

# Tema 6: Funciones

Oscar Perpiñán Lamigueiro - David Álvarez

- 1 Definición de función
- 2 Estructura de un programa
- 3 Ámbito de una variable
- 4 Funciones que llaman a otras funciones
- 5 Funciones en librerías

# ¿Qué es una función?

Una función es un bloque de código que realiza una tarea determinada a partir de unos datos.

## Ventajas

- Permiten **programación estructurada y abstracta**, sin necesidad de conocer el detalle de la implementación de una tarea concreta.
- Mejoran la **legibilidad** del código.
- Facilitan el **mantenimiento** del programa.
- Permiten **reutilizar código** de manera eficiente.

# ¿Cómo se declara una función?

## Prototipo de una función:

- 1 Tipo de valor que devuelve (int, void, ...)
- 2 Nombre de la función (debe ser un identificador válido y **útil**).
- 3 Lista de argumentos que emplea, por tipo y nombre (puede estar vacía).

```
tipo nombre_funcion(tipo1 arg1, tipo2 arg2, ...);
```

## Ejemplos

```
void printHello(int veces);
```

```
float areaTriangulo(float b, float h);
```

# ¿Cómo se define una función?

```
// Definicion de la funcion printHello

// No devuelve nada (void)

// Necesita un argumento llamado veces,
// un entero (int), para funcionar.

void printHello(int veces)
{
    int i;
    for (i = 1; i <= veces; i++)
        printf("Hello World!\n");
}
```

# ¿Cómo se define una función?

```
// Definicion de la funcion areaTriangulo

// Devuelve un real (float)

// Necesita dos argumentos, b y h, reales.

float areaTriangulo(float b, float h)
{
    float area;

    area = b * h / 2.0;

    return area;
}
```

- 1 Definición de función
- 2 Estructura de un programa
- 3 Ámbito de una variable
- 4 Funciones que llaman a otras funciones
- 5 Funciones en librerías

# Estructura de un programa

- *Puede* incluir directivas de inclusión (`include`).
- *Puede* incluir directivas de sustitución (`define`).
- Declaración de funciones (prototipo).
- Todos los programas tienen al menos una función: `main`.
- Definición de las funciones.



# Directivas de inclusión include

Permiten incluir cabeceras (definiciones) procedentes de otros archivos

```
//Librerías del sistema
#include <stdio.h>
#include <math.h>
//Librerías propias del desarrollador
#include "myHeader.h"
```

# Directivas de sustitución `define`

- `define` permite definir símbolos que serán sustituidos por su valor.

```
#include <stdio.h>
//Habitualmente con mayúsculas
//Atención: SIN signo igual NI punto y coma
#define PI 3.141592
#define N 2

int main()
{
    float r1 = 2.0, r2 = 3.0;
    float area[N];
    area[0] = PI * r1 * r1;
    area[1] = PI * r2 * r2;
    printf("Una circunferencia de radio %f, tiene un area de %f\n", r1, area[0]);
    printf("Una circunferencia de radio %f, tiene un area de %f\n", r2, area[1]);

    return 0;
}
```

# Declaración y definición de funciones

```
#include <stdio.h>
// Prototipo de la función (termina en ;)
void printHello(int n);

// Función main
int main() {
    //Uso de la función en main
    printHello(3);
    return 0;
}

// Definición de la función
void printHello(int n)
{
    int i;
    for (i = 1; i <= n; i++)
        printf("Hello World!\n");
}
```

# Declaración y definición de funciones

```
#include <stdio.h>
// Prototipo de la función (termina en ;)
float areaTriangulo(float b, float h);
// Función main
int main(){
    float at;
    //Uso de la función en main
    at = areaTriangulo(1, 2);
    printf("%f", at);
    return 0;
}
// Definición de la función
float areaTriangulo(float b, float h)
{
    float area;
    area = b * h / 2.0;
    return area;
}
```

- 1 Definición de función
- 2 Estructura de un programa
- 3 Ámbito de una variable**
- 4 Funciones que llaman a otras funciones
- 5 Funciones en librerías

# Variables globales

```
#include <stdio.h>

int globalVar = 3; //Variable global

void funcion_ejemplo(void);

int main(){
    printf("main (1):\t globalVar es %d.\n", globalVar);
    funcion_ejemplo();
    globalVar = globalVar*2;
    printf("main (2):\t globalVar es %d.\n", globalVar);
    return 0;
}

void funcion_ejemplo(void){
    globalVar = globalVar + 1;
    printf("funcion:\t globalVar es %d.\n", globalVar);
}
```

# Variables locales

```
#include <stdio.h>

void funcion_ejemplo(void);

int main()
{
    int x = 1; // variable local en main
    printf("main (1):\t x es %d.\n", x);
    funcion_ejemplo();
    printf("main (2):\t x es %d.\n", x);
    return 0;
}

void funcion_ejemplo(void)
{
    int x = 2; // variable local en foo
    printf("funcion:\t x es %d.\n", x);
}
```

- 1 Definición de función
- 2 Estructura de un programa
- 3 Ámbito de una variable
- 4 Funciones que llaman a otras funciones**
- 5 Funciones en librerías



# Ejemplo

```
#include <stdio.h>
#define PI 3.141592

float eleva3(float x);
float volEsfera(float r);

int main(){
    float radio, vol;
    scanf("%f", &radio);
    vol = volEsfera(radio);
    printf("El volumen es %f", vol);
    return 0;
}

float volEsfera(float r){ //Usa eleva3
    return 4.0/3.0 * PI * eleva3(r);
}

float eleva3(float x){
    return x * x * x;
}
```

# Funciones recursivas

```
#include <stdio.h>

int fact(int n);

int main(){
    int x;
    printf("Indica un número:\n");
    scanf("%d", &x);
    printf("El factorial de %d es %d\n", x, fact(x));
    return 0;
}

int fact(int n){
    int res;
    if (n > 1) // Incluye llamada a si misma
        res = n * fact(n - 1);
    else
        res = 1;
    return res;
}
```

- 1 Definición de función
- 2 Estructura de un programa
- 3 Ámbito de una variable
- 4 Funciones que llaman a otras funciones
- 5 Funciones en librerías

# Motivación y uso

## Motivación

Para poder reutilizar las funciones definidas es conveniente alojarlas en un fichero (o colección de ficheros) que puedan ser incluidos en otros proyectos.

## Uso

- Debe existir un (o varios) fichero(s) .h (cabecera) y un fichero .c (código fuente, implementación de las funciones).
- Se debe usar `#include "nombre_lib.h"` al comienzo del programa.
- Hay que compilar conjuntamente (*en un proyecto*).

# Ejemplo (1)

## Fichero myLib.h (cabecera)

```
#define PI 3.141592

float eleva3(float x);
float volEsfera(float r);
```

## Fichero myLib.c (código fuente)

```
#include "myLib.h"

float volEsfera(float r){
    return 4.0/3.0 * PI * eleva3(r);
}

float eleva3(float x){
    return x * x * x;
}
```

## Ejemplo (2)

### Programa principal

```
#include <stdio.h>
// Directiva para incluir la librería local
#include "myLib.h"

int main()
{
    float radio, vol;
    scanf("%f", &radio);
    vol = volEsfera(radio);
    printf("El volumen es %f", vol);
    return 0;
}
```