

# POLITECNICO INTERNACIONAL



POLITECNICO INTERNA





# Uso de Lenguaje de Programación

---

Python

---

Profesor **Leonardo...**

# Ciclos WHILE

Python

# Retomemos



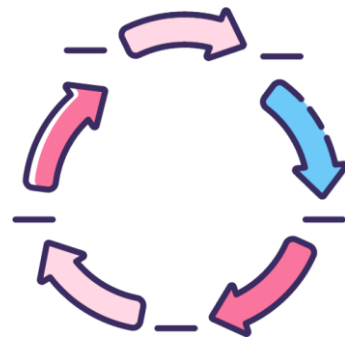
Como se ha visto hasta el momento hemos desarrollado estructuras **SECUENCIALES** y **CONDICIONALES**.

Pero existe otro tipo de estructuras tan importantes como las anteriores que son las estructuras **REPETITIVAS**.

Una estructura repetitiva permite ejecutar una instrucción o un conjunto de instrucciones varias veces.

Una estructura repetitiva se caracteriza por:

- La sentencia o las sentencias que se repiten.
- El test o prueba de condición antes de cada repetición, que motivará que se repitan o no las instrucciones.





# Estructura repetitiva WHILE.



No debemos confundir la representación gráfica de la estructura repetitiva **WHILE (Mientras)** con la estructura condicional **IF (si)**.

**Funcionamiento:** En primer lugar se verifica la condición, si la misma resulta verdadera se ejecutan las operaciones que indicamos por la Validación del Verdadero.

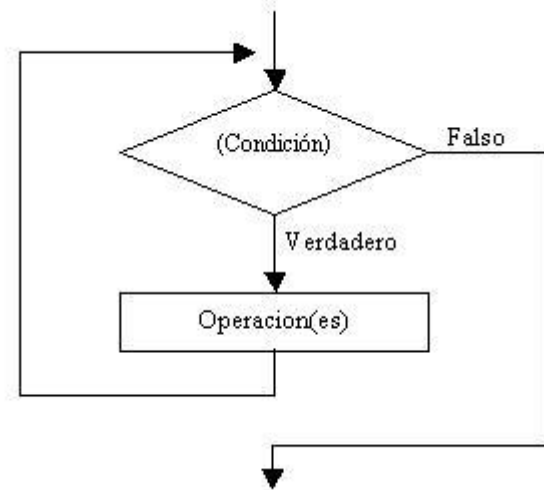
A la Validación del verdadero la graficamos en la parte inferior de la condición. Una línea al final del bloque de repetición la conecta con la parte superior de la estructura repetitiva.

En caso que la condición sea Falsa continúa por la rama del Falso y sale de la estructura repetitiva para continuar con la ejecución del algoritmo.

El bloque se repite **MIENTRAS** la condición sea Verdadera.

**Importante:** Si la condición siempre retorna verdadero estamos en presencia de un ciclo repetitivo infinito. Dicha situación es un error de programación lógico, nunca finalizará el programa.

Representación gráfica de la estructura **WHILE**:

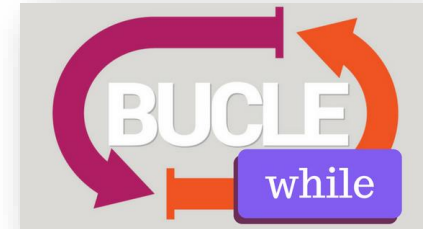


# Bucle o ciclo WHILE

En resumidas cuentas este bucle o también llamado ciclo permite repetir un grupo de instrucciones (escrito en un lenguaje de programación), mientras se cumpla que la condición sea verdadera ( es decir tenga el valor True).

## ¿Cómo funciona el ciclo While?

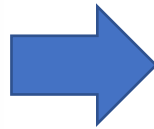
Cuando llega a un bucle while, primero se evalúa la condición, y si es verdadera, se ejecuta las instrucciones (cuerpo del bucle), y luego se vuelve a verificar dicha condición. Este proceso se repite hasta que la condición sea falsa y no se ejecutará el bucle, y después continuará con la ejecución del resto de código.



# Ejemplo:

Uso de Ciclo **While** Números de 1 a 100

```
x=1
while x<=100:
    print(x)
    x=x+1
```

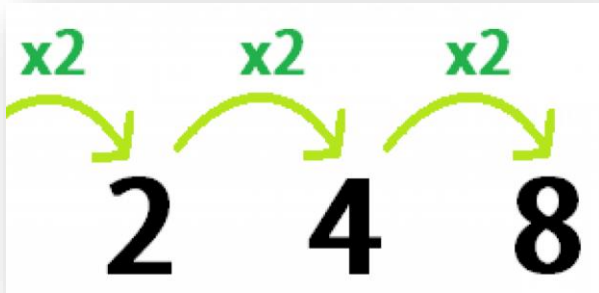


```
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct 5 2020, 15:34:40) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: D:\Mis documentos\acer\Desktop\Algoritmos 2\python Semana 3\while_1.p
Y
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
```

## ALGORITMO # 7: Uso de Ciclo **While**

**DESCRIPCIÓN:** Algoritmo que captura 3 valores enteros:

1. Valor inicial
2. Valor final
3. Intervalo



### Salida:

Señor Usuario digite el numero inicio.

Señor Usuario digite el numero final.

Señor Usuario digite el numero intervalo.

Repeticiones .....

Desea continuar **SI** o **NO**

- Si el valor es **SI** – Regresa y Solicita los valores nuevamente.
- Si el valor es **NO** – Agradece y sale del programa.



## ALGORITMO # 8: Promedio con uso de Ciclo **WHILE**.

**DESCRIPCIÓN:** Calcular la media (**Promedio**) de 3 notas.

Capture el nombre del estudiante, posteriormente cuantas materias tiene este estudiante y finalmente capture 3 notas de cada materia:

### Salida:

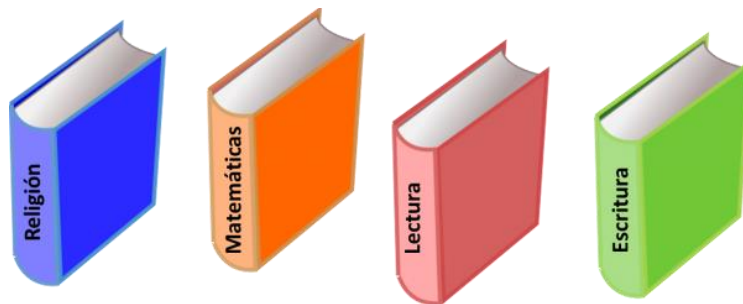
Estudiante **XYZ**

Su nota de la materia 1: XYZ: **???**.

Su nota de la materia 2: XYZ: **???**.

Su nota de la materia x: XYZ: **???**.

Su promedio es: **"Promedio"**.



# Ciclos - FOR

Python

# Estructura repetitiva FOR

Como se pudo ver anteriormente cualquier problema que requiera una estructura repetitiva se puede resolver empleando la estructura **WHILE**, pero hay otra estructura repetitiva cuyo planteo es más sencillo en ciertas situaciones que tenemos que recorrer una lista de datos, esta estructura se llama **FOR**.

En general, la estructura repetitiva **FOR** se usa en aquellas situaciones en las cuales queremos que una variable vaya tomando un valor de una lista definida de valores.

En este Material veremos con una serie de ejemplos el empleo del **FOR**.

```
for (int i=0; i<=10; i++)
```

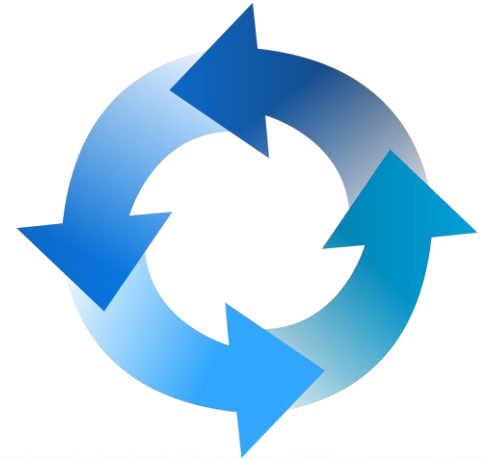
Diagram illustrating the components of the `for` loop structure:

- Inicialización de Variable**: Points to `i=0`.
- Limite de Variable**: Points to `i<=10`.
- Incrementador**: Points to `i++`.

# Bucle o ciclo FOR

**Los ciclos for son estructuras de control cíclicas**, que permite ejecutar una o varias líneas de código en forma iterativa. Para que este proceso se dé a cabo, previamente se tiene que asignar un valor de inicio, un valor final y el tamaño de paso.

La principal diferencia entre **FOR** y **WHILE**, es que el primero se usa cuando se conoce las veces que va a repetir y en el segundo control cíclico no se conoce el número de repeticiones.

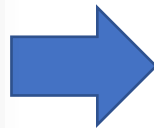


# Ejemplo:

Uso de Ciclo **For** Números de 0 a 100



```
for x in range(101):  
    print(x)
```



```
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct 5 2020, 15:34:40) [MSC v.1927 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>>  
= RESTART: D:\Mis documentos\acer\Desktop\Algoritmos 2\python Semana 3\for_1.py  
0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31
```



## ALGORITMO # 9: Imprimir números consecutivos uso de **FOR**

**DESCRIPCIÓN:** Capturar dos números:

- **Primer numero** que permita saber hasta que numero llegara la impresión
- **Segundo numero** que muestre el intervalo de los números.

**Salida:**

Señor digite el numero al llegar.

1234567890

Señor usuario digite el intervalo de los números.

1234567890

## ALGORITMO # 10: Tablas de multiplicar

**DESCRIPCIÓN:** Capturar dos números:

- **Primer numero** que permita saber hasta que numero llegara la tabla de multiplicar.
- **Segundo numero** que diga el múltiplo.

### Salida:

Señor usuario digite el numero hasta donde llegara la multiplicación.

Señor usuario digite el múltiplo.

**MÚLTIPLOS DE 3**  
**3, 6, 9, 12, 15...**



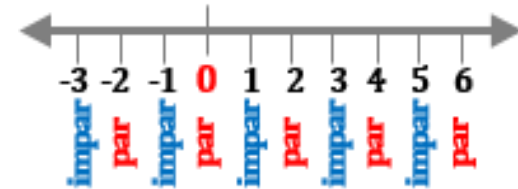
## ALGORITMO # 11: Numero impares y pares

**DESCRIPCIÓN:** Capturar 1 y definir si es par o es impar

### Salida:

Señor usuario digite el numero:

Señor usuario el numero X es (Par / Impar).



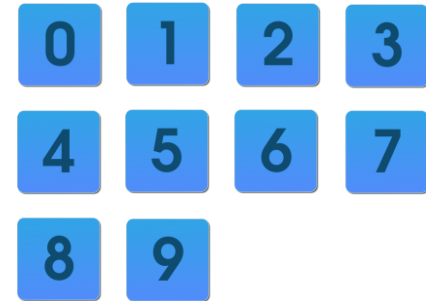
## ALGORITMO # 12: Imprimir números consecutivos y definir si es par o impar

**DESCRIPCIÓN:** Capturar un número que permita saber hasta que numero llegara la impresión

### Salida:

Señor digite el numero al llegar.

1 – IMPAR  
2 – PAR  
3 – IMPAR  
4 – PAR  
... **numero\_llegar**



# ALGORITMO # 13: Horóscopo.

**DESCRIPCIÓN:** Mostrar que signo zodiacal es un usuario  
Capture 2 valores DÍA y MES (Mayúscula) de su cumpleaños.

Capricornio (22/12 – 19/01)	Acuario (20/01 – 17/02)	Piscis (18/02 – 19/03)	Aries (20/03 – 19/04)
			
Tauro (20/04 – 20/05 )	Géminis (21/05 -20/06)	Cáncer (21/06 – 22/07 )	Leo (23/07 – 22/08)
			
Virgo (23/08 – 22/09)	Libra (23/09 – 22/10 )	Escorpio (23/10 – 21/11)	Sagitario (22/11 – 21/12 )
			

**Salida:** Sr Usuario su Fecha de Cumpleaños es: **dia** de **Mes**  
Su Signo zodiacal es : **s\_zodiacal**.



# ALGORITMO # 14: Agenda Familiar.



**DESCRIPCIÓN:** Generar una agenda telefónica con un **For (Para)**

## Salida:

Sr Usuario cuantos familiares tiene: **n\_family**

Sr Usuario digite los **nombres** del 1 familiar: **nom\_family**

Sr Usuario digite los **Apellidos** del 1 familiar: **apel\_family**

Sr Usuario digite el **teléfono** del 1 familiar: **Tel\_family**

Sr Usuario digite la **Fecha de cumpleaños** (día-mes) del 1 familiar: **naci\_family**

Sr Usuario su 1 familiar es: **nom\_family apel\_family**

Su teléfono es **Tel\_family** y Cumple años el **naci\_family**


Sr Usuario digite los **nombres** del 2 familiar: **nom\_family**

**Repetir hasta ..... n\_family**

Sr Usuario su 2 familiar es: **nom\_family apel\_family**

Su teléfono es **Tel\_family** y Cumple años el **naci\_family**





# Optimizar con Ciclos

Python

## ALGORITMO # 15: Cajero Automático – Con Ciclos

**DESCRIPCIÓN:** Con uso de Ciclos optimizar el cajero automático que inicie con un saldo de \$1.000.000.

1. Ingresar Dinero a la cuenta
2. Retirar Dinero de la cuenta
3. Mostrar Dinero de la cuenta
4. Salida



## ALGORITMO # 16: Calculadora – Con Ciclos

**DESCRIPCIÓN:** Con uso de Ciclos optimizar el programa que simule el funcionamiento de una calculadora básica (**Suma – Resta – Multiplicación – división – Porcentaje - Potenciación**).

1. Sumar
2. Restar
3. Multiplicar
4. Dividir
5. Porcentaje
6. Potenciar





**Gracias  
por su atención**



**Profesor Leonardo...**