Slide 1

TEMA 9: SUBPROGRAMAS: PROCEDIMIENTOS

Índice

Slide 2

Ejemplo: Paso de polares a cartesianas en el plano

$$\begin{array}{rccc} f: & [0,\infty)\times[0,2\pi) & \rightarrow & R\times R \\ & (r,z) & & (rcos(z), rsin(z)) \end{array}$$

Primera solución : usar 2 funciones :

$$\begin{array}{rccc} coordx: & [0,\infty)\times[0,2\pi) & \rightarrow & R \\ & (r,z) & & rcos(z) \end{array}$$

$$\begin{array}{rccc} coordy: & [0,\infty)\times[0,2\pi) & \rightarrow & R \\ & (r,z) & & rsin(z) \end{array}$$

Slide 3

```
function coordx (r, z: real): real;
   (* Pre: r >= 0  0<=z<2*PI *)
   begin
       coordx := r * cos (z)
   end ;

function coordy (r, z: real): real;
   (* Pre: r >= 0  0<=z<2*PI *)
   begin
       coordy := r * sin (z)
   end ;
```

Slide 4

(* $r = r_0$, $z = z_0$ x =?, y =? *)

```
x := coordx (r, z); y := coordy (r,z);
```

(* $r = r_0$, $z = z_0$, x, y cartesianas de r, z *)

Módulo PolACart

Entrada: Dos reales r y z

Salida : Coordenadas cartesianas, x, y

Pre: $0 \leq r$, $0 \leq z \leq 2\pi$

(* $r = r_0$, $z = z_0$ x =?, y =? *)

```
PolACart (r, z, x, y) ;
```

(* $r = r_0$, $z = z_0$, x, y cartesianas de r, z *)

Slide 5

```
program p (input, output);
const PI=3.141592;

  procedure PolACart (r, z : real; var x, y : real);
  (* Entrada : parámetros r, z (radio y ángulo)
     Salida: coordenadas cartesianas x, y
     Pre: r >= 0 0<=z<2*PI *)
  begin
       x := r * cos(z); y := r * sin(z)
  end;

var radio, angulo, coordx, coordy : real;
BEGIN ...
   (* radio y angulo asignados, cumpliendo precondición *);
   PolACart (radio, angulo, coordx, coordy); ...
   PolACart (2*radio, angulo/2, coordx, coordy); ...
END.
```

Slide 6

PROCEDIMIENTOS

Definición (Una. Antes del código del algoritmo que lo usará)

procedure <*id-procedimiento*> (<*lista-parámetros*>) **;**
<*secciones de constantes, variables etc locales* >
begin
<*sentencias; asignación a parámetros de salida* >
end ;

<*lista-parámetros*> ::= <*bloque-par*> { <*bloque-par*> }
<*bloque-par*> ::= **var** <*id-par*> { , <*id-par*> } : <*id-tipo*>
| <*id-par*> { , <*id-par*> } : <*id-tipo*>

Uso: sentencia (varios usos)

<*id-procedimiento*> (<*lista de expresiones argumentos*>)

parámetros formales $\overset{\text{posición}}{\longleftrightarrow}$ parámetros actuales (argumentos)

Slide 7

PROCEDIMIENTOS

- *Definición*
  - *Declaración*
    - **nombre**: identificador
    - **tipo**: de parámetros
  - *Asignación* de **valor**: algoritmo de cálculo
- *Uso*: es una acción que modifica el entorno

Entrada: los argumentos (en los parámetros)
Salida: los argumentos de salida modificados (parámetros **var**)
Precondición: sobre los argumentos de entrada

Uso: es una acción que modifica el entorno especificado en los parámetros de salida, a partir de los de entrada.

Algoritmo de cálculo: usa los parámetros y los elementos locales.

Obtención de los resultados : asignación a los identificadores de parámetros de salida.

Slide 8

Ejemplo: Escribir en pantalla el ordinal correspondiente a un número entero:
(* n =n_0 *)
EscOrdinal (m);
(* n =n_0, en la pantalla se ha escrito el ordinal correspondiente *)

Módulo EscOrdinal
Entrada: Un entero n
Salida : -
Otros efectos: escribe en pantalla
Pre: $1 \leq n \leq 9$

Slide 9

Ejemplo: Leer de teclado un número estrictamente positivo
(* n = ? *)
LeerPos (n);
(* n =n_0, valor obtenido del usuario, $n_0 > 0$ *)

Módulo LeerPos
Entrada: -
Salida : entero positivo
Otros efectos: lee de teclado
Post: $n > 0$

Slide 10

Ejemplo: Intercambiar los valores de dos variables enteras
(* m = m_0, n = n_0 *)
SwapI (m, n) ;
(* m = n_0, n = m_0 *)

Módulo SwapI
Entrada-Salida: variables p e q, enteras
(como entrada: los valores,
como salida: las posiciones)
Otros efectos: -
Pre: p=p_0 $\wedge$ q=q_0
Post: p=q_0 $\wedge$ q=p_0

Slide 11

Nombre	**Entradas**	**Salidas**	**Otros**	**Objetivo**
PolACart	r real ≥ 0 $z \in [0, 2 * \pi)$	x, y	-	pasa a cartesianas
EscOrdin	entero $\in [1, 399]$	-	escribe en pantalla	escribe el ordinal
LeePosit	-	entero >0	lee de teclado escribe en p.	entero >0 del usuario
LeeRang	p,q enteros	entero en $[p, q]$	lee de teclado escribe en p.	entero $\in [p, q]$ del usuario
SwapI	p,q enteros	p, q	-	intercambia los valores
BorraP	-	-	escribe en pantalla	borra la pantalla

Slide 12

Módulo `writeln`
Entrada: Un primer parámetro optativo (`output`) (de tipo ...)
Número indeterminado de parámetros de tipos imprimibles
Salida : -
Otros efectos: Escribe en pantalla los valores de los argumentos
y un fin de línea

Módulo `readln`
Entrada : Un primer parámetro optativo (`input`) (de tipo ...)
Salida : Número indeterminado de parámetros de tipos leíbles
Otros efectos: Lee de teclado los valores para los argumentos
y avanza a la siguiente línea

Slide 13

	paso	parám.	argumento	interesa
e	valor	<id>	expresión (no se modifica)	Valor
s	var.	**var** <id>	variable (modificable)	Lugar
e-s	var.	**var** <id>	variable (modificable)	Valor y lugar
result.	vd	-	-	resultado
(	func.	**function**	id. función	código)
(	proc.	**procedure**	id. proc	código)

Slide 14

Paso por valor

parámetro ↔ **argumento**

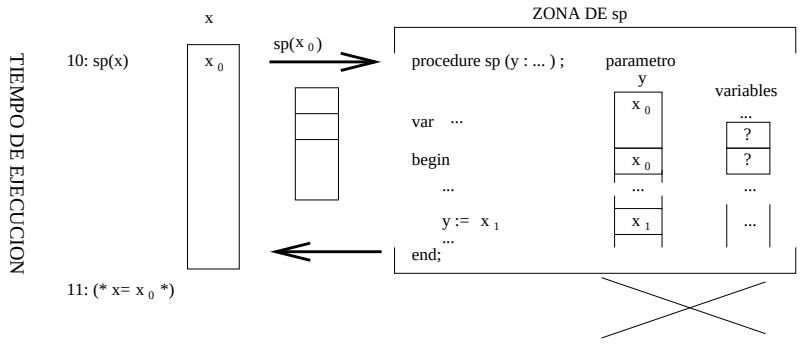
identificador expresión

Elemento de **entrada** al subprograma. El argumento no se modifica.

1. Se evalúa el argumento
2. El valor se asigna al parámetro (ámbito local).
   (El parámetro puede ser modificado)
3. Se ejecuta el código del subprograma, usando y modificando el parámetro
4. Cuando el subprograma termina, el parámetro "desaparece".
   (El argumento no cambia de valor)

Slide 15

Paso por valor



Slide 16

Paso por variable I

parámetro ↔ **argumento**

identificador variable

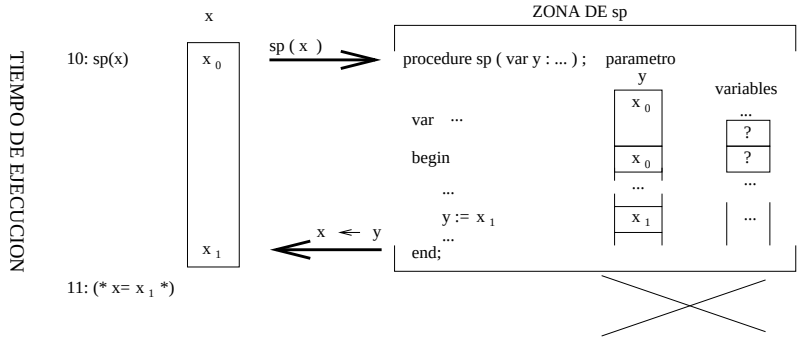
Elemento de **salida** ó **entrada-salida** al subprograma.
Excepcionalmente de entrada. El argumento puede ser modificado.

doble copia

1. Se evalúa el argumento
2. El valor se asigna al parámetro (ámbito local).
   (El parámetro puede ser modificado)
3. Se ejecuta el código del subprograma, usando y modificando el parámetro
4. Cuando el subprograma termina, el valor del parámetro se asigna a la variable argumento, y el parámetro "desaparece".

Slide 17

Paso por variable: doble copia



Slide 18

Paso por variable II

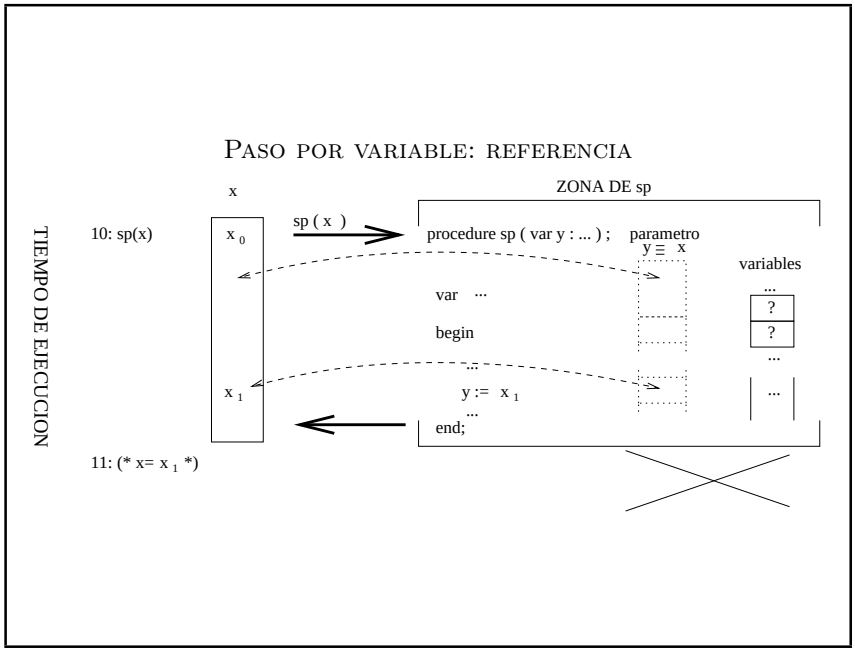
referencia

1. El parámetro se refiere al argumento (ocupa la misma posición: zona de memoria con dos nombres durante la vida del subprograma)
2. Se ejecuta el código del subprograma, usando y modificando el parámetro, es decir, el argumento
3. Cuando el subprograma termina, el identificador del parámetro "desaparece" pero el argumento puede haber sido modificado.

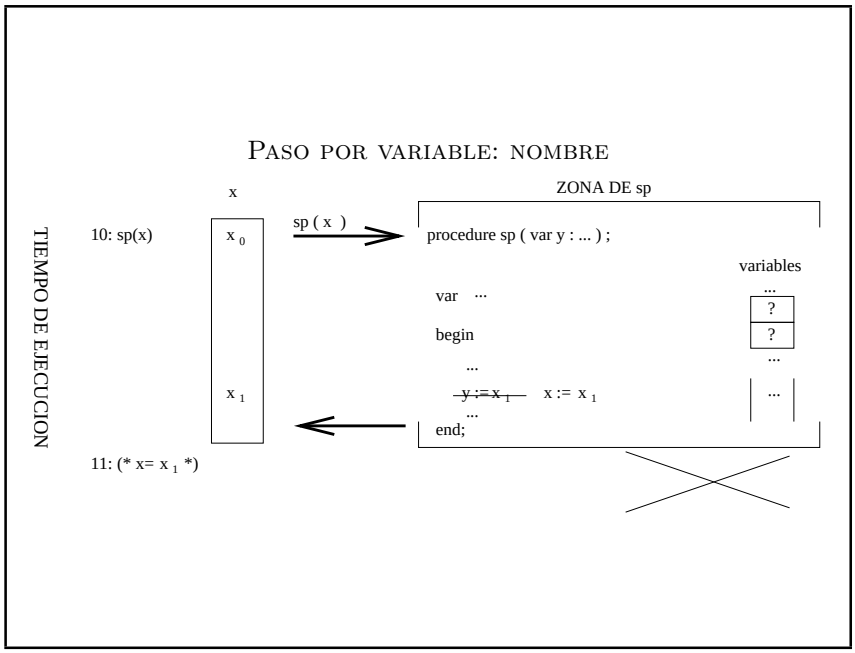
nombre

En el subprograma el identificador del parámetro se sustituye por el identificador del argumento

Slide 19



Slide 20



Slide 21

```
program p (input, output);
uses Crt ;
BEGIN
   ClrScr; (* Borra la pantalla *)
   writeln ('Voy a esperar 12 segundos');
   Delay(1200);
   writeln ('Y ahora escribir aes hasta la pulsación',
            ' de una tecla');
   Delay(600);
   repeat
     write ('a')
   until KeyPressed;
END.
```

Slide 22

```
program PruebaDeAleatorios (output);
var n : integer;
BEGIN
  for n := 1 to 5 do
     write (Random (10))
END.
```

Salida: 5 números aleatorios en $[0, 10)$, siempre los mismos.

```
program PruebaDeAleatorios (input, output);
var n : integer;
BEGIN
  readln (RandSeed);
  for n := 1 to 5 do
     write (Random (10))
END.
```

Salida: 5 números aleatorios en $[0, 10)$, dependientes del valor leído.

Slide 23

```
program PruebaDeAleatorios (output);
var n : integer;
BEGIN
  Randomize;
  for n := 1 to 5 do
     write (Random (10))
END.
```

Salida: 5 números aleatorios en $[0, 10)$, dependientes de cierto valor que depende de la hora del sistema.

Slide 24

UNIDADES FPC (fpcunits)

unit : conjunto de constantes, variables, tipos, funciones y procedimientos

por defecto: system

estándar fpc: crt dos getopts graph keyboard math mmx mouse objects objpas printer sockets strings sysutils typinfo video

según sistemas operativos: ...

específicas: ...

```
program ...
uses Crt, Math ;
...
BEGIN  ...END.
```

Slide 25

system *e/s de ficheros, de cadenas etc. Se usa por defecto*

Constantes: MAXINT ...

Tipos: byte word dword ShortInt SmallInt Integer LongInt Single Double Extended ...

Variables: input output stderr RandSeed ...

Funciones y procedimientos: los estándar y otras : pi power random randomize concat copy chdir paramcount ...

crt: *color y posición del cursor en la consola*

BLACK BLUE ... ClrEol ClrScr Delay DelLine GotoXY KeyPressed ReadKey TextBackground TextColor WhereX WhereY ...

dos: *acceso al S.O.*

DiskFree DiskSize GetDate GetTime SetDate SetTime ...

math: *rutinas matemáticas (trigonométricas, matriciales ...)*