complex

Logical Character

Lógicas

.LT. <

.LE. <=

.GT. >

.GE. >=

.EQ. =

.NE. /=

parameter (nombre = constante, ... , nombre = constante)

** {exponenciación} * /
{multiplicación, división} + -
{suma, resta}

if (*expresión lógica*) **then**
sentencias
endif

select case (*selector*)
 case (*valor*)
 sentencias
 case default
 sentencias
end select

do while (*expr lógica*)
sentencias
end do

do *var* = *inicio*, *fin*, *incremento*
sentencias
end do

do (*expr lógica*)
 expr3 *sentencias*
end do

<http://www.famaf.unc.edu.ar/~vmarconi/numerico1/FortranTutorial.pdf>