

TECNOLOGÍAS EN EDUCACIÓN MATEMÁTICA



MODULO 17

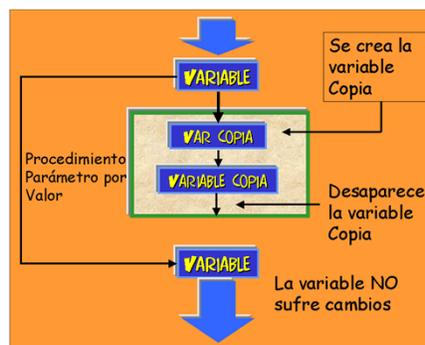
Dpto. de Ciencias e Ingeniería de la Computación
UNIVERSIDAD NACIONAL DEL SUR
Año 2019

Pascal Pasaje de Parámetros

RECORDEMOS...

Un **parámetro por valor** se inicializa en el momento de la invocación del procedimiento con el valor del parámetro real.

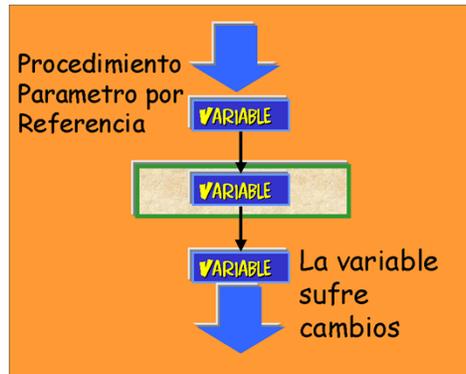
Al terminar la ejecución del procedimiento la variable asociada al parámetro formal se destruye y su valor se pierde.



Pascal Pasaje de Parámetros

RECORDEMOS...

Un **parámetro por referencia** introduce un nuevo nombre para referirse a la celda de memoria que corresponde al parámetro real.



Pascal Pasaje de Parámetros

```

Program Parametros
Var a,b: integer
Procedure Intercambio (x,y: integer)
Var aux: integer
Begin
    aux := x;
    x := y;
    y := aux;
    write ("Adentro: ",x); writeln (y)
End;
Begin
Write (" Ingrese el valor del primer numero:"); readln (a);
Write (" Ingrese el valor del segundo numero:"); readln (b);
Write ("Antes:", a); writeln (b);
Intercambio (a,b);
Write ("Despues:", a); write (b);
End.
    
```

PASAJE PARAMETROS POR VALOR

Supongamos que el usuario ingresa 66 y 80

Memoria	
1000	66 a
1001	80 b
1005	
1013	

Pantalla

Antes: 66 80

Pascal Pasaje de Parámetros

```

Program Parametros
Var a,b: integer
Procedure Intercambio (x,y: integer)
Var aux: integer
Begin
    aux := x;
    x := y;
    y := aux;
    write ("Adentro:",x); writeln (y)
End;
Begin
Write (" Ingrese el valor del primer numero:"); readln (a);
Write (" Ingrese el valor del segundo numero:"); readln (b);
Write ("Antes:", a); writeln (b);
Intercambio (a,b);
Write ("Despues:", a); write (b);
...
End.
        
```

PASAJE PARAMETROS POR VALOR
 Supongamos que el usuario ingresa 66 y 80

Memoria		
1000	66	a
1001	80	b
	...	
1201	66	x
1205	80	y
1206		aux

Pantalla

Antes: 66 80

Invocación

Pascal Pasaje de Parámetros

```

Program Parametros
Var a,b: integer
Procedure Intercambio (x,y: integer)
Var aux: integer
Begin
    aux := x;
    x := y;
    y := aux;
    write ("Adentro:",x); writeln (y)
End;
Begin
Write (" Ingrese el valor del primer numero:"); readln (a);
Write (" Ingrese el valor del segundo numero:"); readln (b);
Write ("Antes:", a); writeln (b);
Intercambio (a,b);
Write ("Despues:", a); write (b);
...
End.
        
```

PASAJE PARAMETROS POR VALOR
 Supongamos que el usuario ingresa 66 y 80

Memoria		
1000	66	a
1001	80	b
	...	
1201	80	x
1205	66	y
1206	66	aux

Pantalla

Antes: 66 80
 Adentro: 80 66

Pascal Pasaje de Parámetros

```

Program Parametros
Var a,b: integer
Procedure Intercambio (x,y: integer)
Var aux: integer
Begin
    aux := x;
    x := y;
    y := aux;
    write ("Adentro:",x); writeln (y)
End;
Begin
Write (" Ingrese el valor del primer numero:"); readln (a);
Write (" Ingrese el valor del segundo numero:"); readln (b);
Write ("Antes:", a); writeln (b);
Intercambio (a,b);
Write ("Despues:", a); write (b);
...
End.
```

PASAJE PARAMETROS POR VALOR

Supongamos que el usuario ingresa 66 y 80

Memoria		
1000	66	a
1001	80	b
1005	...	
1201		
1205		
1206		

Pantalla

Antes: 66 80
 Adentro: 80 66
 Despues: 66 80

Pascal Pasaje de Parámetros

```

Program Parametros
Var a,b: integer
Procedure Intercambio (var x,y: integer)
Var aux: integer
Begin
    aux := x;
    x := y;
    y := aux;
    write ("Adentro:",x); writeln (y)
End;
Begin
Write (" Ingrese el valor del primer numero:"); readln (a);
Write (" Ingrese el valor del segundo numero:"); readln (b);
Write ("Antes:", a); writeln (b);
Intercambio (a,b);
Write ("Despues:", a); write (b);
End.
```

PASAJE PARAMETROS POR VARIABLE

Supongamos que el usuario ingresa 66 y 80

Memoria		
1000	66	a
1001	80	b
1005		
1013		

Pantalla

Antes: 66 80

Pascal Pasaje de Parámetros

```

Program Parametros
Var a,b: integer
Procedure Intercambio (var x,y: integer)
Var aux: integer
Begin
    aux := x;
    x := y;
    y := aux;
    write ("Adentro:",x); writeln (y)
End;
Begin
Write (" Ingrese el valor del primer numero:"); readln (a);
Write (" Ingrese el valor del segundo numero:"); readln (b);
Write ("Antes:", a); writeln (b);
Intercambio (a,b);
Write ("Despues:", a); write (b);
...
End.
                
```

PASAJE PARAMETROS POR REFERENCIA
 Supongamos que el usuario ingresa 66 y 80

Memoria	
1000	66 → a
1001	80 → b
1201	... → x
1205	... → y
1206	... → aux

Pantalla

Antes: 66 80

Invocación

Pascal Pasaje de Parámetros

```

Program Parametros
Var a,b: integer
Procedure Intercambio (var x,y: integer)
Var aux: integer
Begin
    aux := x;
    x := y;
    y := aux;
    write ("Adentro:",x); writeln (y)
End;
Begin
Write (" Ingrese el valor del primer numero:"); readln (a);
Write (" Ingrese el valor del segundo numero:"); readln (b);
Write ("Antes:", a); writeln (b);
Intercambio (a,b);
Write ("Despues:", a); write (b);
...
End.
                
```

PASAJE PARAMETROS POR REFERENCIA
 Supongamos que el usuario ingresa 66 y 80

Memoria	
1000	80 → a
1001	66 → b
1201	... → x
1205	... → y
1206	66 → aux

Pantalla

Antes: 66 80
 Adentro: 80 66

Pascal Pasaje de Parámetros

```

Program Parametros
Var a,b: integer
Procedure Intercambio (var x,y: integer)
Var aux: integer
Begin
    aux := x;
    x := y;
    y := aux;
    write ("Adentro:",x); writeln (y)
End;
Begin
Write (" Ingrese el valor del primer numero:"); readln (a);
Write (" Ingrese el valor del segundo numero:"); readln (b);
Write ("Antes:", a); writeln (b);
Intercambio (a,b);
Write ("Despues:", a); write (b);
...
End.
        
```

PASAJE PARAMETROS POR VALOR

Supongamos que el usuario ingresa 66 y 80

Memoria		
1000	80	a
1001	66	b
1005	...	
1201		
1205		
1206		

Pantalla

Antes: 66 80
 Adentro: 80 66
 Despues: 80 66

Pascal Pasaje de Parámetros

Ejemplos donde **el pasaje de parámetros por referencia** es suficiente:
 sqr (x: real), promedio (a,b: integer), factorial (n: integer)

OBS: la primitiva trabaja con valores, no necesita cambiar los argumentos

Ejemplos donde **el pasaje de parámetros por variable** es necesario:
 intercambiar (var a,b: real), concatenar (cad1, cad2: char; var concat: char),
 DevolvemeAnteriorYSiguiente (num: integer; var ant, sig: integer)



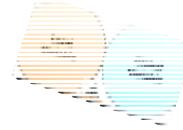
¿Por referencia o por valor?

He ahí la cuestión



Pascal Pasaje de Parámetros

Tipos de variables según su **ámbito**:



Variables globales: están definidas fuera de TODAS las primitivas Perduran cuando las primitivas terminan

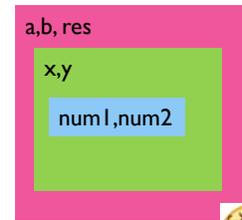
Variables locales: están definidas dentro de la primitiva Cuando la primitiva termina desaparecen

```
Program P
Var a,b, res: integer

Function MaximoComunDivisor (num1,num2: integer)

Function Calcular (x,y integer)
    MaximoComunDivisor (x,y)

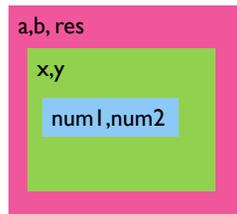
Begin
a:=8; b:=32;
res:=Calcular (a,b)
```



Pascal Pasaje de Parámetros

Alcance de una Variable: lugares donde esa variable es visible

- ¿Cuál es el alcance de res?
- ¿Cuál es el alcance de x?
- ¿Cuál es el alcance de num2?



Pascal Pasaje de Parámetros

Alcance de una Variable: lugares donde esa variable es visible

¿ Que variables hay y cual es su alcance?

```

program FuncionDoble;

function doble( numero: integer): integer;
begin
    doble := numero * 2;
end;

(* Cuerpo del programa *)
var
    n: integer;

begin
    write('Introduce un numero entero: ');
    readln( n );
    writeln('Su doble es: ', doble(n) );
    writeln('Y el doble de 5 es: ', doble(5) );
end.
    
```

Pascal Pasaje de Parámetros

Alcance de una Variable: lugares donde esa variable es visible

¿ Que variables hay y cual es su alcance?

```

program PruebaDePotencia;

function potencia(a,b: integer): integer; (* Definimos la funcion *)
var
    i: integer; (* Locales: para bucles *)
    temporal: integer; (* y para el valor temporal *)
begin
    temporal := 1; (* Inicializacion *)
    for i := 1 to b do
        temporal := temporal * a; (* multiplicamos "b" veces por "a" *)
    potencia := temporal; (* y finalmente devolvemos el valor *)
end;

var
    numero1, numero2: integer;

begin (* Cuerpo del programa *)
    writeln('Potencia de un numero entero');
    writeln;
    writeln('Introduce el primer numero');
    readln( numero1 );
    writeln('Introduce el segundo numero');
    readln( numero2 );
    writeln( numero1 , ' elevado a ', numero2 , ' vale ',
        potencia( numero1, numero2 ) );
end.
    
```

Pascal Pasaje de Parámetros

Alcance de una Variable: lugares donde esa variable es visible

- Cuando comienza la ejecución de un programa en Pascal se reservan una o más celdas en memoria para cada una de las **variables globales** (variables declaradas en el programa principal).
- Cuando un procedimiento o función se invoca se reservan celdas para cada una de las **variables locales al procedimiento o función**, como así también para los parámetros.
 - Para cada parámetro pasado por referencia se reserva una celda que mantiene la referencia al parámetro real.
 - **Todas estas celdas son liberadas cuando el procedimiento o función termina.**



Pascal Pasaje de Parámetros

Alcance de una Variable: lugares donde esa variable es visible

Pasos en la invocación a una primitiva:

- 1) Guardar lugar para parámetros formales, variables locales y nombre de la función (si lo es la primitiva)
- 2) Pasaje de parámetros: Copiar parámetros por valor, poner referencias en parámetros por referencia.
- 3) Ejecutar primitiva
- 4) Liberar espacio de la primitiva

Definición:

procedure p(pvalor: ---; var preferencia: ---); ...

Invocaciones posibles:

p(100, a)
p(a, a)
p(sqr(a)+30, a)



Pascal Pasaje de Parámetros

```

program paramProc;
Var pa1, pa2: integer;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
Begin
    local := pf1+pf2;
    pf1 := pf1+1; pf2 :=local+1;
    writeln(pf1, pf2, local);
End;
Begin
    p(pa1, -10);
    writeln(pa1:5,pa2:5);
End.
    
```

Variables GLOBALES

```

1 program error1;
2 var pa1,pa2:integer;
3 procedure p(pf1: integer; var pf2: integer);
4 var local:integer;
5 Begin
6     local := pf1+pf2;
7     pf1 := pf1+1; pf2 :=local+1;
8     writeln(pf1:5, pf2:5, local:5);
9 End;
10 Begin
11     p(pa1, -10);
12     writeln(pa1:5, pa2:5);
13 End.
14
    
```

Ventana de Mensajes

- Opciones cambiadas, recompila limpiando con -8
- error1.pas(1,14) Error: Variable identifier expected
- error1.pas(1,11) Hint: Found declaration: p(LongInt;var LongInt);
- error1.pas(12,15) Warning: Variable "pa1" does not seem to be initialized
- error1.pas(12,21) Warning: Variable "pa2" does not seem to be initialized
- error1.pas(15) Fatal: There were 1 errors compiling module, stopping

Error en compilación
 El parámetro real no puede ser una constante cuando el pasaje de parámetros es por referencia

Pascal Pasaje de Parámetros

```

program paramProc;
Var pa1, pa2: integer;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
Begin
    local := pf1+pf2;
    pf1 := pf1+1; pf2 :=local+1;
    writeln(pf1, pf2, local);
End;
Begin
    p(pa1, pa2+1);
    writeln(pa1:5,pa2:5);
End.
    
```

```

1 program error1;
2 var pa1,pa2:integer;
3 procedure p(pf1: integer; var pf2: integer);
4 var local:integer;
5 Begin
6     local := pf1+pf2;
7     pf1 := pf1+1; pf2 :=local+1;
8     writeln(pf1:5, pf2:5, local:5);
9 End;
10 Begin
11     p(pa1, pa2+1);
12     writeln(pa1:5, pa2:5);
13 End.
14
    
```

Ventana de Mensajes

- error1.pas(11,11) Warning: Variable "pa2" does not seem to be initialized
- error1.pas(11,16) Error: Variable identifier expected
- error1.pas(3,11) Hint: Found declaration: p(LongInt;var LongInt);
- error1.pas(12,15) Warning: Variable "pa1" does not seem to be initialized
- error1.pas(15) Fatal: There were 1 errors compiling module, stopping

Error en compilación
 El parámetro real no puede ser una expresión cuando el pasaje de parámetros es por referencia

Pascal Pasaje de Parámetros

```

Program parametros;
var pa1, pa2: integer;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
    local:=pf1+pf2;
    writeln(pf1, pf2, local);
    pf1:= pf1+1; pf2:=local+1;
    writeln(pf1, pf2, local);
end;
begin
    pa1:=1; pa2:=3;
    writeln(pa1, pa2);
    p(pa1, pa2);
end.
    
```

Pascal Pasaje de Parámetros

```

program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
    f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
Begin
    local := f(pf1) + f(pf2);
    pf1:= pf1+10; pf2 :=local+1;
    writeln(pf1, pf2, local);
End;
Begin
    pa1:=1; pa2:=5;
    p(pa1, pa2);
    writeln(pa1:5,pa2:5);
End.
    
```

pa1
pa2

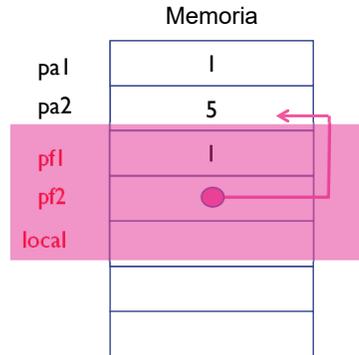
Memoria	
1	
5	

Se agrega la definición de una función

Pascal Pasaje de Parámetros

```

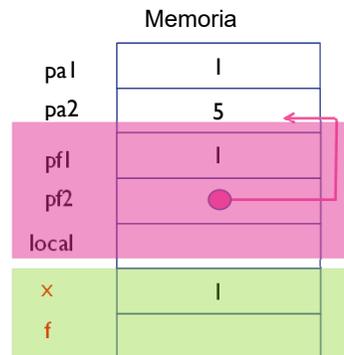
program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
  f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
→ begin
  local := f(pf1) + f(pf2);
  pf1:= pf1+10; pf2 :=local+1;
  writeln(pf1:5,pf2:5,local:5);
end;
begin
  pa1:=1; pa2:=5;
  p(pa1, pa2);
  writeln(pa1:5,pa2:5);
end.
  
```



Pascal Pasaje de Parámetros

```

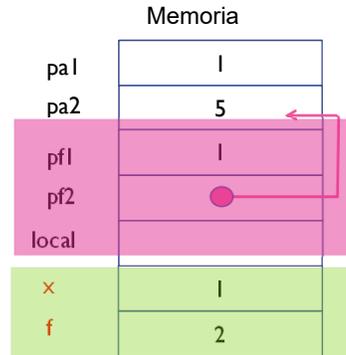
program parampyf;
var pa1, pa2: integer;
→ function f(x: integer): integer;
begin
  f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
  local := f(pf1) + f(pf2);
  pf1:= pf1+10; pf2 :=local+1;
  writeln(pf1:5,pf2:5,local:5);
end;
begin
  pa1:=1; pa2:=5;
  p(pa1, pa2);
  writeln(pa1:5,pa2:5);
end.
  
```



Pascal Pasaje de Parámetros

```

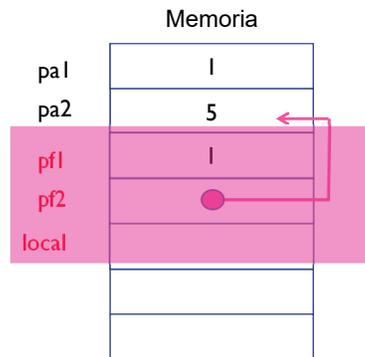
program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
    f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
    local := f(pf1) + f(pf2);
    pf1 := pf1 + 10; pf2 := local + 1;
    writeln(pf1:5, pf2:5, local:5);
end;
begin
    pa1:=1; pa2:=5;
    p(pa1, pa2);
    writeln(pa1:5, pa2:5);
end.
    
```



Pascal Pasaje de Parámetros

```

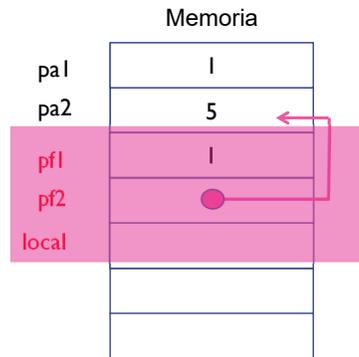
program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
    f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
    local := f(pf1) + f(pf2);
    pf1 := pf1 + 10; pf2 := local + 1;
    writeln(pf1:5, pf2:5, local:5);
end;
begin
    pa1:=1; pa2:=5;
    p(pa1, pa2);
    writeln(pa1:5, pa2:5);
end.
    
```



Pascal Pasaje de Parámetros

```

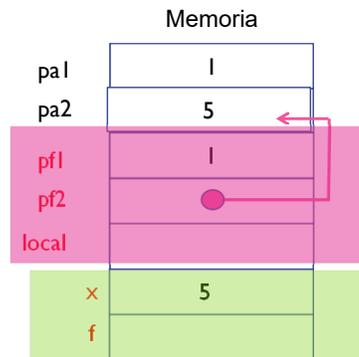
program parampyf;
var pa1, pa2: integer;
➔ function f(x: integer): integer;
begin
  f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
  local := f(pf1) + f(pf2);
  pf1:= pf1+10; pf2 :=local+1;
  writeln(pf1:5,pf2:5,local:5);
end;
begin
  pa1:=1; pa2:=5;
  p(pa1, pa2);
  writeln(pa1:5,pa2:5);
end.
  
```



Pascal Pasaje de Parámetros

```

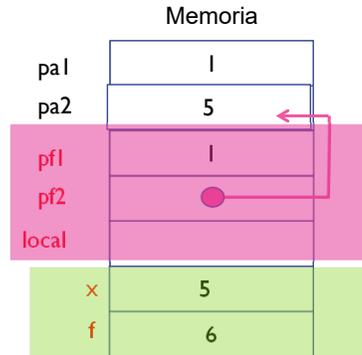
program parampyf;
var pa1, pa2: integer;
➔ function f(x: integer): integer;
begin
  f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
  local := f(pf1) + f(pf2);
  pf1:= pf1+10; pf2 :=local+1;
  writeln(pf1:5,pf2:5,local:5);
end;
begin
  pa1:=1; pa2:=5;
  p(pa1, pa2);
  writeln(pa1:5,pa2:5);
end.
  
```



Pascal Pasaje de Parámetros

```

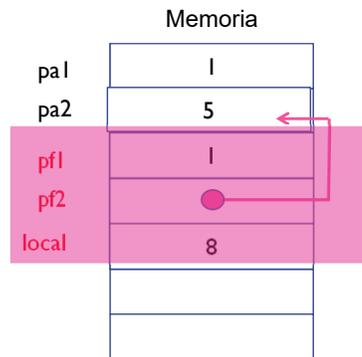
program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
    f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
    local := f(pf1) + f(pf2);
    pf1:= pf1+10; pf2 :=local+1;
    writeln(pf1:5,pf2:5,local:5);
end;
begin
    pa1:=1; pa2:=5;
    p(pa1, pa2);
    writeln(pa1:5,pa2:5);
end.
    
```



Pascal Pasaje de Parámetros

```

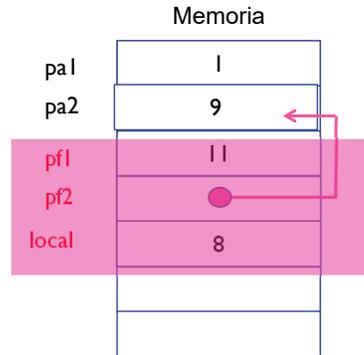
program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
    f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
    local := f(pf1) + f(pf2);
    pf1:= pf1+10; pf2 :=local+1;
    writeln(pf1:5,pf2:5,local:5);
end;
begin
    pa1:=1; pa2:=5;
    p(pa1, pa2);
    writeln(pa1:5,pa2:5);
end.
    
```



Pascal Pasaje de Parámetros

```

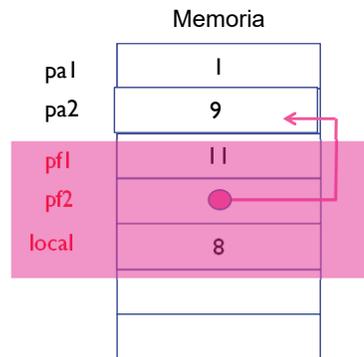
program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
  f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
  local := f(pf1) + f(pf2);
  pf1 := pf1 + 10; pf2 := local + 1;
  writeln(pf1:5, pf2:5, local:5);
end;
begin
  pa1 := 1; pa2 := 5;
  p(pa1, pa2);
  writeln(pa1:5, pa2:5);
end.
  
```



Pascal Pasaje de Parámetros

```

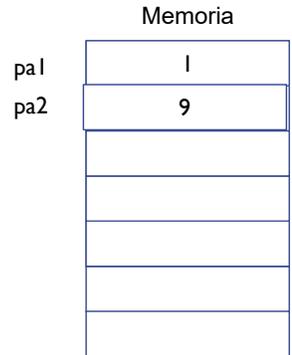
program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
  f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
  local := f(pf1) + f(pf2);
  pf1 := pf1 + 10; pf2 := local + 1;
  writeln(pf1:5, pf2:5, local:5);
end;
begin
  pa1 := 1; pa2 := 5;
  p(pa1, pa2);
  writeln(pa1:5, pa2:5);
end.
  
```



Pascal Pasaje de Parámetros

```

program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
  f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
  local := f(pf1) + f(pf2);
  pf1:= pf1+10; pf2 :=local+1;
  writeln(pf1:5,pf2:5,local:5);
end;
begin
  pa1:=1; pa2:=5;
  p(pa1, pa2);
  ↪ writeln(pa1:5,pa2:5);
end.
    
```



Pascal Pasaje de Parámetros

```

program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
  f:=x+1;
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
  local := f(pf1) + f(pf2);
  pf1:= pf1+10; pf2 :=local+1;
  writeln(pf1:5,pf2:5,local:5);
end;
begin
  pa1:=1; pa2:=f(pa1);
  p(f(pa1*2), f(pa2));
  writeln(pa1:5,pa2:5);
end.
    
```

¿Qué sucede si en lugar de invocar con paf2 invoco con f(pa2)?

Error en compilación
 El parámetro real no puede ser una expresión cuando el pasaje es por referencia

Pascal Pasaje de Parámetros

```

program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
    ...
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
    ...
end;
begin
    ...
end.
    
```

El ambiente de referenciamiento del programa principal se denomina **entorno global**.

Pascal Pasaje de Parámetros

```

program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
var local: ...;
begin
    ...
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
    ...
end;
begin
    ...
end.
    
```

El ambiente de referenciamiento de un sub-programa se compone del **entorno global + el entorno local**.

Pascal Pasaje de Parámetros

```

program parampyf;
var pa1, pa2: integer;
function f(x: integer): integer;
begin
    ...
end;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
    ...
end;
begin
    ...
end.
    
```

El ambiente de referenciamiento de un sub-programa se compone del entorno **global + el entorno local**.

Además siempre se pueden utilizar todos los identificadores predefinidos.

Pascal Pasaje de Parámetros

Aunque Pascal permite que el procedimiento p y la función f accedan a las variables globales pa1 y pa2, la programación modular recomienda no hacerlo y adoptaremos esa recomendación.

La programación modular establece que el bloque ejecutable de cada función y procedimiento acceda únicamente a sus parámetros formales y variables locales.

- La variable x es el parámetro formal de la función f y solo es visible en el bloque de la función.
- Las variables pf1, pf2 y local solo son visibles en el bloque de p.

Cuando p invoca a f las variables pf1, pf2 y local continúan almacenadas en memoria, pero no son accesibles desde el bloque ejecutable de f. Al terminar la ejecución de f, el control vuelve a p y sus parámetros y variables locales vuelven a ser accesibles.

Pascal Pasaje de Parámetros

Un programa en Pascal puede incluir procedimientos y funciones definidos por el programador.

En la **declaración** de un procedimiento o función se establece su **nombre** y la lista de **parámetros**.

Cuando un procedimiento o función ha sido definido puede ser **usado** mediante una instrucción de **invocación**.

Pascal Pasaje de Parámetros

ACTIVACIÓN DE UN PROCEDIMIENTO

- Un **procedimiento** se invoca desde una **instrucción**.
- Una vez invocado el control (la ejecución) pasa al procedimiento.
- El bloque de código que incluye la invocación, se suspende.
- Cuando comienza la ejecución se reserva espacio en memoria para las variables locales y para los parámetros por valor, que se inicializan con los valores de los parámetros reales. También para las referencias almacenadas en los parámetros por variable.

Pascal Pasaje de Parámetros

ACTIVACIÓN DE UN PROCEDIMIENTO

- El procedimiento se ejecuta y cuando termina, el control retorna, provocando algún **efecto**.
- El **efecto de la ejecución de un procedimiento** puede ser modificar uno o más parámetros por variable o producir una salida por consola o por archivo.
- El **bloque de código que había quedado suspendido se reanuda**, en la instrucción siguiente a la invocación.

Pascal Pasaje de Parámetros

ACTIVACIÓN DE UNA FUNCIÓN

- Una **función** se invoca desde una **expresión**.
- Una vez invocada el control (la ejecución) pasa a la función.
- El bloque de código que incluye la invocación, se suspende.
- Cuando comienza la ejecución se reserva espacio en memoria para las variables locales y para los parámetros por valor, que se inicializan con los valores de los parámetros reales.

Pascal Pasaje de Parámetros

ACTIVACIÓN DE UNA FUNCIÓN

- La función se ejecuta y cuando termina, el control retorna al lugar de la expresión desde donde se la llamó.
- El bloque de código que había quedado suspendido se reanuda, continuando con la evaluación de la expresión que contiene a la invocación.
- El efecto de la ejecución de una función es retornar un valor.

TECNOLOGÍAS EN EDUCACIÓN MATEMÁTICA



FIN MODULO 17

Dpto. de Ciencias e Ingeniería de la Computación

UNIVERSIDAD NACIONAL DEL SUR

Año 2019