GUÍA 3

Bucles

Estructura repetitiva while

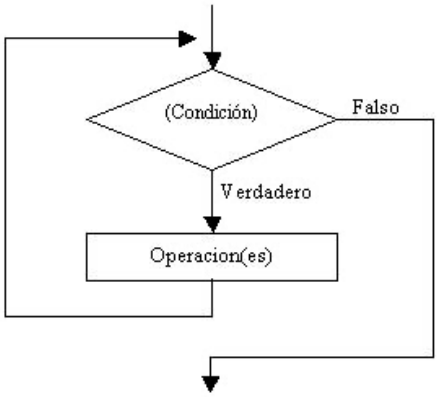
Hasta ahora hemos empleado estructuras SECUENCIALES y CONDICIONALES. Existe otro tipo de estructuras tan importantes como las anteriores que son las estructuras REPETITIVAS.

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.

Representación gráfica de la estructura 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 rama del Verdadero.
A la rama 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.

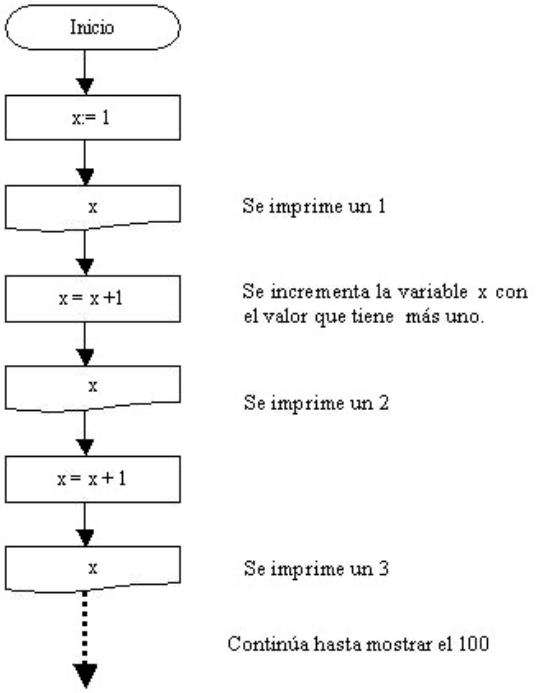
Importante: Si la condición siempre retorna verdadero estamos en presencia de un ciclo repetitivo infinito. Dicha situación es un error de programación lógico, nunca finalizará el programa.

Problema 1:

Realizar un programa que imprima en pantalla los números del 1 al 100.

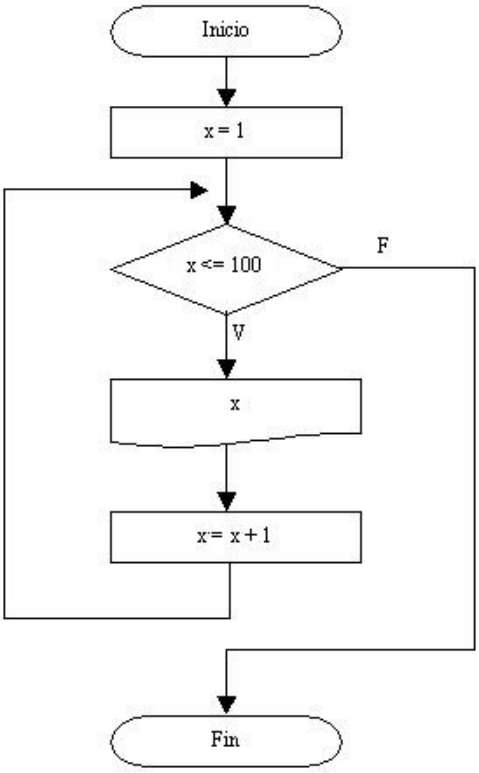
Sin conocer las estructuras repetitivas podemos resolver el problema empleando una estructura secuencial. Iniciamos una variable con el valor 1, luego imprimimos la variable, incrementamos nuevamente la variable y así sucesivamente.

Diagrama de flujo:



Si continuamos con el diagrama veríamos que es casi interminable.
Emplear una estructura secuencial para resolver este problema produce un diagrama de flujo y un programa en Python muy largo.

Ahora veamos la solución empleando una estructura repetitiva while:



Es muy importante analizar este diagrama:
La primera operación inicializa la variable x en 1, seguidamente comienza la estructura repetitiva while y disponemos la siguiente condición (x <= 100), se lee MIENTRAS la variable x sea menor o igual a 100.

Al ejecutarse la condición retorna VERDADERO porque el contenido de x (1) es menor o igual a 100. Al ser la condición verdadera se ejecuta el bloque de instrucciones que contiene la estructura while. El bloque de instrucciones contiene una salida y una operación.
Se imprime el contenido de x, y seguidamente se incrementa la variable x en uno.

La operación x=x+1 se lee como "en la variable x se guarda el contenido de x más 1". Es decir, si x contiene 1 luego de ejecutarse esta operación se almacenará en x un 2.

Al finalizar el bloque de instrucciones que contiene la estructura repetitiva se verifica nuevamente la condición de la estructura repetitiva y se repite el proceso explicado anteriormente.

Mientras la condición retorne verdadero se ejecuta el bloque de instrucciones; al retornar falso la verificación de la condición se sale de la estructura repetitiva y continua el algoritmo, en este caso finaliza el programa.

Lo más difícil es la definición de la condición de la estructura while y que bloque de instrucciones se van a repetir. Observar que si, por ejemplo, disponemos la condición x >=100 (si x es mayor o igual a 100) no provoca ningún error sintáctico pero estamos en presencia de un error lógico porque al evaluarse por primera vez la condición retorna falso y no se ejecuta el bloque de instrucciones que queríamos repetir 100 veces.

No existe una RECETA para definir una condición de una estructura repetitiva, sino que se logra con una práctica continua solucionando problemas.

Una vez planteado el diagrama debemos verificar si el mismo es una solución válida al problema (en este caso se debe imprimir los números del 1 al 100 en pantalla), para ello podemos hacer un seguimiento del flujo del diagrama y los valores que toman las variables a lo largo de la ejecución:

```
        x
        1
        2
        3
        4
        .
        .
         100
         101  Cuando x vale 101 la condición de la estructura repetitiva r
etorna falso,
              en este caso finaliza el diagrama.
```

Importante: Podemos observar que el bloque repetitivo puede no ejecutarse ninguna vez si la condición retorna falso la primera vez.
La variable x debe estar inicializada con algún valor antes que se ejecute la operación x=x+1 en caso de no estar inicializada aparece un error de compilación.

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

Recordemos que un problema no estará 100% solucionado si no codificamos el programa en nuestro caso en Python que muestre los resultados buscados.

Es importante notar que seguido de la palabra clave while disponemos la condición y finalmente los dos puntos. Todo el código contenido en la estructura repetitiva debe estar indentado (normalmente a cuatro espacios)

Probemos algunas modificaciones de este programa y veamos que cambios se deberían hacer para:

```
1 – Imprimir los números del 1 al 500.
```

```
2 – Imprimir los números del 50 al 100.
3 – Imprimir los números del –50 al 0.
4 – Imprimir los números del 2 al 100 pero de 2 en 2 (2,4,6,8 ....100).
```

Respuestas:

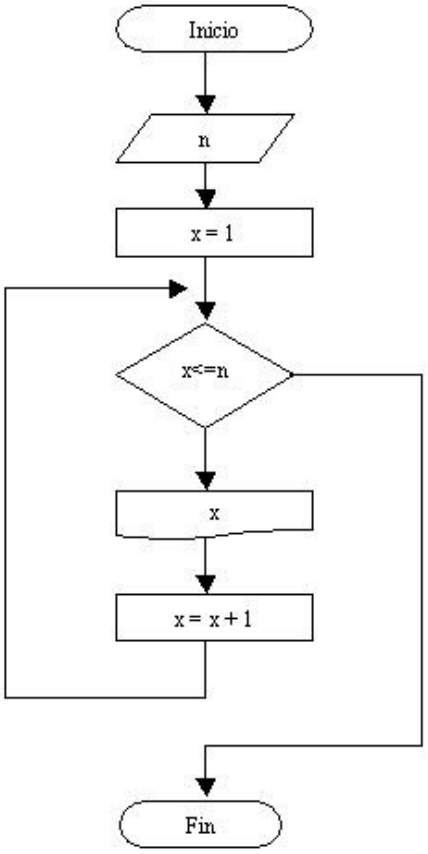
```
1 – Debemos cambiar la condición del while con x<=500.
2 – Debemos inicializar x con el valor 50.
3 – Inicializar x con el valor –50 y fijar la condición x<=0.
4 – Inicializar a x con el valor 2 y dentro del bloque repetitivo increme
ntar a x en 2
    ( x = x + 2 )
```

Problema 2:

Codificar un programa que solicite la carga de un valor positivo y nos muestre desde 1 hasta el valor ingresado de uno en uno.
Ejemplo: Si ingresamos 30 se debe mostrar en pantalla los números del 1 al 30.

Es de FUNDAMENTAL importancia analizar los diagramas de flujo y la posterior codificación en Python de los siguientes problemas, en varios problemas se presentan otras situaciones no vistas en el ejercicio anterior.

Diagrama de flujo:



Podemos observar que se ingresa por teclado la variable n. El operador puede cargar cualquier valor.
Si el usuario carga 10 el bloque repetitivo se ejecutará 10 veces, ya que la condición es “Mientras x<=n ”, es decir “mientras x sea menor o igual a 10”; pues x comienza en uno y se incrementa en uno cada vez que se ejecuta el bloque repetitivo.

A la prueba del diagrama la podemos realizar dándole valores a las variables; por ejemplo, si ingresamos 5 el seguimiento es el siguiente:

n	x
5	1 (Se imprime el contenido de x)
	2 " "
	3 " "
	4 " "
	5 " "

6 (Sale del while porque 6 no es menor o igual a 5)

```
n=int(input("Ingrese el valor final:"))
x=1
while x<=n:
    print(x)
    x=x+1
```

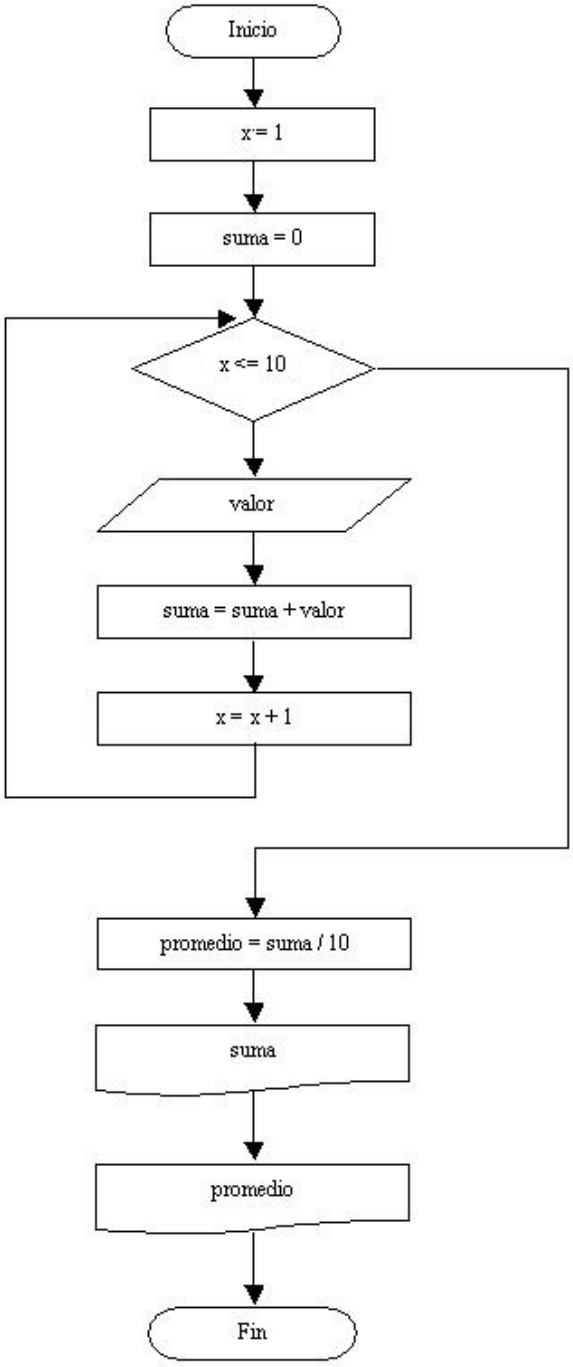
Los nombres de las variables n y x pueden ser palabras o letras (como en este caso)

La variable x recibe el nombre de CONTADOR. Un contador es un tipo especial de variable que se incrementa o disminuye con valores constantes durante la ejecución del programa.
El contador x nos indica en cada momento la cantidad de valores impresos en pantalla.

Problema 3:

Desarrollar un programa que permita la carga de 10 valores por teclado y nos muestre posteriormente la suma de los valores ingresados y su promedio.

Diagrama de flujo:



En este problema, a semejanza de los anteriores, tenemos un CONTADOR llamado x que nos sirve para contar las vueltas que debe repetir el while.
También aparece el concepto de ACUMULADOR (un acumulador es un tipo especial de variable que se incrementa o disminuye con valores variables durante la ejecución del programa)

Hemos dado el nombre de suma a nuestro acumulador. Cada ciclo que se repita la estructura repetitiva, la variable suma se incrementa con el contenido ingresado en la variable valor.

La prueba del diagrama se realiza dándole valores a las variables:

```
valor          suma           x         promedio
                 0            0
(Antes de entrar a la estructura repetitiva estos son los valores).
 5              5             1
16             21             2
 7             28             3
10             38             4
 2             40             5
20             60             6
 5             65             7
 5             70             8
10             80             9
 2             82            10
 8             90            11
                                        9
```

Este es un seguimiento del diagrama planteado. Los números que toma la variable valor dependerá de qué cifras cargue el operador durante la ejecución del programa.
El promedio se calcula al salir de la estructura repetitiva (es decir primero sumamos los 10 valores ingresados y luego los dividimos por 10)

Hay que tener en cuenta que cuando en la variable valor se carga el primer valor (en este ejemplo 5) al cargarse el segundo valor (16) el valor anterior 5 se pierde, por ello la necesidad de ir almacenando en la variable suma los valores ingresados.

```
x=1
suma=0
while x<=10:
   valor=int(input("Ingrese un valor:"))
   suma=suma+valor
```

```
   x=x+1
promedio=suma/10
print("La suma de los 10 valores es")
print(suma)
print("El promedio es")
print(promedio)
```

El resultado del promedio es un valor real es decir con coma. Si queremos que el resultado de la división solo retorne la parte entera del promedio debemos utilizar el operador //:

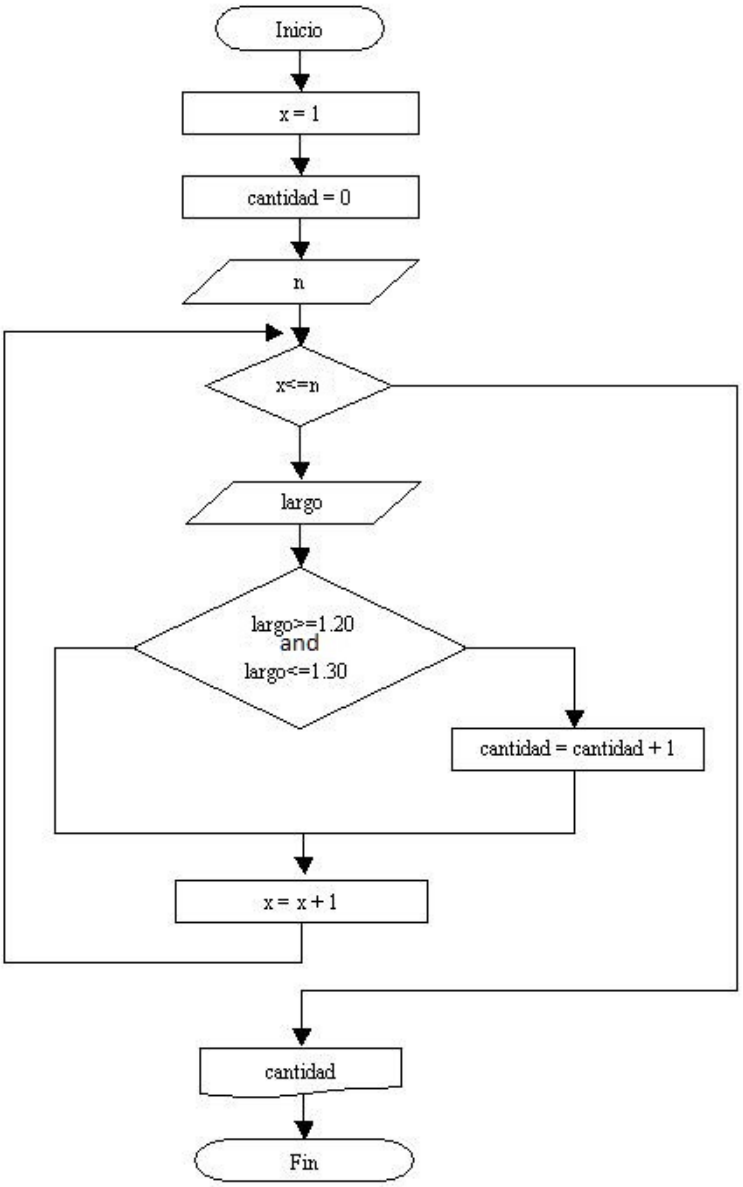
```
promedio=suma//10
```

El intérprete de Python sabe que el promedio se calcula al finalizar el while ya que se encuentra codificado en la columna 1. Las tres instrucciones contenidas en el while están indentadas.

Problema 4:

Una planta que fabrica perfiles de hierro posee un lote de n piezas.
Confeccionar un programa que pida ingresar por teclado la cantidad de piezas a procesar y luego ingrese la longitud de cada perfil; sabiendo que la pieza cuya longitud esté comprendida en el rango de 1.20 y 1.30 son aptas. Imprimir por pantalla la cantidad de piezas aptas que hay en el lote.

Diagrama de flujo:



Podemos observar que dentro de una estructura repetitiva puede haber estructuras condicionales (inclusive puede haber otras estructuras repetitivas que veremos más adelante)

En este problema hay que cargar inicialmente la cantidad de piezas a ingresar (n), seguidamente se cargan n valores de largos de piezas.
Cada vez que ingresamos un largo de pieza (largo) verificamos si es una medida correcta (debe estar entre 1.20 y 1.30 el largo para que sea correcta), en caso de ser correcta la CONTAMOS (incrementamos la variable cantidad en 1)

Al contador cantidad lo inicializamos en cero porque inicialmente no se ha cargado ningún largo de pieza. Cuando salimos de la estructura repetitiva porque se han cargado n largos de piezas mostramos por pantalla el contador cantidad (que representa la cantidad de piezas aptas)

En este problema tenemos dos CONTADORES:

```
x               (Cuenta la cantidad de piezas cargadas hasta el momento)
cantidad        (Cuenta los perfiles de hierro aptos)
```

```
cantidad=0
x=1
n=int(input("Cuantas piezas cargara:"))
while x<=n:
   largo=float(input("Ingrese la medida de la pieza:"))
   if largo>=1.2 and largo<=1.3:
      cantidad=cantidad+1
   x=x+1
print("La cantidad de piezas aptas son")
print(cantidad)
```

Veamos algunas cosas nuevas:

Cuando queremos cargar por teclado un valor con decimales debemos utilizar la función float en lugar de int:

```
    largo=float(input("Ingrese la medida de la pieza:"))
```