

Práctico 7 - Subprogramas (Funciones y Procedimientos)

Programación 1
InCo - Facultad de Ingeniería, Udelar

Pasaje por valor: Son los parámetros no precedidos por VAR (marcados en Amarillo).

Pasaje por referencia: Son los parámetros precedidos por VAR (también se los denomina parámetros de variables).

1. (a) Explique la diferencia entre un parámetro pasado por valor y uno pasado por referencia.
- (b) Identifique cuáles parámetros están pasados por valor y cuáles están pasados por referencia en los siguientes encabezados de procedimientos y funciones:
 - I) `function areaRectangulo(largo, ancho : real) : real;`
 - II) `procedure leerTriangulo(var base, altura : real);`
 - III) `procedure perimetroAreaCuadrado(lado : real; var perimetro, area : real);`
- (c) Explique la diferencia entre un parámetro de entrada y un parámetro de salida. ¿Cómo se relacionan con los dos tipos de pasaje de parámetros vistos?
- (d) Indique cuáles de las siguientes invocaciones son correctas. Suponga que las variables x,y,z son reales y que las variables i,j son enteras. Explique.

- | | |
|--|---|
| <input type="checkbox"/> leerTriangulo(x, y) | <input type="checkbox"/> areaRectangulo(x, y) |
| <input type="checkbox"/> leerTriangulo(i, j) | <input type="checkbox"/> perimetroAreaCuadrado(2.5, y, z) |
| <input type="checkbox"/> leerTriangulo(2.2, 3.5) | <input type="checkbox"/> perimetroAreaCuadrado(x, 3.2, 4.7) |
| <input type="checkbox"/> z := areaRectangulo(3, 5) | |

2. Dado el siguiente fragmento de programa:

```
program Pr7Ej2
...
procedure inicio(tiempo, espacio: real; var dia: real; signo: char);
begin {cuerpo del procedimiento}
...
end;
...
begin {programa principal}
...
  inicio(3.5, 6.0, hora, 'Z');
...
end.
```

- (a) Explique la diferencia entre un parámetro nominal (o formal) y un parámetro efectivo (o verdadero). Indique cuáles son los parámetros nominales en el programa.
 - (b) ¿Es necesario que los parámetros efectivos sean siempre variables? Explique.
 - (c) En caso de que los parámetros efectivos sean variables, ¿es necesario que tengan el mismo nombre que sus correspondientes parámetros nominales? Explique.
3. Dado el siguiente procedimiento:

```
procedure prueba(x: real; y: integer; var z: real);
```

Indique cuáles de las siguientes invocaciones son correctas. Suponga que en el contexto en el que se hacen las invocaciones, la variable x es de tipo real y la variable n es de tipo integer.

- | | |
|--|--|
| <input type="checkbox"/> prueba(1, 2.0, x) | <input type="checkbox"/> prueba(5*n, round(7.3), x) |
| <input type="checkbox"/> prueba(n, 3, x) | <input type="checkbox"/> prueba(x, 3, x); |
| <input type="checkbox"/> prueba(n, 3, 2.0) | <input type="checkbox"/> prueba(Prueba(5, 33.8, x), 92, x) |
| <input type="checkbox"/> prueba(1, 3, n) | |

4. Dado el siguiente programa:

```

program Pr7Ej4;
var tum, num, temp : integer;

procedure proc(a, b : integer; var c : integer);
var aux : integer;
begin
    aux := a * b;
    aux := aux + 1;
    c := aux + a;
    writeln(a, b, c, aux)
end;

{ Programa principal }
begin
    tum := 1;
    num := 2;
    proc(tum, num, temp);
    writeln(temp);
    tum := 0;
    num := 1;
    proc(tum, num, temp);
    writeln(temp)
end.

```

Por valor Se envía una copia del valor, sin afectar la variable original.

Por referencia: Se pasa el mismo valor, modificando directamente la

- (a) Indique cuáles son los parámetros formales del procedimiento *proc*. Indique cuáles de ellos son parámetros por valor y cuáles son parámetros por referencia.
 - (b) Indique cuáles son los parámetros efectivos que aparecen en el programa.
 - (c) Indique qué se exhibirá al ejecutar el programa.
5. (a) Escriba un procedimiento llamado *corrimiento* con tres parámetros enteros: *a*, *b* y *c*. El procedimiento debe hacer un corrimiento a la derecha de los valores de los parámetros de manera que, después de la invocación, el valor que originalmente estaba en *a* quede en *b*, el que estaba en *b* quede en *c* y el que estaba en *c* quede en *a*.
- (b) Escriba un programa principal que lea tres valores y exhiba el resultado de invocar al procedimiento *corrimiento*.

Ejemplo:

Ingrese tres números: 4 1 7
El corrimiento es: 7 4 1

6. Sea el siguiente encabezado del procedimiento *raices* que calcula las raíces reales de un polinomio de segundo grado, de la forma $ax^2 + bx + c$.

```

procedure raices(a,b,c : integer; cant : integer; raiz1, raiz2 : real)

```

donde:

- *a*, *b* y *c* son los coeficientes del polinomio.
- *cant* indica la cantidad de raíces reales diferentes(0, 1 ó 2).
- *raiz1* y *raiz2* son las raíces reales del polinomio. Si el polinomio tiene una raíz real, se almacena en *raiz1*. Si el polinomio tiene dos raíces reales diferentes, se almacenan en *raiz1* y *raiz2*.

Ejemplos:

Entrada: a = 2, b = -3, c = -2
Resultado: cant = 2, raiz1 = 2, raiz2 = -0.5
Entrada: a = 1, b = -6, c = 9
Resultado: cant = 1, raiz1 = 3, raiz2 = ?
Entrada: a = 5, b = 4, c = 1
Resultado: cant = 0, raiz1 = ?, raiz2 = ?

- (a) Determine para cada parámetro del encabezado del procedimiento si debe ser pasado por *valor* o *referencia*.

Modifique el encabezado anterior según su respuesta.

- (b) Escriba el cuerpo del procedimiento **raices** (análogo a los casos de raíces reales del ejercicio 12, práctico 3).

- (c) Escriba un programa principal que lea los tres coeficientes e invoque al procedimiento para calcular las raíces del polinomio. El programa exhibe las raíces devueltas por el procedimiento.

7. (a) Escriba una función llamada *distancia* que tenga cuatro parámetros de entrada (reales) llamados **x1**, **y1**, **x2**, **y2**, que representan las coordenadas en el plano de los puntos $(x1, y1)$ y $(x2, y2)$. La función debe calcular y retornar la distancia entre ambos puntos.
- (b) Escriba un programa principal que lea las coordenadas de dos puntos y exhiba la distancia entre ambos. El programa debe invocar a la función *distancia*.

Ejemplo:
Ingrese las coordenadas del primer punto: 1 2 Ingrese las coordenadas del segundo punto: 5 7 La distancia entre los puntos es: 6.40

8. (a) Escriba una función llamada *esPrimo* que, dado un entero mayor o igual que 0, devuelva **true** si es primo y **false** en caso contrario. En el ejercicio 7 del práctico 5 se escribió un programa principal que resolvía el mismo algoritmo.
- (b) Escriba un programa principal que resuelva el ejercicio 8 del práctico 6 invocando a la función *esPrimo*. El programa lee dos enteros positivos **m** y **n** y exhibe todos los números primos en el rango. En caso de que **m** sea mayor que **n**, no se exhibirá ninguno.
9. (a) Escriba una función llamada *multiplo* que, dados dos enteros positivos **m** y **n**, devuelva **true** si **m** es múltiplo de **n** o si **n** es múltiplo de **m**. En otro caso contrario, devuelve **false**.
- (b) Escriba un programa principal, que lea, en una misma línea, parejas de enteros positivos e invoque a la función *multiplo* para indicar si uno de los dos números es múltiplo del otro. La secuencia de números terminar con el valor -1.

Ejemplo:
4 7 No 4 8 Si 16 8 Si -1

10. (a) Escriba el siguiente procedimiento llamado *multiplicidadFactor* que, dado un entero positivo **numero** y un entero positivo **factor**, devuelve en **multiplicidad** la cantidad máxima de veces que **factor** divide a **n** y en **residuo** el resto.

```
procedure multiplicidadFactor(numero, factor : integer;
                             var multiplicidad, residuo : integer);
```

Ejemplos:
Entrada: numero = 39, factor = 2 Salida: multiplicidad = 0, residuo = 39
Entrada: numero = 39, factor = 3 Salida: multiplicidad = 1, residuo = 13
Entrada: numero = 42, factor = 2 Salida: multiplicidad = 1, residuo = 21
Entrada: numero = 12, factor = 2 Salida: multiplicidad = 2, residuo = 3

- (b) Escriba un programa principal que lea una secuencia de enteros positivos mayores a 1 y exhiba la descomposición en factores primos de cada número. La secuencia de números termina con el valor -1. El programa debe invocar a la función *multiplicidadFactor*.

Ejemplo:
<pre>39 17 1517 42 12 18 -1 39 = 3 * 13 17 = 17 1517 = 37 * 41 42 = 2 * 3 * 7 12 = 2 * 2 * 3 18 = 2 * 3 * 3</pre>