



# COMPUTACIÓN (7501)

“No mires nunca de donde vienes,  
sino a donde vas.”  
BEAUMARCHAIS, Pierre Augustin

## FUNCIONES Y PROCEDIMIENTOS

### Clase 7

Facultad de INGENIERÍA    Enero-Febrero, 2008

---

---

---

---

---

---

---

---

“No mires nunca de donde vienes, sino a donde vas.” BEAUMARCHAIS, P. A.

## Agenda

- Revisión, Estructuras
- Programación **Modular**
- **Funciones**
- **Procedimientos**
- **Comunicación** entre subprogramas
- **Ámbitos** de las variables
- **Funciones Recursivas**
- **Reglas Básicas**

Facultad de INGENIERÍA    U B A    2008

---

---

---

---

---

---

---

---

“No mires nunca de donde vienes, sino a donde vas.” BEAUMARCHAIS, P. A.

1. Concepto de subprograma
2. Procedimientos
3. Funciones
4. Comunicación entre subprogramas
5. Ejemplos. Conceptos básicos
6. Materiales y bibliografía

Facultad de INGENIERÍA    U B A    2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Revisión

Todo programa puede ser desarrollado utilizando solamente tres tipos de estructuras de control:

- Secuenciales
- Selectivas
- Repetitivas

Facultad de INGENIERÍA UBA 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Revisión

- Secuenciales - Selectivas - Repetitivas

Facultad de INGENIERÍA UBA 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Revisión

Estr. Selectivas	{	If	{	for
		Case switch		
Estr. Repetitivas	{	Contador	{	while
		Bandera		

Facultad de INGENIERÍA UBA 2008

---

---

---

---

---

---

---

---

## Revisión

Ejemplos de combinación de condiciones:

Dos o más **Condiciones Lógicas Simples** pueden ser relacionadas por medio de **conectores** formar **Condiciones Compuestas**.

Conectores { or |  
and &



---

---

---

---

---

---

---

---

## Revisión

**Anidar** significa colocar una estructura dentro de otra.

Esto quiere decir que una estructura contiene en su interior a otra.

Las estructuras pueden ser de igual o de distinto tipo, siempre que se cumpla con las condiciones siguientes:



---

---

---

---

---

---

---

---

## Anidación



---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Programación Modular

Probabilidades según la cantidad de líneas

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Subprogramas

**Técnica:** resolución de problemas consiste en dividir el problema principal en subproblemas más pequeños o módulos: descomposición **modular**.

Cada subprograma resuelve como máximo un subproblema, quiere decir que realiza una y solo una tarea específica.

En Pascal se tienen funciones y procedimientos.  
En MatLab sólo funciones

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Subprogramas

**Método:** **diseño descendente** desde el problema principal:

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

## Subprogramas

La modularización facilita:

- la escritura y la lectura de los programas,
- la comprobación individual,
- la división de tareas de programación
- el diseño descendente y la reutilización del código a través de librerías.



---

---

---

---

---

---

---

---

## Subprogramas en Pascal

- la declaración de los subprogramas debe hacer antes del **begin** del cuerpo del programa principal,
- la declaración de los subprogramas conviene que se realice previamente a la declaración de variables **var**,
- si un subprograma es llamado por otro, el que es llamado debe ser declarado antes del que lo llama.



---

---

---

---

---

---

---

---

## Subprogramas en MatLab

**MatLab** almacena cada subprograma como un programa independiente en el directorio **work** y puede ser llamado por cualquier otro programa



---

---

---

---

---

---

---

---

## Funciones

1. Son subprogramas que pueden **recibir o no** del programa parámetros (datos) y **siempre** devuelven **un** resultado al programa que es el valor de esa función.
2. El valor se obtiene cada vez que se requiere, ejecutándose el código de dicha función.
3. Se referencian a través de una asignación.



## Funciones en Pascal

Program xxxxxxxx;

Uses crt, dos;

**Function** nombref (p1, p2:tipo): tipo;

{declaraciones locales y subprogramas}

**begin**

<cuerpo de la función>

nombref := valor de la función; { \*sentencia de asignación\* }

**end;**

**Var** va1, va2, xxxx : .....;

**Begin**

R := nombref (va1, va2);

→ Asignación

**End.**

**Cabecera**  
Parámetros formales su tipos  
y tipo de datos que devuelve

→ **Declaraciones**  
(variables, otros procedimientos)  
→ **Cuerpo.**

## Funciones en MatLab

**function** variable=nombref (p1, p2);

<cuerpo de la función>

variable := valor de la función; { \*sentencia de asignación\* }

R := nombref (va1, va2);

→ Asignación

**Cabecera**  
Parámetros formales su tipos  
y tipo de datos que devuelve



→ **Cuerpo.**

**Nombre del archivo  
es el de la función**

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Funciones en Pascal

EJEMPLOS

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

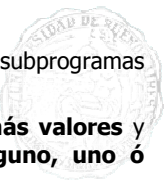
---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Procedimientos en Pascal

1. Pueden ser **"invocados"** por otros subprogramas o por el programa principal.
2. Pueden recibir **ninguno, uno ó más valores** y pueden devolver al llamador **ninguno, uno ó más valores**.



Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Procedimientos en Pascal

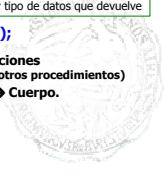
```

Program xxxxxxxx;
Uses crt, dos;
Procedure nombreproc (p1, p2:tipo; var p3: tipo);
{declaraciones locales y subprogramas}
begin
  <cuerpo de la función>
  p3 := .....; {*sentencia de asignación*}
end;
Var va1, va2, va3, xxxx : .....
Begin
  nombreproc (va1, va2, va3);
End.
  
```

**Cabecera**  
Parámetros formales su tipos y tipo de datos que devuelve

**Declaraciones**  
(variables, otros procedimientos)  
→ **Cuerpo.**

→ **Invocación**



Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---


---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Procedimientos en Pascal

**EJEMPLOS**

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

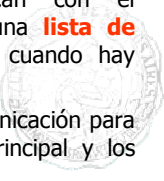
"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Comunicación. Parámetros

Los subprogramas se comunican con el programa principal a través de una **lista de parámetros**. Existen parámetros cuando hay comunicación.

Los **parámetros** son canales comunicación para pasar datos entre el programa principal y los subprogramas y para devolverlos.

Los parámetros se denominan **actuales** o **reales** los del **programa** y **formales** los del proceso.



Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

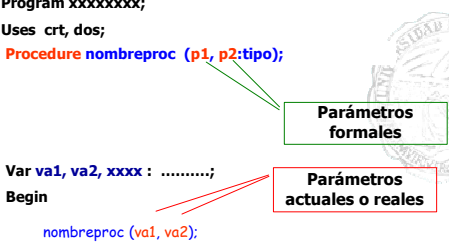

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Comunicación. Parámetros

```

Program xxxxxxxx;
Uses crt, dos;
Procedure nombrepoc (p1, p2:tipo);

Var va1, va2, xxxx : .....;
Begin
    nombrepoc (va1, va2);
End.
  
```

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---



"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Comunicación Pasaje por valor

```

Program xxxxxxxx;
Uses crt, dos;
Procedure nombreproc (p1, p2:tipo);

  Var va1, va2, xxxx : .....;
  Begin
    nombreproc (va1, va2);
  End.
  
```

va1  
va2

p1 = va1  
p2 = va2

Modificaciones hechas en p1 (y/o p2) no modifican el valor de va1 (o va2);

Facultad de INGENIERÍA    U B A    2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Comunicación Pasaje por referencia

```

Program xxxxxxxx;
Uses crt, dos;
Procedure nombreproc (p1, p2:tipo; var p3:tipo);

  Var va1, va2, va3, xxxx : .....;
  Begin
    nombreproc (va1, va2, va3);
  End.
  
```

va1  
va2  
va3

p3 =

p1 = va1  
p2 = va2

Modificaciones hechas en p3 **modifican** el valor de va3

Facultad de INGENIERÍA    U B A    2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Variables Globales y Locales

```

Program xxxxxxxx;
Uses crt, dos;
Procedure nombreproc (p1, p2:tipo);
  Var a, b, c : .....;
  Begin
    .....
  End;
  Var va1, va2, xxxx : .....;
  Begin
    nombreproc (va1, va2);
  End.
  
```

Variables Locales

Variables Globales

Facultad de INGENIERÍA    U B A    2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Variables Globales y Locales

```

Program xxxxxxxx;
Uses crt, dos;
Procedure nombreproc (p1, p2:tipo);
  Var a, b, c : .....;
  Begin
    va1 := ..... + va2;
  End;
Var va1, va2, xxx : .....;
Begin
  nombreproc (va1, va2);
End.

```

**Nunca una variable Global se debe usar en una función o procedimiento**

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Ámbito de las Variables Locales y Globales

El **ámbito** es el espacio de validez o dominio de visibilidad de una variable.

Una variable **global** se declara en el programa principal y está disponible durante su ejecución.

**Nunca** debe usarse una variable global en un subprograma

Una variable **local** se declara dentro de un subprograma (procedimiento o función) y sólo está disponible durante la ejecución del subprograma.

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Ámbito de las Variables Globales y Locales

```

Program xxxxxxxx;
Uses crt, dos;
Procedure nombreproc (p1, p2:tipo);
  Var a, b, c : .....;
  Begin
    .....
  End;
Var va1, va2, a, xxx : .....;
Begin
  nombreproc (va1, va2);
End.

```

**Variables Locales** (pointing to the procedure's local variables)

**Variables Globales** (pointing to the global variables)

$a \neq a$

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Funciones Recursivas

Para que una función sea "recursiva" debe cumplir dos condiciones:

1. En el cuerpo de ella **debe** haber un **llamado a sí misma**.
2. **Debe** tener una **salida por último caso**.



Facultad de INGENIERÍA

U B A

2008

M  
g  
i  
n  
g  
F  
e  
r  
n  
a  
n  
d  
o  
J  
L  
A  
G  
E

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Funciones Recursivas

```
Program xxxxxxxx;  
Uses crt, dos;  
function factorialR (p:byte):longint;  
begin  
    if p<=1 then factorialR := 1  
    else factorialR :=p*factorialR(p-1)  
end;  
Var va1, va2, n, xxxx : .....;  
Begin  
    R:= factorialR (n);  
End.
```



Facultad de INGENIERÍA

U B A

2008

M  
g  
i  
n  
g  
F  
e  
r  
n  
a  
n  
d  
o  
J  
L  
A  
G  
E

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Funciones Recursivas

```
function v = factorialR (p);  
begin  
    if p<=1    factorialR = 1;  
    else factorialR = p * factorialR(p-1);  
end;
```



Facultad de INGENIERÍA


U B A

2008

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Reglas Básicas

1. La interdependencia entre los subprogramas se denomina acoplamiento y siempre se desea que sea mínimo.
2. La cantidad de responsabilidades de un subprograma se denomina cohesión. Siempre se desea una alta cohesión. (Una única responsabilidad.

 Se recomienda usar transferencia de información a través las listas de parámetros utilizando la cantidad mínima posible de ellos (Regla básica!!).

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## Preguntas y Respuestas

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---

"No mires nunca de donde vienes, sino a donde vas." BEAUMARCHAIS, P. A.

## FIN

Facultad de INGENIERÍA U B A 2008

---

---

---

---

---

---

---

---