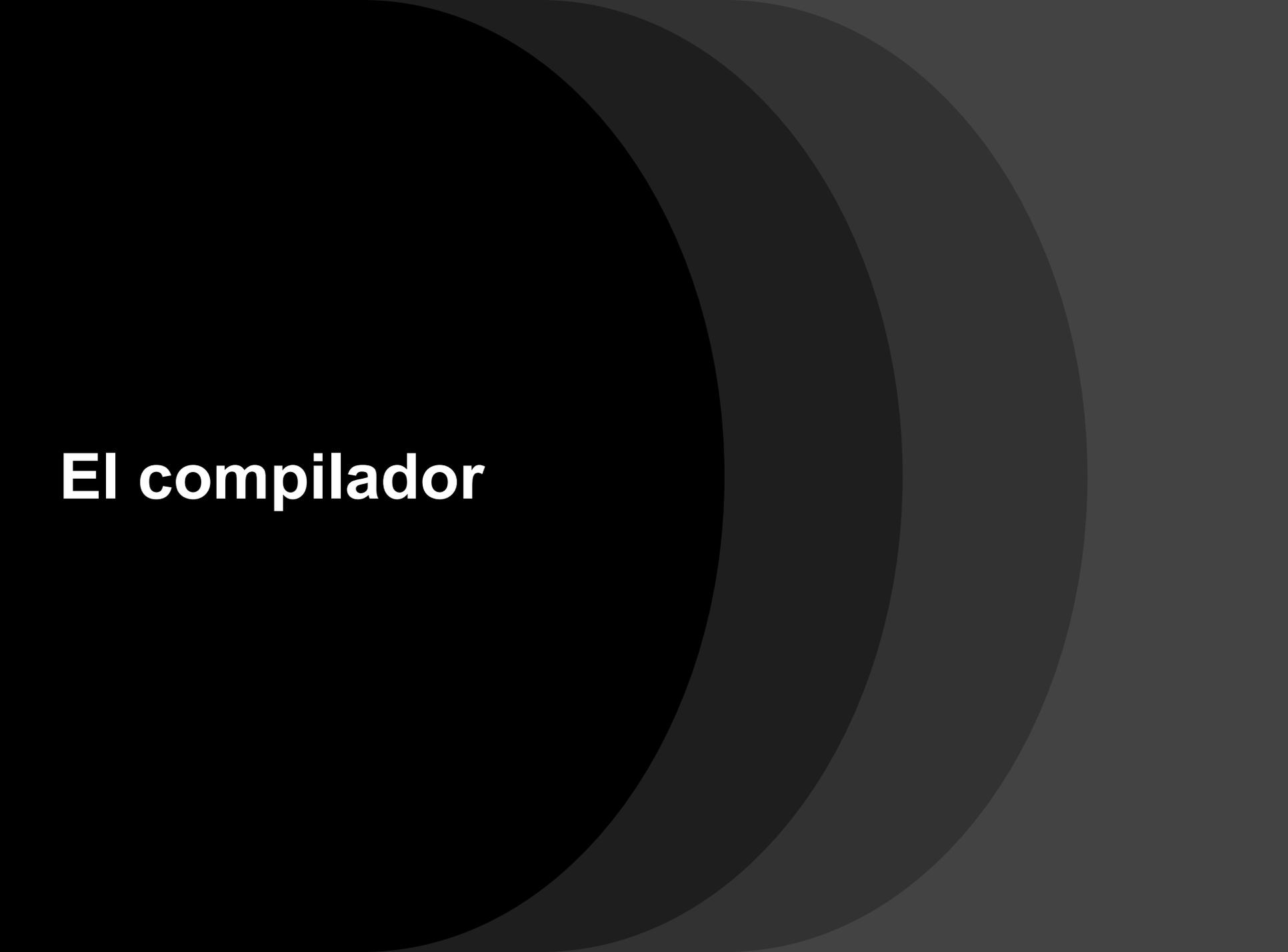


# Sesión 2

Aprende C desde cero

# El compilador



# Compilador

¡ C es un lenguaje compilado no interpretado !

```
compilador [flags] fichero.c -o nombre [bibliotecas];
```

Dónde:

- **compilador** traduce el código en C a código máquina.
  - gcc, icc, clang...
- **flags** son parámetros opcionales:
  - optimización, warnings...
- **fichero.c** contiene el código escrito por nosotros.
- **-o** Opción para que el compilador genere un binario ejecutable.
- **nombre** es fichero a ejecutar.
- **bibliotecas** es una lista de bibliotecas utilizadas en el código.

```
#include <stdio.h>

int main(){

    int a;
    a = 2;
    printf("El valor de a es %d\n",
a);

    return 0;
}
```

fichero.c



```
gcc fichero.c -o fichero
```

Compilamos con gcc



```
El valor de a es 2
```

Obtenemos la salida



```
./fichero
```

Ejecutamos

## Nosotros

```
#include <stdio.h>

int main(){

    int a;
    a = 2;
    printf("El valor de a es %d\n",
a);

    return 0;
}
```

fichero.c

```
El valor de a es 2
```

Obtenemos la salida

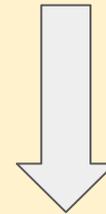
## Jupyter

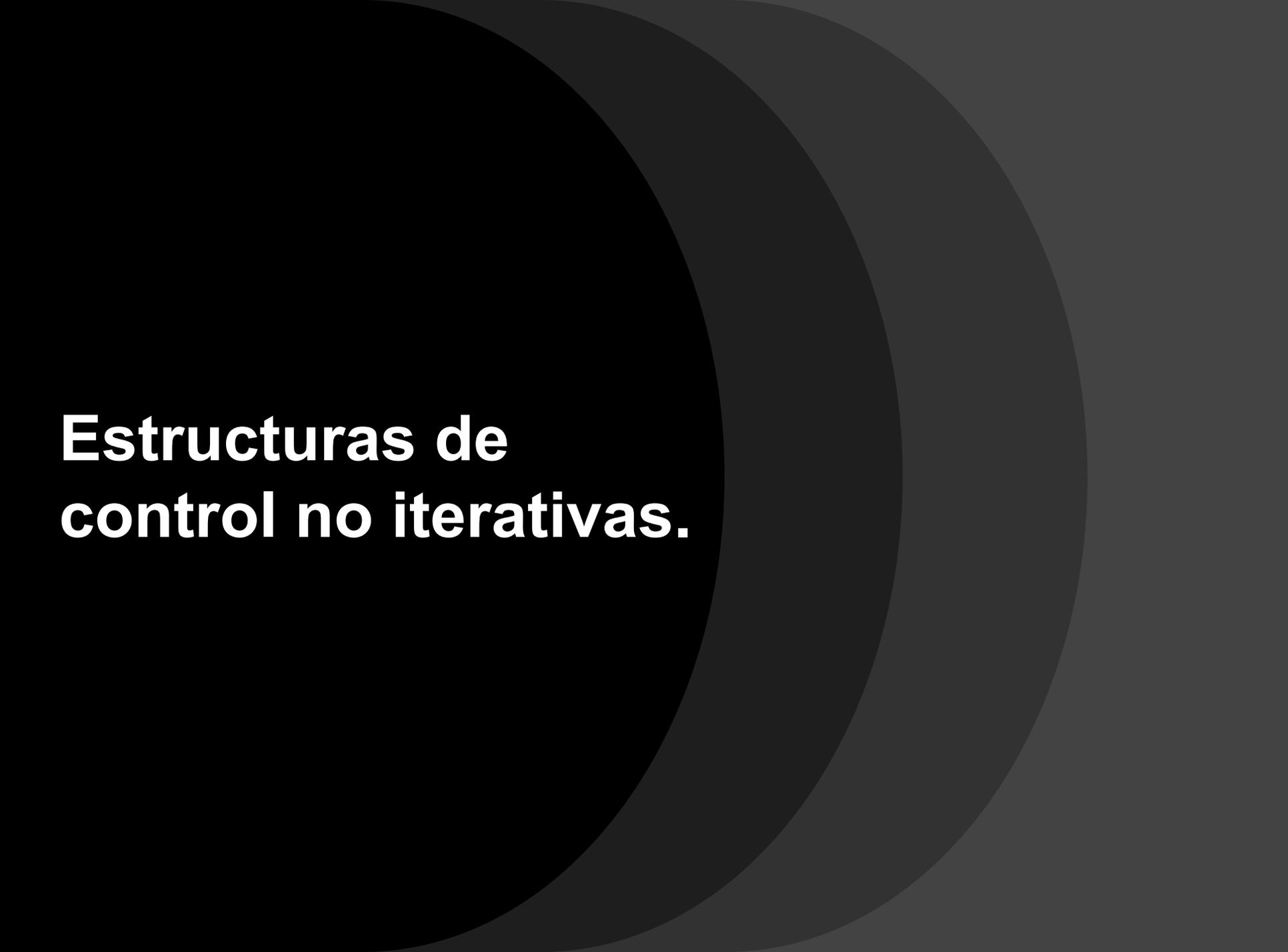
```
gcc fichero.c -o fichero
```

Compilamos con gcc

```
./fichero
```

Ejecutamos





**Estructuras de  
control no iterativas.**

# Ejecución Completa

```
/* Bibliotecas que utiliza el programa */
#include <stdio.h>

/* Punto de inicio del programa */
int main(void) {

    /* Declaraciones e inicializaciones */
    ...
    /* Cuerpo del programa */
    instrucción 1;
    instrucción 2;
    instrucción 3;
    instrucción 4;
    instrucción 5;
    ...
    instrucción n-1;
    instrucción n;

    /* Acaba la ejecución del programa */
    return 0;
}
```

# Ejecución condicional

```
/* Bibliotecas que utiliza el programa */
#include <stdio.h>

/* Punto de inicio del programa */
int main(void) {

    /* Declaraciones e inicializaciones */
    ...
    /* Cuerpo del programa */
    instrucción 1;
    instrucción 2;
    instrucción 3 o instrucción 4;
    instrucción 5;
    ...
    instrucción n-1;
    instrucción n;

    /* Acaba la ejecución del programa */
    return 0;
}
```

# If

Permite la ejecución de un código dependiendo de su condición:

```
if(condición) {  
    código;  
}
```

Dónde:

- **if** es la palabra clave de esta estructura.
- **condición** es una comparación entre valores que se resolverá como verdadero o falso.
- **código** es el bloque de instrucciones que se ejecutarán si la condición es verdadera.

Ejemplo:

```
int main() {  
    int dividendo = 12;  
    int divisor = 0;  
    int resultado;  
  
    resultado = dividendo/divisor;  
    return 0;  
}
```

# If

Permite la ejecución de un código dependiendo de su condición:

```
if(condición) {  
    código;  
}
```

Dónde:

- **if** es la palabra clave de esta estructura.
- **condición** es una comparación entre valores que se resolverá como verdadero o falso.
- **código** es el bloque de instrucciones que se ejecutarán si la condición es verdadera.

Ejemplo:

```
int main() {  
    int dividendo = 12;  
    int divisor = 0;  
    int resultado;  
    if (divisor != 0) {  
        resultado = dividendo/divisor;  
    }  
    return 0;  
}
```

# Comparadores de condición

Representar	Marca de formato
Mayor	>
Menor	<
Igual	==
Distinto	!=
Mayor o igual	>=
Menor o igual	<=
Negación	!

Es importante distinguir entre el símbolo de asignación (=) y el de igual (==).

# Verdadero y falso

Una condición se evalúa a **falso** cuando su resultado sea **cero**.

Es decir, las condiciones evaluadas a un **valor distinto de cero** (sea positivo o negativo) serán **verdaderas**.

Por ejemplo:

Condición	Evaluación
$2 + 2 == 4$	Verdadero
$2 == 4$	Falso
$0 == 0$	Verdadero
0	Falso
-3	Verdadero

# Concatenar condiciones

Representar	Marca de formato
AND lógica	<b>&amp;&amp;</b>
OR lógica	<b>  </b>

En el siguiente ejemplo el código solamente se ejecutará si la condición1 es cierta y la condición2 también

```
if (condición1 && condición2) {  
    código;  
}
```

Por el contrario, en el siguiente ejemplo, el código se ejecutará si una de las dos condiciones es cierta

```
if (condición1 || condición2) {  
    código;  
}
```

# If - else

Forma completa de esta estructura de control

```
if(condición) {  
    código1;  
}  
else{  
    código2;  
}
```

Dónde:

- **if** es la palabra clave de esta estructura.
- **condición** es una comparación entre valores que se resolverá como verdadero o falso.
- **código1** es el bloque de instrucciones que se ejecutarán si la condición es verdadera.
- **else** es otra palabra clave y su contenido se ejecutará si la comparación inicial ha sido falsa.
- **código2** es el bloque de instrucciones que se ejecutarán si la condición es falsa.

Además, dentro de *código1* y *código2* puede haber una o más estructura `if-else`, lo que se conoce como ifs anidados.

# If - else

```
/* Bibliotecas que utiliza el programa */
#include <stdio.h>

/* Punto de inicio del programa */
int main(void) {
/* Declaraciones e inicializaciones */

    int dividendo = 12;
    int divisor = 0;
    int resultado;

/* Cuerpo del programa */
    if (divisor != 0){
        resultado = dividendo/divisor;
    }
    else{
        printf("El divisor no puede ser 0\n");
    }
/* Acaba la ejecución del programa */
    return 0;
}
```

# Switch

Es otra estructura de control no iterativa

```
switch (opción) {  
    case op1:  
        código1;  
        break;  
    case op2:  
        código2;  
        break;  
    [case default:  
        código;  
    ]  
}
```

Dónde:

- **switch** es la palabra clave de esta estructura.
- **opción** es un valor constante para elegir la porción de código a ejecutar.
- **case** es la palabra clave que indica el inicio de una opción.
- **opX** es otro valor constante que se comparará con el valor **opción**.
- **break** palabra clave que finaliza la porción de código y la estructura.
- **default** es la palabra clave para indicar el código a ejecutar si **opción** no es igual a ninguna **opX**



**Ejercicio**

# Ejercicio: Calculadora 2.0

Desarrolla un programa en C que dados dos números calcule su suma, resta, producto o división dependiendo la operación elegida.

- Los dos números deberán almacenarse en variables.
- El resultado de la operación deberá guardarse en una variable.
- La operación se almacenará en una variable entera utilizando: 0 para la suma, 1 para la resta, 2 para la multiplicación y 3 para la división.
- Habrá que tener en cuenta si al elegir la división, el divisor es 0.

Ejemplo:

Dados los números 3 y 4 y elegida la división, la salida del programa debería ser:

```
Bienvenido/a a "Calcooladora v2.0".  
  
Primera cifra: 3.00  
Segunda cifra: 4.00  
Operación: División  
  
...calculando resultado...  
  
Resultado: 0.75  
  
¡Adiós!
```

## Recuerda que puedes:

Interactuar con otros alumnos en el foro del curso:

<http://mooc.uji.es/mod/forum/view.php?id=1654>

Acceder a todos los materiales en el repositorio:

<https://siserte@bitbucket.org/siserte/repositoriocursoc.git>

Contactar con los profesores:

- [adcastel@uji.es](mailto:adcastel@uji.es)
- [siserte@uji.es](mailto:siserte@uji.es)