

# PROGRAMACIÓ 1

Grado en Ingeniería Informática e I2ADE

## Tema 5 Recursividad



Dept. de Ciència de la Computació i Intel·ligència *a*rtificial  
Dpto. de Ciencia de la Computación e Inteligencia *a*rtificial



Universitat d'Alacant  
Universidad de Alicante

# Índice

2

1. Concepto de recursividad
2. Esquema básico de un módulo recursivo
3. Ejemplos de recursividad
4. Codificación en C
5. Características de la recursividad
6. Traza de un módulo recursivo
7. Ejercicios

# 1. Concepto de recursividad

3

- Un módulo es recursivo cuando entre la lista de instrucciones que lo forman, se encuentra una **llamada a sí mismo**, directa o indirectamente.
- Hay muchas funciones matemáticas que se definen de forma natural de manera recursiva. Por ejemplo:
  - Factorial de un número  $n$ : El factorial de un número  $n$  es el número  $n$  multiplicado por el factorial de  $n-1$ .
$$\text{factorial}(n) = n * \text{factorial}(n-1)$$
  - Potencia de dos números:  $x^n = x * x^{n-1}$

## 2. Esquema básico de un módulo recursivo

4

### ■ Uno o más **casos base**

- No hay llamadas recursivas en ellos. Especifican la “condición de terminación” o “condición de parada” de la recursión.

### ■ Uno o más **casos generales o recursivos**

- Incluye una o más llamadas al propio módulo. Estas llamadas recursivas deben resolver versiones “más simples” del problema a resolver por el módulo. Es decir, se trata de un proceso en el que cada llamada recursiva al propio módulo va recibiendo una versión más simple del problema, hasta llegar al caso base.

### 3. Ejemplos de recursividad

5

#### Cálculo del factorial de un número

`factorial(n) = n * factorial(n-1)`

`factorial(3) = 3 * factorial(2)`



`= 2 * factorial(1)`



`= 1 * factorial(0)`



`= 0 * ...`

¿Cuándo acaba?

### 3. Ejemplos de recursividad

6

#### Cálculo del factorial de un número

Hay que añadir el **caso base** para parar la recursividad

Si **n es igual a 0** Entonces

factorial = 1

Si no

factorial = n \* factorial de (n-1)

Llamada recursiva al propio módulo (**caso recursivo**)

### 3. Ejemplos de recursividad

7

#### Cálculo del factorial de un número

Si (n es 0) Entonces

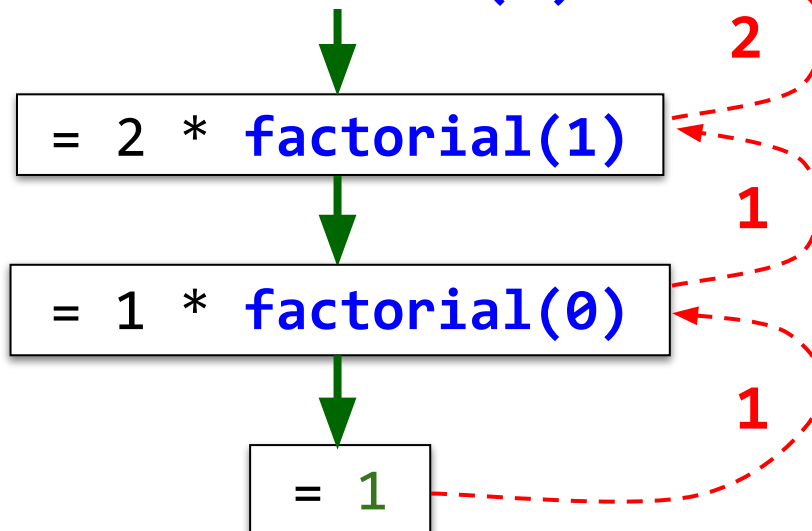
factorial = 1

Si no

factorial = n \* factorial(n-1)

factorial(3) = 3 \* factorial(2)

↓  
6



# 4. Codificación en C

8

```
#include<stdio.h>
```

```
// Declaración de módulos
```

```
int factorial( int n );
```

```
int main() {
```

```
    int num;
```

```
    printf("Introduce un número: ");
```

```
    scanf("%d", &num);
```

```
    printf("El factorial de %d es: %d\n", num, factorial( num ) );
```

```
    return 0;
```

```
}
```

```
// Definición de módulos
```

```
int factorial( int n ) {
```

```
    int res;
```

```
    if ( n == 0 ) // caso base
```

```
        res = 1;
```

```
    else // caso recursivo
```

```
        res = n * factorial( n - 1 );
```

```
    return res;
```

```
}
```




# 5. Características de la recursividad

9

- Idónea para la **resolución de aquellos problemas** que pueden definirse **de modo natural** en términos recursivos.
- **Tiene su equivalente iterativo.**
- **Necesitan mayor cantidad de memoria** para su ejecución.
- **Son más lentos** en su ejecución.

## Importante:

Hay que evitar crear una recursividad infinita que haría que el programa no parara.



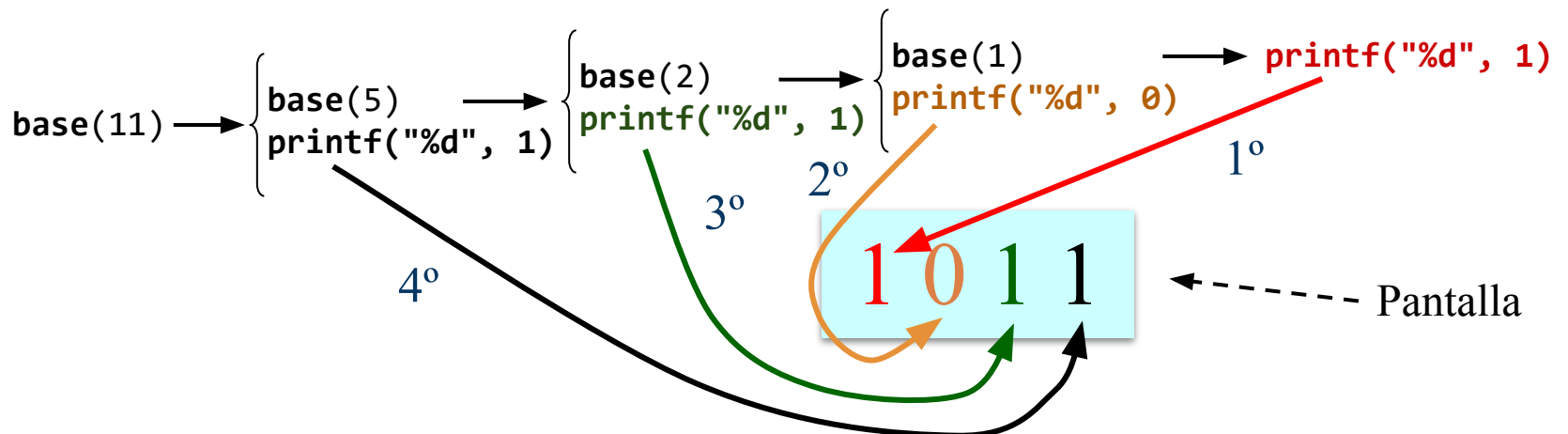
```
void escribe(int n){  
    escribe(n / 10);  
    printf("%d\n", n % 10 );  
}
```

# 6. Traza de un módulo recursivo

10

```
void base(int n){  
    if (n < 2) // caso base  
        printf("%d", n);  
    else { // caso recursivo  
        base(n / 2);  
        printf("%d", n % 2);  
    }  
}
```

```
int main() {  
    ...  
    base(11);  
    ...  
    return 0;  
}
```



## 6. Traza de un módulo recursivo

11

### Ejemplo:

Dado el siguiente módulo:

```
void recursivo (int num){  
    if (num != 0){ // caso recursivo  
        recursivo(num / 2);  
        printf( num % 2 );  
    }  
}
```

1. ¿Cuál es la salida que se obtiene si se le llama de la siguiente forma:  
`recursivo(16)`?  
A) 00001                      B) 11111  
C) 10000                      D) 00100  
E) ninguna de las anteriores
2. ¿Cuál es el caso base?

## 6. Traza de un módulo recursivo

12

### ¿Qué hace este código?

```
int main(){
    char letra;

    printf("Introduce una frase terminada en punto: ");
    scanf("%c", &letra);

    modulo(letra);

    printf("%c\n", letra);
    return 0;
}
```

```
void modulo(char l){
    if (l == '.') // caso base
        printf("\n");
    else { // caso recursivo
        scanf("%c", &l);
        modulo(l);
        printf("%c", l);
    }
}
```

# 7. Ejercicios

13

1. Diseñar un módulo recursivo que para un número natural  $n$  muestre por pantalla la serie creciente de números naturales del 1 al  $n$ , es decir, 1 2 3...  $n$ .
2. Diseñar un módulo recursivo que para un número natural  $n$  devuelva la suma de los cuadrados de los números del 1 hasta el  $n$ . Por ejemplo, para  $n=4$ , el módulo debe devolver 30 ya que  $1^2 + 2^2 + 3^2 + 4^2 = 30$ .
3. Diseñar un módulo que, dado un número natural, muestre por pantalla el número formado por los mismos dígitos en sentido contrario. Por ejemplo: para el número 2089 debe mostrar 9802.
4. Diseñar un módulo que reciba un número en sistema decimal y muestre en pantalla su equivalente en binario. Por ejemplo, para el número 12, debe mostrar en pantalla 1100.
5. Implementa una función recursiva que devuelva el número de dígitos impares de un número. Ejemplo:  $\text{rec}(321)=2$ ,  $\text{rec}(28)=0$ .