

# cÓNQUER



## Pseudocódigo en *PseInt (V)*

Módulo 2: IDE, Pseudocódigo y Python para principiantes



# Contenidos

1. Completando nuestra “**caja de herramientas**” (por el momento... 😊)
  - a. Variables (entero, real, lógica, texto)
  - b. Estructuras de datos: arrays unidimensionales y bidimensionales
  - c. Instrucciones lectura/escritura
  - d. Funciones: matemáticas y de cadenas de texto
  - e. Estructura condicional SI-ENTONCES
  - f. Estructura selectiva SEGÚN
  - g. Estructuras iterativas MIENTRAS, REPETIR - HASTA QUE, PARA
- h. **Funciones propias**
2. Algunos **ejemplos** sobre lo aprendido
3. **Ejercicios** de la semana
4. Y en la **próxima clase...**

# 1. Nuestra “caja de herramientas” actual (i)

## 1. Variables

- Es un espacio en memoria que tiene asignado un identificador (*sumaTotal*) donde se guarda un dato de un tipo en particular
- **Tipo de datos:** entero, real, texto y lógico
- **Operaciones:** definición (identificador + tipo de dato), inicialización y modificación

```
Definir sumaTotal Como Entero
```

```
sumaTotal = 12
```

```
sumaTotal = sumaTotal + 4
```

# 1. Nuestra “caja de herramientas” actual (i)

## 2. Estructuras de datos: arrays unidimensionales y bidimensionales

- Combinación de varios espacios en memoria que tiene asignado un identificador (*edadesAlumnos*) donde se guardan varios datos de un tipo en particular
- Tienen un **tamaño determinado**
- **Operaciones:** definición (identificador + tipo de dato + dimensión), inicialización y modificación

### Array unidimensional

```
Definir edadesAlumnos Como Entero  
Dimension edadesAlumnos[20]
```

```
Para i = 0 Hasta 19 Con Paso 1 Hacer  
    edadesAlumnos[i] = 0 //Cada posición, se inicializa a 0  
FinPara
```

```
//Solo se pueden guardar números enteros  
edadesAlumnos[1] = 16  
edadesAlumnos[9] = edadesAlumnos[1]
```

### Array bidimensional

```
Definir edadesAlumnosClase Como Entero  
Dimension edadesAlumnosClase[3,20]
```

```
Para filas = 0 Hasta 2 Con Paso 1 Hacer  
    Para columnas = 0 Hasta 19 Con Paso 1 Hacer  
        edadesAlumnosClase[filas,columnas] = 0  
    FinPara  
FinPara
```

```
edadesAlumnosClase[2,19] = 14  
edadesAlumnosClase[1,2] = edadesAlumnosClase[2,19]
```

# 1. Nuestra “caja de herramientas” actual (ii)

## 3. Instrucciones lectura/escritura

- **Lectura:** lee un dato introducido por consola y lo almacena en una variable/posición del array
- **Escritura:** escribe en consola el valor de variables/posiciones del array

```
Leer precio
```

```
Escribir "El precio es: ", precio
```

```
Escribir "El precio es: ", precio Sin Saltar
```

```
Leer edadesAlumnos[2]
```

```
Escribir edadesAlumnos[2]
```

```
Leer edadesAlumnosClase[2,19]
```

```
Escribir edadesAlumnosClase[2,19]
```

# 1. Nuestra “caja de herramientas” actual (v)

## 4. Funciones matemáticas

- Permiten calcular el valor de una magnitud a partir del valor de otra

- Valor absoluto: devuelve el número “sin signo”

```
resultado = abs(-3.6) //resultado = 3.6
```

- Valor truncado: devuelve la “parte entera” de un número real

```
resultado = trunc(-3.6) //resultado = -3
```

- Valor redondeado: “aproxima” la parte entera del número

```
resultado = redon(-3.6) //resultado = -4
```

- Valor al azar: devuelve un número aleatorio entre “0” y “numero-1”

```
resultado = azar(21) //resultado = [0 - 20]
```

# 1. Nuestra “caja de herramientas” actual (v)

## 5. Funciones de cadenas de texto

- Permiten realizar operaciones con variables de tipo *texto*

- Longitud: devuelve la cantidad de caracteres de la cadena

```
resultado = Longitud("Nota de texto") //resultado = 13
```

- SubCadena: devuelve una parte de la cadena

```
resultado = SubCadena("Nota de texto", 4, 13) //resultado = " es de texto"
```

- Concatenar: devuelve la unión de dos cadenas

```
resultado = Concatenar("Hola a ", "todos!") //resultado = "Hola a todos!"
```

# 1. Nuestra “caja de herramientas” actual (v)

## 5. Funciones de cadenas de texto

- Permiten realizar operaciones con variables de tipo *texto*

- ConvertirANumero: convierte un cadena en un número

```
resultado = ConvertirANumero("23") //resultado = 23
```

- ConvertirATexto: convierte un número en una cadena de caracteres

```
resultado = ConvertirATexto(23) //resultado = "23"
```

- Mayusculas: devuelve la cadena en mayúsculas

```
resultado = Mayusculas("Hola a todos!") //resultado = "HOLA A TODOS!"
```

- Minusculas: devuelve la cadena en minúsculas

```
resultado = Minusculas("Hola A Todos!") //resultado = "hola a todos!"
```

# 1. Nuestra “caja de herramientas” actual (iii)

## 6. Estructura condicional SI-ENTONCES

- Permite ejecutar un código diferente en función de una o varias expresiones lógicas
- Posibilidad de anidación de varias estructuras condicionales

```
Definir num Como Entero
num = 0

Escribir "Introduce un número entero de dos cifras"
Leer num

Si NO (num > 0) Entonces
    Escribir "No has introducido un número entero"
SiNo
    Si (num ≥ 10) Y (num ≤ 99) Entonces
        Escribir "Bien! El número tiene dos cifras"
    SiNo
        Escribir "El número no tiene dos cifras..."
    FinSi
FinSi
```

# 1. Nuestra “caja de herramientas” actual (iv)

## 7. Estructura selectiva SEGÚN

- Permite ejecutar un código diferente en función del valor de una variable

```
Definir nombre Como Texto
nombre = ""

Escribir "Introduce tu nombre"
Leer nombre

Segun nombre Hacer
    "Juan":
        Escribir "Bienvenido Juan!"
    "Maria":
        Escribir "Bienvenida Maria!"
    "Pepa":
        Escribir "Bienvenida Pepa!"
De otro Modo:
    Escribir "Bienvenido, seas quien seas!"
FinSegun
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 8. Estructuras iterativas (bucles)

- Es una secuencia de instrucciones que se ejecuta varias veces, mientras se cumpla cierta condición

**MIENTRAS**

**REPETIR - HASTA QUE**

**PARA**

# 1. Nuestra “caja de herramientas” actual (vi)

## 8. Estructuras iterativas (bucles)

### MIENTRAS

- Se ejecuta “mientras” se cumpla cierta condición (exp/op lógica)

```
Definir temporizador Como Entero  
temporizador = 0  
  
Escribir "Introduce el número de segundos del temporizador"  
Leer temporizador  
  
Escribir "Comienza el temporizador..."  
Mientras (temporizador > 0) Hacer  
    Escribir "Quedan ", temporizador, " segundos"  
    Esperar 1 Segundos  
    temporizador = temporizador - 1  
FinMientras  
Escribir "El temporizador ha finalizado!!"
```

### NOTAS!!

- Si la primera vez, la op/expr lógica es **FALSA**, el bucle **NO SE EJECUTARÁ NINGUNA VEZ**
- Si la op/expr lógica es siempre **VERDADERA**, será un bucle **INFINITO**

# 1. Nuestra “caja de herramientas” actual (vi)

## 8. Estructuras iterativas (bucles)

### REPETIR - HASTA QUE

- Similar al bucle “mientras”, solo que la exp/op lógica se evalúa al final (ejec. 1<sup>a</sup> vez)

```
Definir temporizador Como Entero  
temporizador = 0  
  
Escribir "Introduce el número de segundos del temporizador"  
Leer temporizador  
  
Escribir "Comienza el temporizador..."  
Repetir  
    Escribir "Quedan ", temporizador, " segundos"  
    Esperar 1 Segundos  
    temporizador = temporizador - 1  
Hasta Que (temporizador ≤ 0)  
Escribir "El temporizador ha finalizado!!"
```

### NOTA!!

Si la op/expr lógica es siempre **VERDADERA**, será un bucle **INFINITO**

# 1. Nuestra “caja de herramientas” actual (vi)

## 8. Estructuras iterativas (bucles)

### PARA

- El nº de repeticiones depende del valor de una variable, que toma un valor inicial y se va incrementando/decrementando en cada repetición

```
Definir i, temporizador Como Entero
temporizador = 0
i = 0

Escribir "Introduce el número de segundos del temporizador"
Leer temporizador

Escribir "Comienza el temporizador..."
Para i = temporizador Hasta 1 Con paso -1 Hacer
    Escribir "Quedan ", i, " segundos"
    Esperar 1 Segundos
FinPara
Escribir "El temporizador ha finalizado!!"
```

### NOTA!!

La variable “i” del bucle **PARA** tiene que **DEFINIRSE**, como cualquier otra variable

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

### Funciones pre-definidas en PseInt

[ - ] Func. Matemáticas  
abs (valor absoluto)  
trunc (valor truncado)  
redon (valor redondeado)  
raiz (raíz cuadrada)  
sen (seno)  
cos (coseno)  
tan (tangente)  
asen (arcoseno)  
acos (arcocoseno)  
atan (arcotangente)  
ln (logaritmo natural)  
exp (func. exponencial)  
azar (numero aleatorio)

[ - ] Func. p/Cadenas  
Longitud  
SubCadena  
Concatenar  
ConvertirANumero  
ConvertirATexto  
Mayusculas  
Minusculas

resultado = abs(num)

subcad = SubCadena(nota, inicio, 13)  
long = Longitud(subcad)

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

### Funciones pre-definidas en PseInt

[ - ] Func. Matemáticas  
abs (valor absoluto)  
trunc (valor truncado)  
redon (valor redondeado)  
raiz (raíz cuadrada)  
sen (seno)  
cos (coseno)  
tan (tangente)  
asen (arcoseno)  
acos (arcocoseno)  
atan (arcotangente)  
ln (logaritmo natural)  
exp (func. exponencial)  
azar (numero aleatorio)

[ - ] Func. p/Cadenas  
Longitud  
SubCadena  
Concatenar  
ConvertirANumero  
ConvertirATexto  
Mayusculas  
Minusculas

Parámetros  
resultado = abs(num)  
subcad = SubCadena(nota, inicio, 13)  
long = Longitud(subcad)

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

### Funciones pre-definidas en PseInt

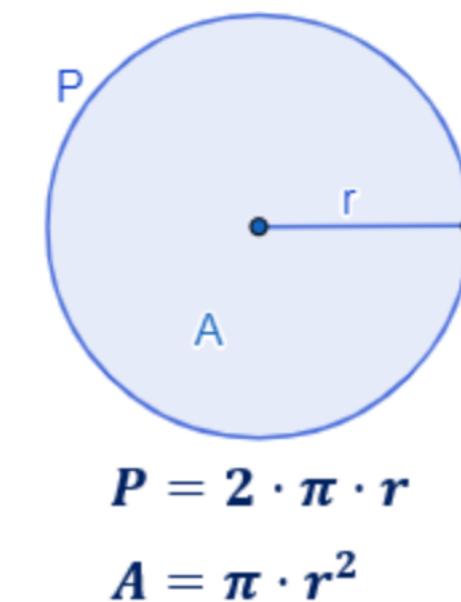
[ - ] Func. Matemáticas  
abs (valor absoluto)  
trunc (valor truncado)  
redon (valor redondeado)  
raiz (raíz cuadrada)  
sen (seno)  
cos (coseno)  
tan (tangente)  
asen (arcoseno)  
acos (arcocoseno)  
atan (arcotangente)  
ln (logaritmo natural)  
exp (func. exponencial)  
azar (numero aleatorio)

[ - ] Func. p/Cadenas  
Longitud  
SubCadena  
Concatenar  
ConvertirANumero  
ConvertirATexto  
Mayusculas  
Minusculas

¿Cómo podemos  
desarrollar nuestras propias  
funciones?

Parámetros

```
resultado = abs(num)
subcad = SubCadena(notas, inicio, 13)
long = Longitud(subcad)
```



# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Definir funciones propias permite organizar mejor el código, haciéndolo más sencillo de seguir
- Además, son útiles para no repetir un mismo conjunto de instrucciones que aparece varias veces en el algoritmo

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Definir funciones propias permite organizar mejor el código, haciéndolo más sencillo de seguir
- Además, son útiles para no repetir un mismo conjunto de instrucciones que aparece varias veces en el algoritmo

```
Algoritmo
inst1
inst2
inst3
inst4
inst5
inst6
inst2
inst3
inst4
inst7
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

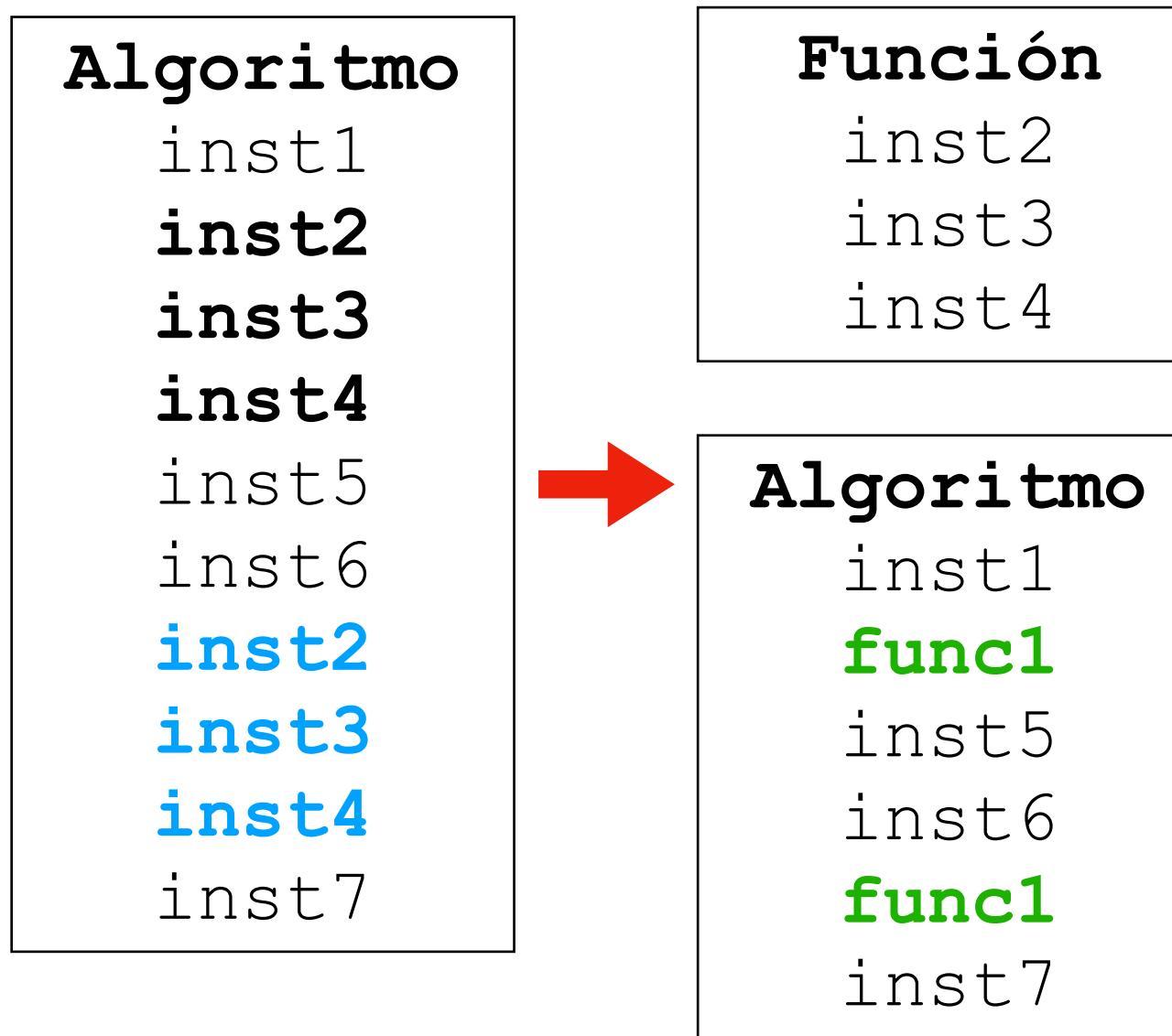
- Definir funciones propias permite organizar mejor el código, haciéndolo más sencillo de seguir
- Además, son útiles para no repetir un mismo conjunto de instrucciones que aparece varias veces en el algoritmo

```
Algoritmo
    inst1
    inst2
    inst3
    inst4
    inst5
    inst6
    inst2
    inst3
    inst4
    inst7
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Definir funciones propias permite organizar mejor el código, haciéndolo más sencillo de seguir
- Además, son útiles para no repetir un mismo conjunto de instrucciones que aparece varias veces en el algoritmo



# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Definir funciones propias permite organizar mejor el código, haciéndolo más sencillo de seguir
- Además, son útiles para no repetir un mismo conjunto de instrucciones que aparece varias veces en el algoritmo

```
Algoritmo
inst1
inst2
inst3
inst4
inst5
inst6
inst2
inst3
inst4
inst7
```

```
Función
inst2
inst3
inst4
```

→

```
Algoritmo
inst1
func1
inst5
inst6
func1
inst7
```

¿Cómo podemos  
desarrollar nuestras propias  
funciones en *PseInt*?

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Se desarrollan siempre antes del algoritmo, para que puedan ser “llamadas” desde el mismo
- Además, pueden necesitar datos con los que “trabajar” (**parámetros**), y pueden devolver un valor (**variable de retorno**)

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Se desarrollan **siempre antes del algoritmo**, para que puedan ser “llamadas” desde el mismo
- Además, pueden necesitar datos con los que “trabajar” (**parámetros**), y pueden devolver un valor (**variable de retorno**)
  - Funciones **sin parámetros y sin variable de retorno** (no devuelven nada)
  - Funciones **con parámetros y sin variable de retorno** (no devuelven nada)
  - Funciones **sin parámetros y con variable de retorno** (devuelven un dato)
  - Funciones **con parámetros y con variable de retorno** (devuelven un dato)

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **SIN** variable de retorno (no devuelven nada)

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN parámetros** y **SIN variable de retorno** (no devuelven nada)

**Sintaxis**

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

```
Funcion NuestraFuncion()
    f_instr1
    f_instr2
    ...
FinFuncion

Algoritmo NuestroAlgoritmo
    a_instr1
    a_instr2
    NuestraFuncion()
    a_instr3
FinAlgoritmo
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

```
Funcion NuestraFuncion()
    f_instr1
    f_instr2
    ...
FinFuncion

Algoritmo NuestroAlgoritmo
    a_instr1
    a_instr2
    NuestraFuncion()
    a_instr3
FinAlgoritmo
```

### Ejemplo

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

```
Funcion NuestraFuncion()
    f_instr1
    f_instr2
    ...
FinFuncion

Algoritmo NuestroAlgoritmo
    a_instr1
    a_instr2
    NuestraFuncion()
    a_instr3
FinAlgoritmo
```

### Ejemplo

```
Funcion NuestraFuncion()
    Escribir "Hola! Pero no se quién eres..."
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto
    Escribir "Dime tu nombre"
    Leer nombre
    NuestraFuncion()
    Escribir "Adios!"
FinAlgoritmo
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

```
Funcion NuestraFuncion()
    f_instr1
    f_instr2
    ...
FinFuncion

Algoritmo NuestroAlgoritmo
    a_instr1
    a_instr2
    NuestraFuncion()
    a_instr3
FinAlgoritmo
```

### Ejemplo

```
Funcion NuestraFuncion()
    Escribir "Hola! Pero no se quién eres..."
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion()

    Escribir "Adios!"
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola! Pero no se quién eres...
Adios!
*** Ejecución Finalizada. ***
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

```
Funcion NuestraFuncion(parametro1, parametro2,...)
    f_instr1
    f_instr2
    ...
FinFuncion

Algoritmo NuestroAlgoritmo
    a_instr1
    a_instr2
    NuestraFuncion(p1, p2,...)
    a_instr3
FinAlgoritmo
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

```
Funcion NuestraFuncion(parametro1, parametro2,...)
    f_instr1
    f_instr2
    ...
FinFuncion

Algoritmo NuestroAlgoritmo
    a_instr1
    a_instr2
    NuestraFuncion(p1, p2,...)
    a_instr3
FinAlgoritmo
```

### Ejemplo

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

```
Funcion NuestraFuncion(parametro1, parametro2,...)
    f_instr1
    f_instr2
    ...
FinFuncion

Algoritmo NuestroAlgoritmo
    a_instr1
    a_instr2
    NuestraFuncion(p1, p2,...)
    a_instr3
FinAlgoritmo
```

### Ejemplo

```
Funcion NuestraFuncion(name)
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto
    Escribir "Dime tu nombre"
    Leer nombre
    NuestraFuncion(nombre)
    Escribir "Adios!"
FinAlgoritmo
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

```
Funcion NuestraFuncion(parametro1, parametro2,...)
    f_instr1
    f_instr2
    ...
FinFuncion

Algoritmo NuestroAlgoritmo
    a_instr1
    a_instr2
    NuestraFuncion(p1, p2,...)
    a_instr3
FinAlgoritmo
```

### Ejemplo

```
Funcion NuestraFuncion(name)
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto
    Escribir "Dime tu nombre"
    Leer nombre
    NuestraFuncion(nombre)
    Escribir "Adios!"
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Salva
Adios!
*** Ejecución Finalizada. ***
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Sintaxis

```
Funcion NuestraFuncion(parametro1, parametro2,...)
  f_instr1
  f_instr2
  ...
FinFuncion

Algoritmo NuestroAlgoritmo
  a_instr1
  a_instr2
  NuestraFuncion(p1, p2,...)
  a_instr3
FinAlgoritmo
```

### Ejemplo

```
Funcion NuestraFuncion(name)
  Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
  Definir nombre Como Texto
  Escribir "Dime tu nombre"
  Leer nombre
  NuestraFuncion(nombre)
  Escribir "Adios!"
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Salva
Adios!
*** Ejecución Finalizada. ***
```

### OJO CON LOS PARÁMETROS!!

Si “pasamos” una variable desde el algoritmo a la función, ésta se copia en otro espacio de memoria, con un nuevo identificador. Por tanto, **si se modifica la variable en la función, no se modifica la original, sino la copia**

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

```
Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

```
Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo
```



Memoria			
nombre			
“Salva”			



```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

```
Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo
```



```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```

Memoria

nombre			
“Salva”			

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

```
→ Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```

Memoria			
nombre "Salva"			
			name "Salva"

### NOTA!!

La copia de la variable “nombre” (el parámetro “name”) es del mismo tipo de dato que la original

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

```
Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo

*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```

Memoria

nombre			
"Salva"			
			name
			"Juan"

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

```
Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo
```

Memoria

nombre "Salva"			
			name "Juan"

```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

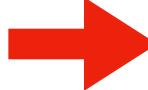
```
Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo
```



Memoria			
nombre			
"Salva"			

```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```

### NOTA!!

Al terminar la función, se borran todas las “copias” de las variables (los parámetros) y las variables que hayan sido definidas dentro de dicha función

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

```
Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo
```



```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```

Memoria

nombre			
“Salva”			

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

```
Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo
```



```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```



Memoria

nombre			
"Salva"			

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON** parámetros y **SIN** variable de retorno (no devuelven nada)

### Ejemplo

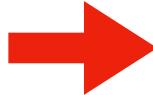
```
Funcion NuestraFuncion(name)
    name = "Juan"
    Escribir "Hola ", name
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir nombre Como Texto

    Escribir "Dime tu nombre"
    Leer nombre

    NuestraFuncion(nombre)

    Escribir "Perdón!"
    Escribir "Tu te llamabas ", nombre
FinAlgoritmo
```



```
*** Ejecución Iniciada. ***
Dime tu nombre
> Salva
Hola Juan
Perdón!
Tu te llamabas Salva
*** Ejecución Finalizada. ***
```

Memoria			

### NOTA!!

Al igual que con las funciones, al terminar el algoritmo, se borran las variables que hayan sido definidas dentro de dicho algoritmo

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

**Sintaxis**

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Sintaxis

```
Funcion variable_retorno = NuestraFuncion()
    Definir variable_retorno Como TIPO_DATO
    f_instr1
    f_instr2
    variable_retorno = ...
FinFuncion

Algoritmo NuestroAlgoritmo
    a_instr1
    a_instr2
    valorDevuelto = NuestraFuncion()
    a_instr3
FinAlgoritmo
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
    Definir num1, num2, resultado Como Entero
    num1 = 0
    num2 = 0

    num1 = azar(10) + 1//1-10
    num2 = azar(10) + 1//1-10
    Escribir "Los números son: ", num1, " y ", num2

    resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
    Definir suma Como Entero
    suma = 0

    Escribir "Voy a calcular una suma aleatoria :)"
    suma = NuestraFuncion()

    Escribir "La suma es: ", suma
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Voy a calcular una suma aleatoria :)
Los números son: 9 y 10
La suma es: 19
*** Ejecución Finalizada. ***
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
  Definir num1, num2, resultado Como Entero
  num1 = 0
  num2 = 0

  num1 = azar(10) + 1//1-10
  num2 = azar(10) + 1//1-10
  Escribir "Los números son: ", num1, " y ", num2

  resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
  Definir suma Como Entero
  suma = 0

  Escribir "Voy a calcular una suma aleatoria :)"
  suma = NuestraFuncion()

  Escribir "La suma es: ", suma
FinAlgoritmo
```

### Memoria

suma			
0			



```
*** Ejecución Iniciada. ***
Voy a calcular una suma aleatoria :(
Los números son: 9 y 10
La suma es: 19
*** Ejecución Finalizada. ***
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
Definir num1, num2, resultado Como Entero
num1 = 0
num2 = 0

num1 = azar(10) + 1//1-10
num2 = azar(10) + 1//1-10
Escribir "Los números son: ", num1, " y ", num2

resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
Definir suma Como Entero
suma = 0

Escribir "Voy a calcular una suma aleatoria :)"
suma = NuestraFuncion()

Escribir "La suma es: ", suma
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Voy a calcular una suma aleatoria :)
Los números son: 9 y 10
La suma es: 19
*** Ejecución Finalizada. ***
```

### Memoria

suma			
0			resultado
		num1	
			num2

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
Definir num1, num2, resultado Como Entero
num1 = 0
num2 = 0

num1 = azar(10) + 1//1-10
num2 = azar(10) + 1//1-10
Escribir "Los números son: ", num1, " y ", num2

resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
Definir suma Como Entero
suma = 0

Escribir "Voy a calcular una suma aleatoria :)"
suma = NuestraFuncion()

Escribir "La suma es: ", suma
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Voy a calcular una suma aleatoria :)
Los números son: 9 y 10
La suma es: 19
*** Ejecución Finalizada. ***
```

### Memoria

suma			
0			
			resultado
		num1	
			num2

### NOTA!!

Las variables definidas en la función no  
pueden ser leídas por el algoritmo

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
  Definir num1, num2, resultado Como Entero
  num1 = 0
  num2 = 0

  num1 = azar(10) + 1//1-10
  num2 = azar(10) + 1//1-10
  Escribir "Los números son: ", num1, " y ", num2

  resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
  Definir suma Como Entero
  suma = 0

  Escribir "Voy a calcular una suma aleatoria :)"
  suma = NuestraFuncion()

  Escribir "La suma es: ", suma
FinAlgoritmo
```

### Memoria

summa			
0			resultado
		num1	
		9	
			num2
			10



```
*** Ejecución Iniciada. ***
Voy a calcular una suma aleatoria :(
Los números son: 9 y 10
La suma es: 19
*** Ejecución Finalizada. ***
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
  Definir num1, num2, resultado Como Entero
  num1 = 0
  num2 = 0

  num1 = azar(10) + 1//1-10
  num2 = azar(10) + 1//1-10
  Escribir "Los números son: ", num1, " y ", num2

  resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
  Definir suma Como Entero
  suma = 0

  Escribir "Voy a calcular una suma aleatoria :)"
  suma = NuestraFuncion()

  Escribir "La suma es: ", suma
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Voy a calcular una suma aleatoria :(
Los números son: 9 y 10
La suma es: 19
*** Ejecución Finalizada. ***
```

### Memoria

suma			
0			
			resultado
			19
		num1	
		9	
			num2
			10

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
→ Funcion resultado = NuestraFuncion()
  Definir num1, num2, resultado Como Entero
  num1 = 0
  num2 = 0

  num1 = azar(10) + 1//1-10
  num2 = azar(10) + 1//1-10
  Escribir "Los números son: ", num1, " y ", num2

  resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
  Definir suma Como Entero
  suma = 0

  Escribir "Voy a calcular una suma aleatoria :)"
  suma = NuestraFuncion()

  Escribir "La suma es: ", suma
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Voy a calcular una suma aleatoria :(
Los números son: 9 y 10
La suma es: 19
*** Ejecución Finalizada. ***
```

### Memoria

suma			
0			resultado
			19

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
  Definir num1, num2, resultado Como Entero
  num1 = 0
  num2 = 0

  num1 = azar(10) + 1//1-10
  num2 = azar(10) + 1//1-10
  Escribir "Los números son: ", num1, " y ", num2

  resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
  Definir suma Como Entero
  suma = 0

  Escribir "Voy a calcular una suma aleatoria :)"
  suma = NuestraFuncion()

  Escribir "La suma es: ", suma
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
Voy a calcular una suma aleatoria :(
Los números son: 9 y 10
La suma es: 19
*** Ejecución Finalizada. ***
```

### Memoria

suma			
19			
			resultado
			19

### OJO!!

El tipo de dato de la variable de retorno tiene que ser el mismo que el de la variable que almacena el valor en el algoritmo

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
  Definir num1, num2, resultado Como Entero
  num1 = 0
  num2 = 0

  num1 = azar(10) + 1//1-10
  num2 = azar(10) + 1//1-10
  Escribir "Los números son: ", num1, " y ", num2

  resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
  Definir suma Como Entero
  suma = 0

  Escribir "Voy a calcular una suma aleatoria :)"
  suma = NuestraFuncion()

  Escribir "La suma es: ", suma
FinAlgoritmo
```

### Memoria

suma			
19			

\*\*\* Ejecución Iniciada. \*\*\*
Voy a calcular una suma aleatoria :-
Los números son: 9 y 10
La suma es: 19
\*\*\* Ejecución Finalizada. \*\*\*

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
  Definir num1, num2, resultado Como Entero
  num1 = 0
  num2 = 0

  num1 = azar(10) + 1//1-10
  num2 = azar(10) + 1//1-10
  Escribir "Los números son: ", num1, " y ", num2

  resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
  Definir suma Como Entero
  suma = 0

  Escribir "Voy a calcular una suma aleatoria :)"
  suma = NuestraFuncion()

  Escribir "La suma es: ", suma
FinAlgoritmo
```

### Memoria

suma			
19			

\*\*\* Ejecución Iniciada. \*\*\*
Voy a calcular una suma aleatoria :)
Los números son: 9 y 10
La suma es: 19
\*\*\* Ejecución Finalizada. \*\*\*

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **SIN** parámetros y **CON** variable de retorno (devuelven un dato)

### Ejemplo

```
Funcion resultado = NuestraFuncion()
  Definir num1, num2, resultado Como Entero
  num1 = 0
  num2 = 0

  num1 = azar(10) + 1//1-10
  num2 = azar(10) + 1//1-10
  Escribir "Los números son: ", num1, " y ", num2

  resultado = num1 + num2
FinFuncion

Algoritmo NuestroAlgoritmo
  Definir suma Como Entero
  suma = 0

  Escribir "Voy a calcular una suma aleatoria :)"
  suma = NuestraFuncion()

  Escribir "La suma es: ", suma
FinAlgoritmo
```

### Memoria




```
*** Ejecución Iniciada. ***
Voy a calcular una suma aleatoria :(
Los números son: 9 y 10
La suma es: 19
*** Ejecución Finalizada. ***
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON parámetros** y **CON variable de retorno** (devuelven un dato)

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON parámetros** y **CON variable de retorno** (devuelven un dato)

### Sintaxis

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON parámetros** y **CON variable de retorno** (devuelven un dato)

### Sintaxis

```
Funcion variable_retorno = NuestraFuncion(parametro1, parametro2, ...)  
    Definir variable_retorno Como TIPO_DATO  
    f_instr1  
    f_instr2  
    variable_retorno = ...  
FinFuncion  
  
Algoritmo NuestroAlgoritmo  
    a_instr1  
    a_instr2  
    valorDevuelto = NuestraFuncion(p1, p2, ...)  
    a_instr3  
FinAlgoritmo
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON parámetros** y **CON variable de retorno** (devuelven un dato)

### Sintaxis

### Ejemplo

```
Funcion variable_retorno = NuestraFuncion(parametro1, parametro2, ...)  
    Definir variable_retorno Como TIPO_DATO  
    f_instr1  
    f_instr2  
    variable_retorno = ...  
FinFuncion  
  
Algoritmo NuestroAlgoritmo  
    a_instr1  
    a_instr2  
    valorDevuelto = NuestraFuncion(p1, p2, ...)  
    a_instr3  
FinAlgoritmo
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON parámetros** y **CON variable de retorno** (devuelven un dato)

### Sintaxis

```
Funcion variable_retorno = NuestraFuncion(parametro1, parametro2, ...)  
    Definir variable_retorno Como TIPO_DATO  
    f_instr1  
    f_instr2  
    variable_retorno = ...  
FinFuncion  
  
Algoritmo NuestroAlgoritmo  
    a_instr1  
    a_instr2  
    valorDevuelto = NuestraFuncion(p1, p2, ...)  
    a_instr3  
FinAlgoritmo
```

### Ejemplo

```
Funcion resultado = AreaCirculo(r)  
    Definir resultado como Real  
    resultado = 0  
  
    Escribir "Vamos a calcular el área del círculo"  
    resultado = PI * r ↑ 2 //Fórmula del área del círculo  
FinFuncion  
  
Algoritmo Areas  
    Definir radio Como Entero  
    Definir area Como Real  
  
    Escribir "Introducir el radio del círculo"  
    Leer radio  
  
    area = AreaCirculo(radio)  
  
    Escribir "El área del círculo es: ", area  
FinAlgoritmo
```

# 1. Nuestra “caja de herramientas” actual (vi)

## 9. Funciones propias

- Funciones **CON parámetros** y **CON variable de retorno** (devuelven un dato)

### Sintaxis

```
Funcion variable_retorno = NuestraFuncion(parametro1, parametro2, ...)  
    Definir variable_retorno Como TIPO_DATO  
    f_instr1  
    f_instr2  
    variable_retorno = ...  
FinFuncion  
  
Algoritmo NuestroAlgoritmo  
    a_instr1  
    a_instr2  
    valorDevuelto = NuestraFuncion(p1, p2, ...)  
    a_instr3  
FinAlgoritmo
```

### Ejemplo

```
Funcion resultado = AreaCirculo(r)  
    Definir resultado como Real  
    resultado = 0  
  
    Escribir "Vamos a calcular el área del círculo"  
    resultado = PI * r ↑ 2 //Fórmula del área del círculo  
FinFuncion  
  
Algoritmo Areas  
    Definir radio Como Entero  
    Definir area Como Real  
  
    Escribir "Introducir el radio del círculo"  
    Leer radio  
  
    area = AreaCirculo(radio)  
  
    Escribir "El área del círculo es: ", area  
FinAlgoritmo
```

```
*** Ejecución Iniciada. ***  
Introducir el radio del círculo  
> 2  
Vamos a calcular el área del círculo  
El área del círculo es: 12.5663706144  
*** Ejecución Finalizada. ***
```

**Y ahora...**  
**vamos a ver unos ejemplos sobre lo que**  
**hemos aprendido en esta clase**  
:-)