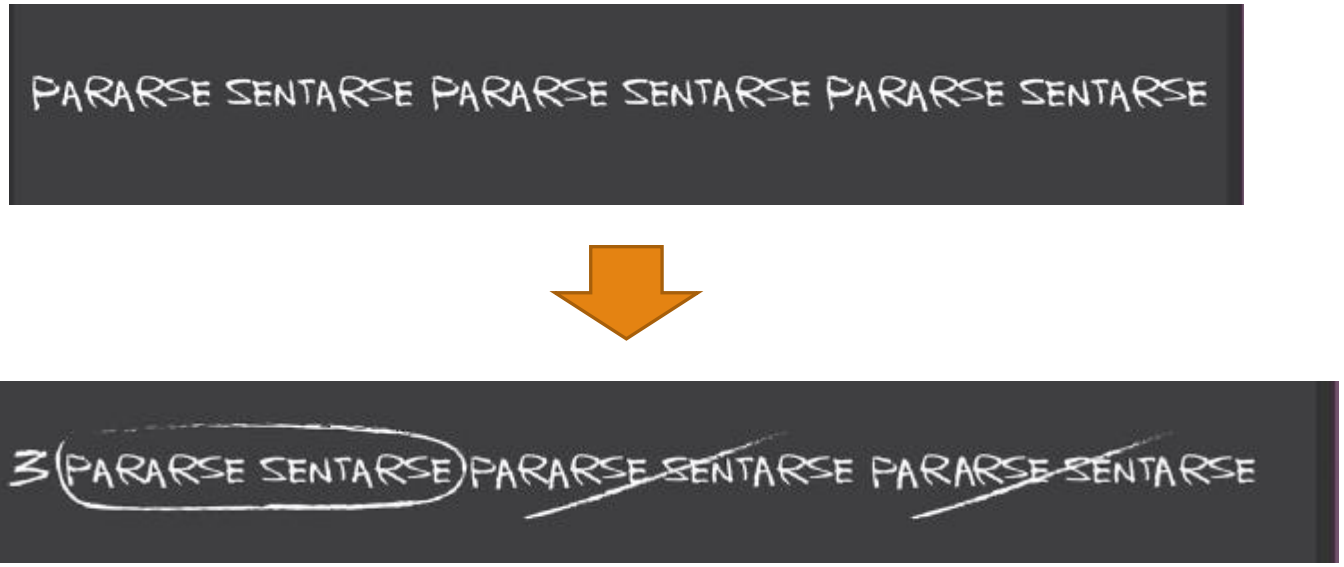
PROGRAMACIÓN ESTRUCTURADA

UNIDAD 3

Estructura de repetición

Estructuras de repetición (ciclos)

- Necesidad de repetir varias veces una serie de instrucciones.
- Los lenguajes de programación ofrecen distintas construcciones que nos permiten expresar repeticiones sin necesidad de reiterar comandos en forma explícita.



Estructuras de repetición (ciclos)

- Existen dos tipos de ciclo de repetición
  - EXACTOS
    - Cuando se conoce de antemano la cantidad de veces que se necesita repetir un bloque de código
  - INEXACTOS
    - Cuando no se conoce de antemano la cantidad de iteraciones necesarias.

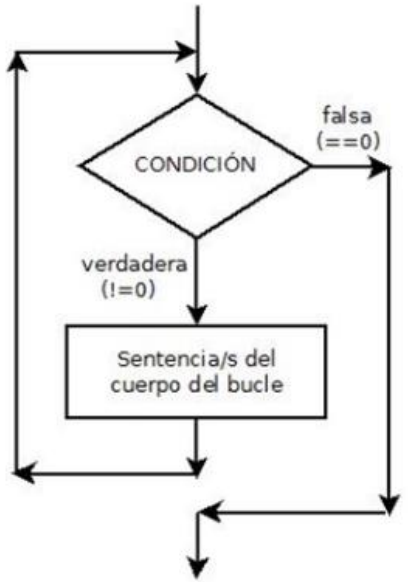
Estructuras de repetición (ciclos)

- **CICLO INEXACTO (WHILE-DO)**
  - En el ciclo no exacto WHILE se ingresa un dato, se controla el valor y si se cumple se ingresa al programa, al final se vuelve a ingresar otro dato y se vuelve a controlar, cuando no se cumpla la condición nos saca del proceso.

Bucle *while* en C:

```
while (condición)
    {
        sentencia_1;
            . . .
        sentencia_n;
    }
```

Bucle *while* (mientras):



Estructuras de repetición (ciclos)

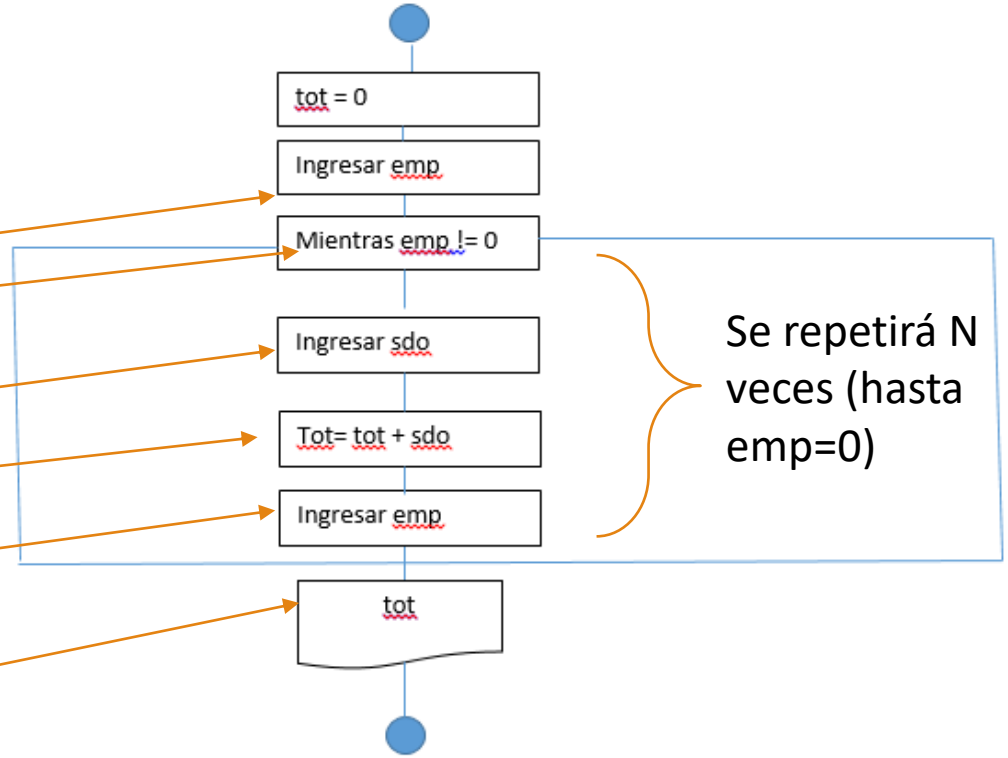
- **CICLO INEXACTO (WHILE-DO)**

```
int main()
{
int emp;
float  sdo,tot;
system("cls");
tot=0;
printf("ingrese el nro de empleado ");
scanf("%d",&emp);

while(emp!=0)
    {
    printf("ingrese el sueldo ");
    scanf("%f",&sdo);

    tot =tot+sdo;
    printf("ingrese el nro de empleado ");
    scanf("%d",&emp);

    }
printf("el total de sueldos es %4.1f \n",tot);
system("pause");
}
```

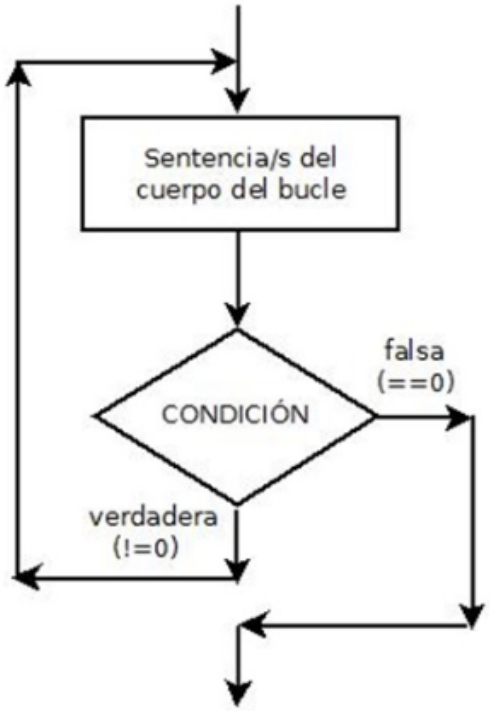


Estructuras de repetición (ciclos)

- **CICLO INEXACTO (DO-WHILE)**
  - En ciclo no exacto DO-WHILE es similar al anterior pero el control se realiza una vez ejecutado al menos una vez el proceso.

Bucle *do while* en C:

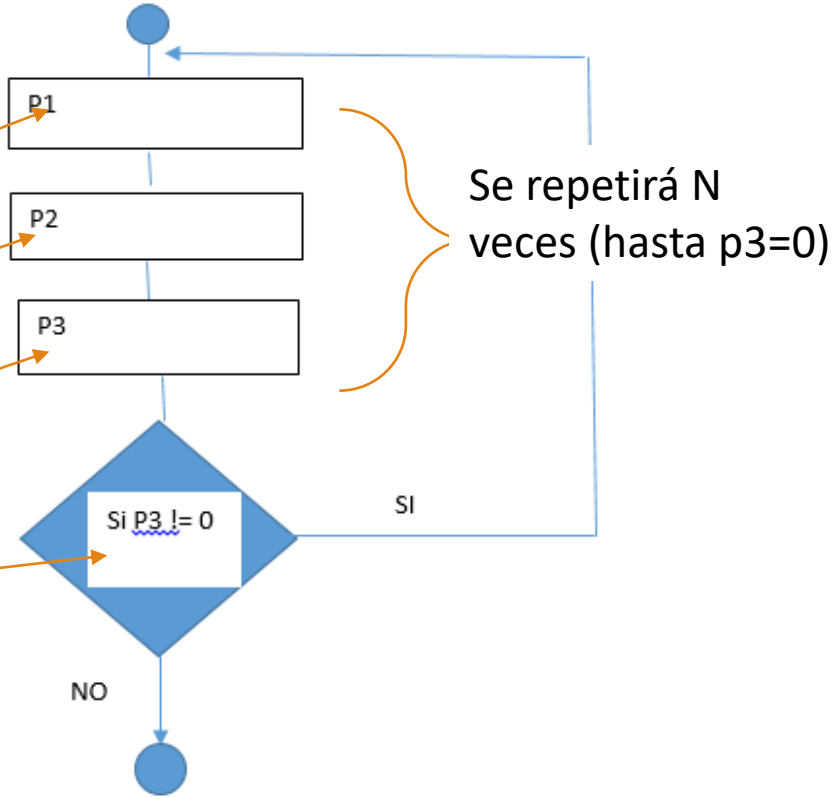
```
do
    {
        sentencia_1;
            . . .
        sentencia_n;
    }
while (condición);
```



Estructuras de repetición (ciclos)

- **CICLO INEXACTO (DO-WHILE)**

```
4
5  int main(int argc, char** argv) {
6
7
8      int p1,p2,p3;
9
10     do{
11
12         printf("Ingrese P1: ");
13         scanf("%d",&p1);
14
15         printf("Ingrese P2: ");
16         scanf("%d",&p2);
17
18         printf("Ingrese P3: ");
19         scanf("%d",&p3);
20
21     } while(p3!=0);
22
23
24     return 0;
25 }
```



Estructuras de repetición (ciclos)

- **INSTRUCCIÓN BREAK**
  - La instrucción break se utiliza, además de para salir de una instrucción switch, para terminar anticipadamente la ejecución de un bucle for, while o do-while. Hace que el bucle desde el que se invoca termine inmediatamente.
  - Si es invocado dentro de un bucle anidado, termina solamente el bucle desde el que ha sido invocado, pero no el bucle externo a éste.
  - Proporciona una forma conveniente de terminar un bucle cuando se detecta un error o alguna condición irregular.
- **INSTRUCCIÓN CONTINUE**
  - Es muy similar a la instrucción *break* con la salvedad de que *continue* salta el resto del código asociado al cuerpo del bucle, pero retorna a la condición del bucle, y si esta es cierta, ejecuta la siguiente iteración. Recordemos que *break* rompe el bucle definitivamente

Máximos y Mínimos

Máximos y mínimos

- **Se utilizan variables auxiliares para obtener máximos y mínimos.**
- **Representan el valor máximo y el valor mínimo asignados a una variable durante la ejecución de un programa**
- **Por ejemplo: Se ingresan N temperaturas y se desea saber cual es la máxim y la mínima**
  - **Consideraciones:**
    - **Siempre el primer valor ingresado es el máximo y el mínimo a la vez y debe almacenarse ese valor en variables auxiliares**
    - **Luego, se debe comparar el valor próximo ingresado con el valor de las variables auxiliares y en caso de ser mayor o menor, actualizar la variable auxiliar con el nuevo máximo o mínimo.**