**‌Estructuras de control de flujo de datos**

Las estructuras de control de flujo, son instrucciones que nos permiten evaluar si se puede cumplir una condición o no, incluso nos puede ayudar a evaluarla **n** cantidad de veces.

* **Condicional (if):**
* **Simple**
* **Compuesto**
* **Anidado**

**CONDICIONAL (IF):**

Los condicionales nos permiten evaluar si una condición cumple o no con lo que estemos evaluando. Su sintaxis es muy sencilla, podemos evaluar si la condición es verdadera o falsa. Incluso añadir una condición intermedia en el caso de que no se cumpla la primera condición y se deban evaluar más.

A continuación, un de estructura condicional if/elif/else completo:

**print** ('El número ingresado es 1.**\n**')

**else**:

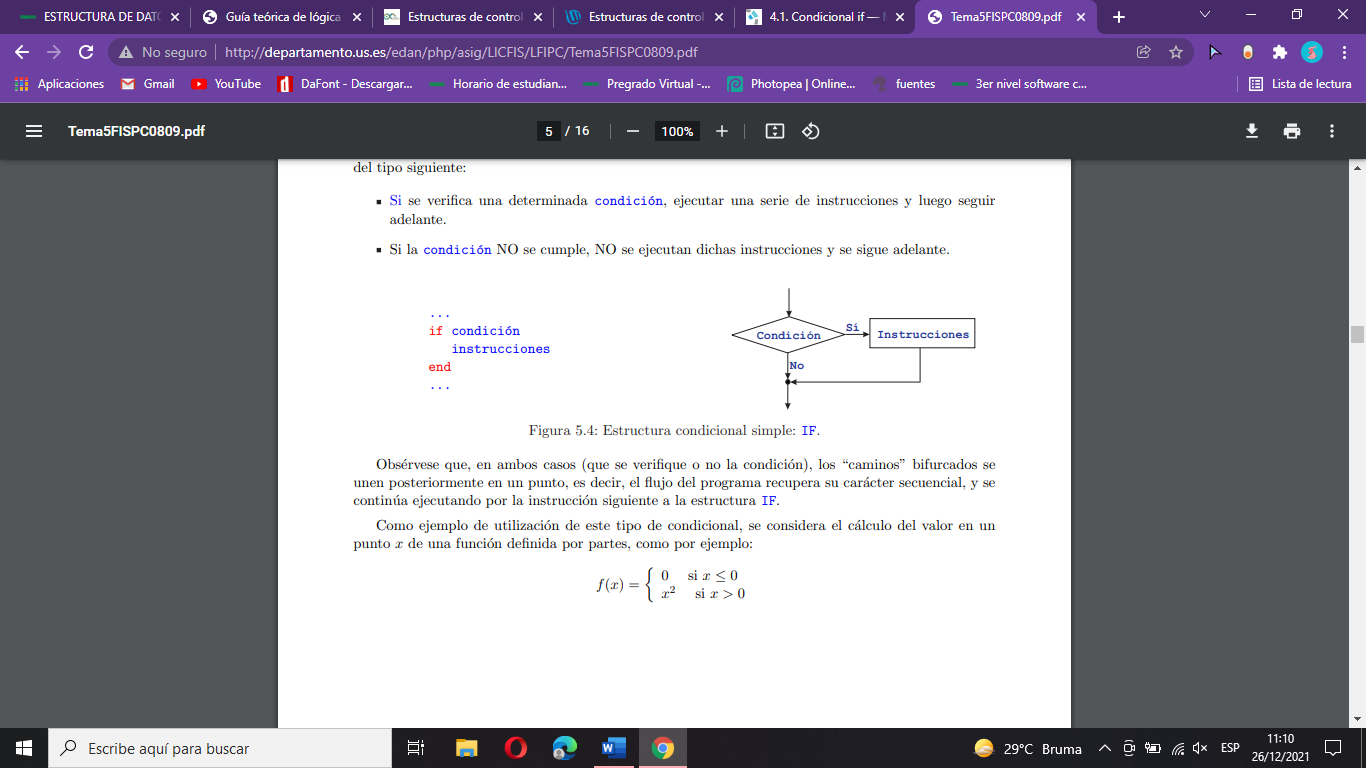
**print** ('El número ingresado es mayor que uno.**\n**')

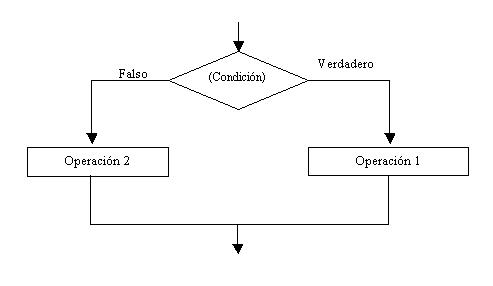
**Sentencia if.-**La sentencia if EXPRESION, significa, **Si** se cumple la [expresión condicional](https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion4/condicional_if.html#python-expresiones-condicional) se ejecuta el bloque de sentencias seguidas.

## **Sentencia elif.-**La sentencia elif EXPRESION, significa, **De lo contrario Si** se cumple la [expresión condicional](https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion4/condicional_if.html#python-expresiones-condicional) se ejecuta el bloque de sentencias seguidas.

**Sentencia els**e.-La sentencia else, significa, **De lo contrario** se cumple sin evaluar ninguna [expresión condicional](https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion4/condicional_if.html#python-expresiones-condicional) y ejecuta el bloque de sentencias seguidas.

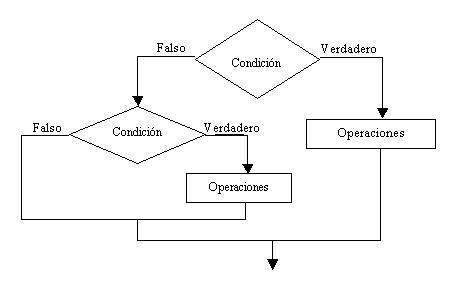
**SIMPLE:**

Las estructuras selectivas simples o condicionales están compuesta únicamente de una sola condición si es verdadera ejecutara la acción o acciones si la condición es falsa no hará nada.

Cuando se presenta la elección tenemos la opción de realizar una actividad u otra. Es decir tenemos actividades por el verdadero y por el falso de la condición. Lo más importante que hay que tener en cuenta que se realizan las actividades de la rama del verdadero o las del falso, NUNCA se realizan las actividades de las dos ramas.

**ANIDADO:**

Decimos que una estructura condicional es anidada cuando por la rama del verdadero o el falso de una estructura condicional hay otra estructura condicional.



**Estructuras de control de flujo de datos iterativas**

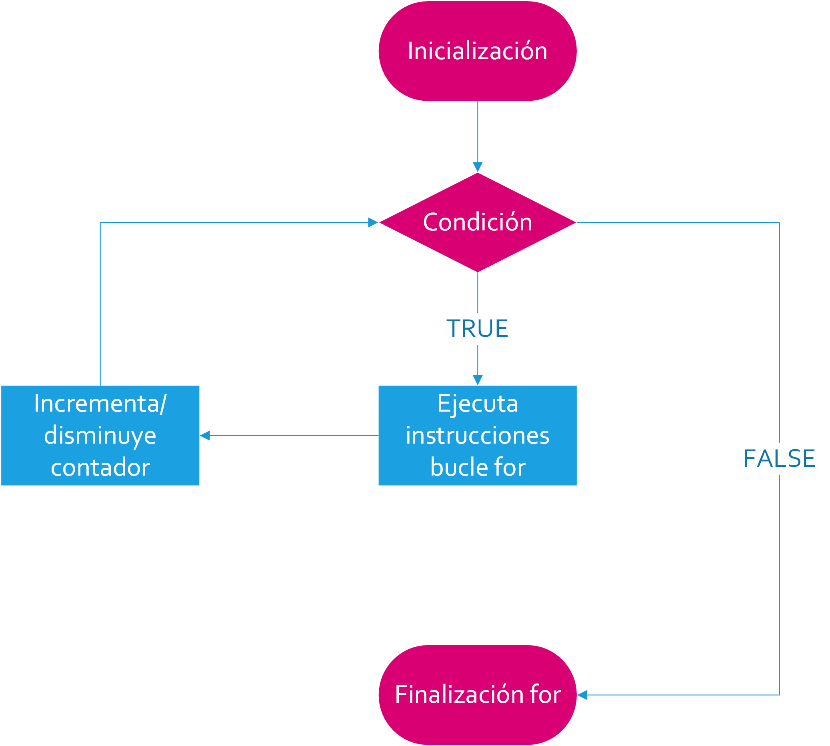
Las instrucciones de repetición, de iteración o bucles, facilitan la repetición de un bloque de instrucciones, un número determinado de veces o mientras se cumpla una condición.

Por lo general, existen dos tipos de estructuras iterativas o bucles en los lenguajes de programación. Encontraremos un tipo de bucle que se ejecuta un número preestablecido de veces, que es controlado por un contador o índice, incrementado en cada iteración. Este tipo de bucle forma parte de la familia for.

Por otro lado, encontraremos un tipo de bucle que se ejecuta mientras se cumple una condición. Esta condición se comprueba al principio o el final de la construcción. Esta variante pertenece a la familia While or Do While.

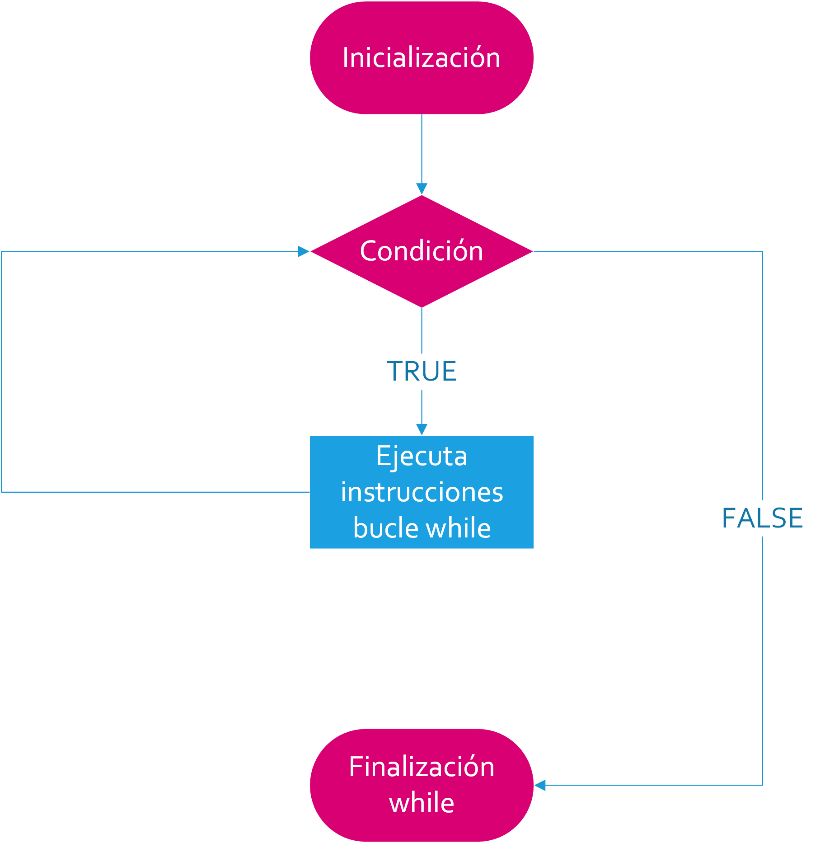
* **FOR**

El bucle for es una estructura iterativa que se ejecuta un número preestablecido de veces, que es controlado por un contador o índice, incrementado en cada iteración.

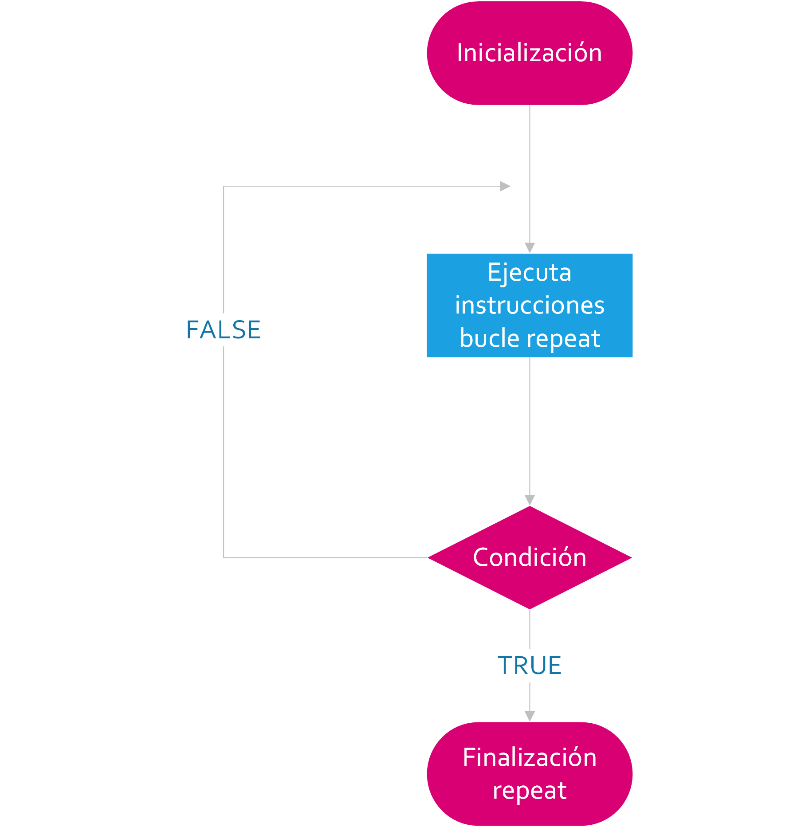


## WHILE

Cuando nos encontramos en la situación en la que no conocemos el número de iteraciones de antemano, podemos hacer uso del bucle while. Este bucle se ejecuta mientras se cumple una condición que se comprueba al principio de la construcción.



* Do While

El bucle DO WHILE es similar a WHILE, excepto que la instrucción o bloque de instrucciones de DO WHILE es ejecutado al menos una vez, sin importar cual es el resultado de la condición.

**PROCEDIMEINTO**

Un procedimiento está compuesto por un grupo de sentencias a las que asigna un nombre (identificador o simplemente nombre del procedimiento) y constituye una unidad de programa. La tarea asignada al procedimiento se ejecutará siempre que se encuentre el identificador (nombre del procedimiento) en el conjunto de sentencias que definen el programa.

Cuando entre las instrucciones de un algoritmo vemos el nombre de un procedimiento (acción o función), decimos que estamos llamando o invocando al procedimiento.

* **ACCIONES. –** Es un tipo de funciones no tienen retorno y se utilizan para realizar tareas que pueden ser más o menos comunes dentro del programa, pero no necesitan retornar nada, sino que simplemente realizan su tarea y al acabar vuelven al flujo normal del programa.
* **FUNCIONES. -** Al igual que las acciones con conjuntos de instrucciones con un nombre, pero se caracterizan por retornar (enviar o devolver) un valor al algoritmo que la llama.

**TIPOS DE PARAMETROS**

Parámetros identifican los valores que se transmiten en una [función](https://www.greelane.com/es/ciencia-tecnolog%C3%ADa-matem%C3%A1ticas/ciencias-de-la-computaci%C3%B3n/definition-of-void-958182/).

Cada parámetro de función tiene un tipo seguido por un identificador, y cada parámetro se separa del siguiente parámetro por una coma.

* **Paso por valor**

El paso de parámetros por valor consiste en copiar el contenido de la variable que queremos pasar en otra dentro del ámbito local de la subrutina, consiste pues en copiar el contenido de la [memoria](https://es.wikipedia.org/wiki/Memoria_principal) del argumento que se quiere pasar a otra [dirección de memoria](https://es.wikipedia.org/wiki/Direcci%C3%B3n_de_memoria), correspondiente al argumento dentro del ámbito de dicha subrutina. Se tendrán dos valores duplicados e independientes, con lo que la modificación de uno no afecta al otro.

* **Paso por referencia**

El paso de parámetros por referencia consiste en proporcionar a la subrutina a la que se le quiere pasar el argumento la [dirección de memoria](https://es.wikipedia.org/wiki/Direcci%C3%B3n_de_memoria) del dato. En este caso se tiene un único valor referenciado (o apuntado) desde dos puntos diferentes, el programa principal y la subrutina a la que se le pasa el argumento, por lo que cualquier acción sobre el parámetro se realiza sobre la misma posición de memoria.