Buenos Aires aprende
Agencia de Habilidades para el Futuro

Talento Tech

Iniciación con Python

Clase 06 - “Bucles while”

Buenos Aires Ciudad

¡Les damos la bienvenida!

 Vamos a comenzar a grabar la clase

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Buenos Aires Ciudad

Talento Tech

Clase 05.

Condicionales

1. Cadena de caracteres.
2. Operadores lógicos.
3. Control de flujo: estructuras condicionales (if, else, elif).

Clase 06.

Bucles while

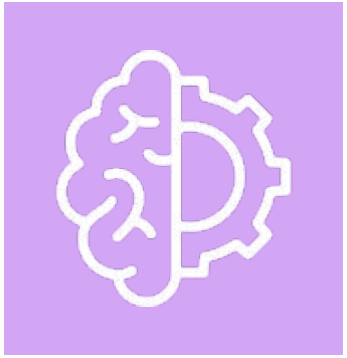
1. Control de flujo: bucles while.
2. Concepto y usos de los contadores.
3. Concepto y usos de los acumuladores.

Clase 07.

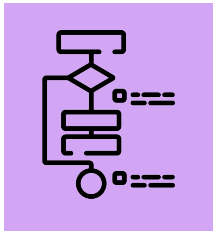
Listas y Tuplas

1. Listas y tuplas: creación y manipulación.
2. Uso de subíndices.
3. Métodos de listas y tuplas.
4. Recorrer una lista con while.

Pero antes...



¡Resolvamos los “**Ejercicios Prácticos**” de la clase anterior!



Bucles while

El bucle while

El **bucle while** te permite repetir acciones en tu programa de manera controlada. En lugar de escribir el mismo código una y otra vez, el bucle hace el trabajo por vos hasta que se cumpla una condición que le hayas asignado.

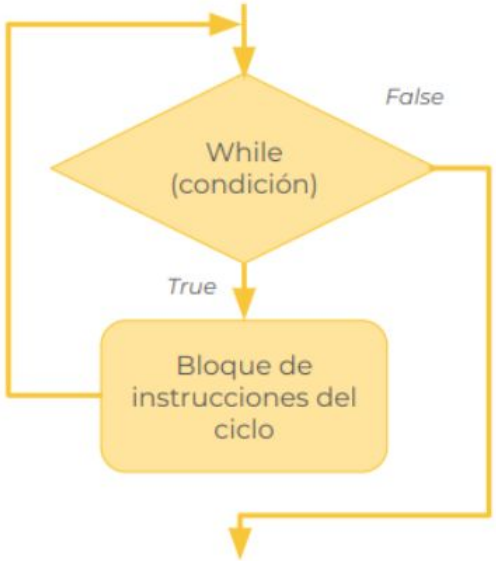
Funciona como una puerta giratoria: el programa sigue girando hasta que le digas que pare.

El bucle while

En pocas palabras, un bucle while es una herramienta súper útil que te permite hacer que tu programa repita una acción hasta que se cumpla una condición.

Podés verlo como una conversación en la que el programa se pregunta algo una y otra vez hasta obtener una respuesta que cumpla con lo que espera.

Mientras la condición que definas al comienzo del bucle siga siendo verdadera (True), seguirá ejecutándose. Una vez que esa condición ya no se cumpla, el bucle se detiene y el programa continúa con el resto de las instrucciones.



¡Probemos while!

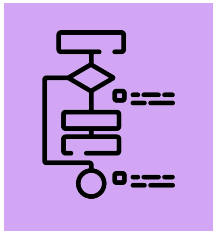
Nuestro primer programa con bucles usa una variable, numero, que comienza en 1. El bucle while sigue ejecutándose mientras numero sea menor o igual a 10.

En cada iteración, el programa imprime el valor actual de **numero** y luego lo incrementa en 1. Cuando **numero** llega a 11, la condición numero **<= 10** ya no se cumple y el bucle se detiene.

El resultado es que se muestran los números del 1 al 10, uno por línea.

```
# Inicializamos la variable
numero = 1

while numero <= 10:
    print(numero)
    # Incrementamos el valor de
numero
    # en 1 en cada iteración
    numero = numero + 1
```

```
1
2
3
4
5
6
7
8
9
10
```



Contadores

Contadores

El contador es uno de los conceptos más básicos y útiles en programación. Básicamente, un contador es una variable que suma (o resta) de a uno cada vez que ocurre algo, como cuando se repite una acción dentro de un bucle. La idea es llevar un registro de cuántas veces ha sucedido algo.

¡Si! En el ejemplo anterior, cuando imprimimos los números de 1 a 10, la variable “numero” era un contador.

Fijate, aqui está otra vez, con comentarios nuevos:

```
# Inicializamos el contador
numero = 1

while numero <= 10:
    print(numero)
    # Incrementamos el contador en 1
    # en cada iteración
    numero = numero + 1
```

¿Cómo funciona un contador?

Vas a usar mucho los contadores, así que necesitas recordar esto:

- Lo primero que hacés es inicializar el contador en cero (o en otro valor si es necesario).
- Luego, dentro del bucle, cada vez que sucede la acción que estás contando le sumás uno al contador.
- Cuando el bucle termina, el contador va a tener guardada la cantidad total de veces que se realizó esa acción.

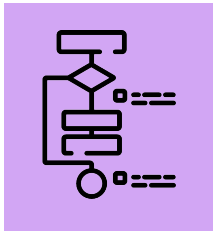
¡Es muy fácil!

Ejemplo

Este ejemplo arranca inicializando el contador en 0. Luego, tenemos un bucle while que se repite mientras contador sea menor que 5. En cada iteración del bucle el programa muestra qué número de “vuelta” estamos ejecutando (sumando 1 al contador para que empiece en 1 en lugar de 0), y después incrementa el valor del contador en 1. Cuando el bucle termina, el programa muestra el valor final del contador, que es 5.

El contador va sumando de a uno cada vez que sucede la acción y, al final del bucle, posee el valor total de veces que el bucle se ejecuta.

```
# En bucle se repite 5 veces
while contador < 5:
    print("Vuelta", contador + 1)

    # Cada vez que ocurre la acción,
    # sumamos 1 al contador
    contador = contador + 1

# Al final del bucle, mostramos el
# valor del contador
print("El contador llegó a:",
contador)
```



Acumuladores

Acumuladores

El acumulador opera de manera similar a un contador pero, en lugar de sumar de a uno, lo que hace es acumular un valor que puede ir variando. Mientras que el contador te dice cuántas veces pasó algo, esta herramienta indica cuánto se acumuló a lo largo del tiempo o de una serie de acciones.

Por ejemplo, en el contexto del **Trabajo Final Integrador (TFI)**, podrías usar un bucle con un acumulador para sumar la cantidad total de productos que tenés en el depósito. O sumar los importes de las ventas a lo largo de todo un mes.
Lo importante aquí es recordar que así como el contador suma de a uno, el acumulador suma el valor que vos le asignes en cada vuelta del bucle.

Ejemplo

Este código simula las ventas de productos en una tienda durante 3 días. Cada día, se pide que se ingrese el monto de las ventas y ese valor se acumula en la variable total. Al mismo tiempo, un contador (dia) se usa para controlar cuántas veces se repite el ciclo. Al final del bucle, se muestra el monto total acumulado de todas las ventas.

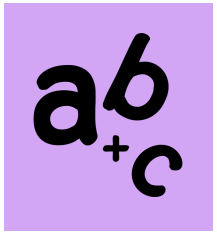
```
total = 0 # Inicializamos el acumulador en cero
dia = 1   # Inicializamos el contador en cero

# Simulamos ventas de productos en una tienda, usando un
bucle
# Supongamos que ingresamos productos durante 3 días
while dia <= 3:
   print("Día", dia)
   venta = float(input("Monto de las ventas del día: "))

   total = total + venta # Acumulamos el monto en el
acumulador
   dia = dia + 1         # Incrementamos el contador:

# Al final del bucle, mostramos el total acumulado
print("El total de las ventas ingresadas es: $", total)
```

¿Dónde usar un acumulador?

Los acumuladores son útiles en situaciones donde necesitás sumar valores de manera progresiva.
Por ejemplo:

- Suma de productos: Llevar el total de productos agregados a un inventario.
- Cálculo del valor del stock: Acumular el valor total del inventario sumando el precio de cada producto disponible por la cantidad existente.
- Sumar montos de ventas: Acumular el total de ventas realizadas en un día o una semana. Sería algo similar a lo visto en el ejemplo anterior.

Estamos seguros que en el TFI vas a usar acumuladores y/o contadores.



Bucles y cadenas

Bucle while y cadenas

Los bucles while son muy útiles para recorrer cadenas de texto en Python, permitiendo que el programa repita una acción mientras se cumpla una condición. Usando un while, podés recorrer cada carácter de la cadena hasta llegar al final.
Para hacerlo, necesitás una variable que funcione como índice, comenzando en 0. Este índice representa la posición del carácter a mostrar o manipular.

A medida que el bucle avanza, el índice se incrementa para moverse por la cadena. Mientras el índice sea menor que la longitud de la cadena (obtenida con len()), el bucle seguirá ejecutándose. Podés acceder a cada carácter utilizando el índice entre corchetes []. Una vez que el índice alcanza la longitud de la cadena, el bucle se detiene. Así, podés usar while para procesar o mostrar cada carácter de una cadena de manera controlada, ya sea para imprimirlos o manipularlos.

¿Lo vemos con un ejemplo? 🤔

Bucle while y cadenas

El programa recorre la cadena de texto. Inicializamos el índice en 0, que será utilizado para recorrer la cadena carácter por carácter mientras el valor del índice sea menor que la longitud de la misma. Usamos print() para mostrar tanto el número del carácter (calculado como indice + 1) como el carácter en sí, accediendo a la cadena con cadena[indice]. Luego, incrementamos el índice para avanzar al siguiente carácter de la cadena y así hasta terminar. ¡Probalo!

```
# Definimos la cadena de texto
cadena = "Talento Tech"

# Inicializamos el contador para los índices
indice = 0

# Usamos un bucle while para recorrer la cadena
while indice < len(cadena):
    # Mostramos el número y el carácter
    print("caracter", indice + 1, ":", cadena[indice])

    # Incrementamos el contador en 1
    indice = indice + 1
```

¡Vamos a la práctica! 

Ejercicios Prácticos

Optativos | No entregables

Control de stock de productos

Desarrollá un programa que permita al usuario ingresar el nombre de varios productos y la cantidad en stock que hay de cada uno. El programa debe seguir pidiendo que ingrese productos hasta que el usuario decida salir, ingresando "salir" como nombre de producto. Después de que el bucle termine, el programa debe mostrar la cantidad total de productos ingresados.

Tips:

- Vas a necesitar un contador.
- Tené presente las estructuras condicionales.

Ejercicios Prácticos

Optativos | No entregables

Validación de precios de productos

Escribí un programa que permita al usuario ingresar el precio de un producto, pero que sólo acepte valores mayores a 0. Si el usuario ingresa un valor inválido (negativo o cero), el programa debe mostrar un mensaje de error, y volver a pedir el valor hasta que se ingrese uno válido. Al final, el programa debe mostrar el precio aceptado.

Tips:

- Antes de empezar, pensá si es necesario usar contadores o acumuladores.
- Recordá que input() siempre devuelve cadenas de caracteres.

¡NUEVO CUESTIONARIO EN CAMPUS!

La resolución del cuestionario es de carácter obligatorio para desbloquear los contenidos de las próximas 2 clases

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Buenos Aires Ciudad

Talento Tech