Escuela Politécnica Superior

# SEMINARIO-TALLER DE SOFTWARE

# (STI-S)

UNIDAD 3. Programación en C
(Sentencias de Control)

Sesión 2

# Índice

1. **Sentencias de control.**
2. **Sentencia *if-else***
3. **Sentencia *switch***
4. **Sentencia *for***
5. **Sentencia *while* y *do-while***

# Índice

# Sentencias de control

- Por defecto el código de un programa se ejecuta de forma secuencial.
- Frecuentemente los programadores necesitan que una parte especifica de código se ejecute *varias veces*, o *mientras se cumpla una condición*, etc…
- Las **sentencias de control** permiten modificar el orden de ejecución creando bucles o condiciones sobre ciertas partes del código.
- Las sentencias de control son: **if-else**, **switch**, **for**, **while** y **do-while**.
- C permite utilizar la sentencia ***goto*** pero …. **No debe utilizarse en NINGUNA circunstancia!!**

# Índice

# *Sentencia If-else*

- **If-else** se usa cuando la ejecución de un bloque de código depende del resultado de una condición (sentencia lógica).
- Su estructura es la siguiente:

```
if( <condicion> ) {
    …
    …
}
else{
    …
}
```

# *Sentencia If-else*

- La parte del **else** es opcional:

```
if( <condicion> ) {
    …
    …
}
```

- También se pueden crear sentencias **else-if**:

```
if( <condicion 1> ) {
    …
    …
}
else if( <condicion 1> ) {
    …
    …
}
```

# Índice

# *Sentencia Switch*

- La sentencia **switch** permite ejecutar diferentes bloques de código dependiendo del valor de una variable.
- Su estructura es la siguiente:

```
switch (variable1){
case <valor1>:
    …
    break;
case <valor2>:
    …
    break;
default:
    …
}
```

- La sentencia ***break*** se utiliza para evitar el ejecución del resto de bloques.
- El caso ***default*** representa todos los posibles valores que puede tomar la variable y no están especificados en los casos previos.

# *Sentencia Switch*

- Varios valores para ejecutar un mismo bloque de código:

```
switch (variable1){
    case <valor1>:
    case <valor2>:
    case <valorj>:
    …
    break;
    case <valorj+1>:
    case <valorj+2>:
    case <valor>:
    …
    break;
default:
    …
}
```

# Índice

# *Sentencia For*

- *Se utiliza para repetir un bloque de código varias veces.*
- Están compuestos por tres sentencias:
  - ***Sentencia inicial***: sentencia de inicialización que solo se ejecuta la primera vez.
  - ***Sentencia de control***: condición que controla cuantas veces se ejecuta el bucle. Mientras que la condición sea verdadera se repetirá el bloque de código indicado.
  - ***Sentencia de modificación***: sentencia que se ejecuta cada vez que se cumple la condición y que sirve para cambiar el valor de las variables involucradas en las sentencias anteriores.

```
for(<inicial>; <control>; <modificacion>){
    …
}
```

# *Sentencia For*

- *Ejemplo:*

```
int var1;
for(var1=1; var1<10; var1++){
   printf("%d\n",var1);
}
```

"Desde var1 = 1, se incrementa var1 en 1 cada vez, y se ejecuta:

```
printf("%d\n",var1);
```

mientras que var1 sea menor que 10"

# Índice

# *Sentencias While y Do-While*

- Necesita una sentencia lógica y el bloque de código se repetirá hasta que esta sentencia tome el valor de falso.

```
while(<sentencia logica>){
    …
}
```

- La estructura de la sentencia do-while es:

```
do{
    …
}
while(<sentencia logica>);
```