Tiempo de lectura
8 min

Contenido de la clase

Grabación de la Clase 3

Principales Objetivos de Aprendizaje para esta Clase

- Conocer el concepto de los Flujos de Control
- Conocer cómo se usan los Condicionales en Programación
- Conocer el concepto de Ciclos Iterativos

Flujos de Control

Condicionales

Dejanos tu feedback! 👍

Contenido de la clase

Esto se logra mediante la sentencia **if**.
Con la sentencia **elif** se puede agregar un número arbitrario de condiciones.
Por otra parte, se puede ejecutar código si la/s condición/es no se cumple/n con la sentencia **else**.

```
>>> valor = 0
>>> if (valor < 0):
>>>     print('El número es negativo')
>>> elif (valor > 0):
>>>     print('El número es positivo')
>>> else:
>>>     print('El número es igual a cero')
El número es igual a cero
```

Ciclos Iterativos o Loops

Son bloques de código que se repiten una cierta cantidad de veces en función de ciertas condiciones.

Un ciclo **for** repite un bloque de código tantas veces como elementos haya dentro del rango entre 1 y 10:

```
>>> for n in range(1,10):
>>>     print(n)
1
2
3
4
5
6
7
8
9
```

Un ciclo **while** repite un bloque de código mientras que cierta condición se cumpla:

```
>>> n = 1
>>> while (n < 10):
```

Dejanos tu feedback! 👍

Contenido de la clase


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

Hemos llegado hasta este punto y se repasaron algunos de los conceptos más fundamentales de la programación y de Python, pero es necesario detenerse en algunos detalles, que tienen que ver precisamente con el lenguaje que estamos utilizando:

- **En Python es importante la indentación, notar que el código que se ejecuta dentro de una sentencia if, for o while está indentado.**
- **También es importante notar los ":"**
- **En Python, cada vez que hagamos referencia a un rango, por ejemplo "1,10" el primer número se incluye y el último no.**

Sentencia Break

La sentencia break permite alterar el comportamiento de los bucles while y for. Concretamente, permite terminar con la ejecución del bucle. Esto significa que una vez se encuentra la palabra break, el bucle se habrá terminado.
Veamos cómo podemos usar el break con bucles for. El range(5) generaría 5 iteraciones, donde la i valdría de 0 a 4. Sin embargo, en la primera iteración, terminamos el bucle prematuramente.
El break hace que nada más empezar el bucle, se rompa y se salga sin haber hecho nada.

```
>>> for i in range(5):
>>>   print(i)
>>>   break
0
```

Dejanos tu feedback! 👍

Contenido de la clase

algo muy útil porque si ya encontramos lo que estábamos buscando, no tendría mucho sentido seguir iterando la lista, ya que desperdiciaríamos recursos.

```
>>> cadena = 'Python'
>>> for letra in cadena:
>>>     if letra == 'h':
>>>         print("Se encontró la h")
>>>         break
>>>     print(letra)
P
y
t
Se encontró la h
```

El break también nos permite alterar el comportamiento del while. En el ejemplo, la condición while True haría que la sección de código se ejecutará indefinidamente, pero al hacer uso del break, el bucle se romperá cuando x valga cero.

```
>>> x = 5
>>> while True:
>>>     x -= 1
>>>     print(x)
>>>     if x == 0:
>>>         break
>>> print("Fin del bucle")
4
3
2
1
0
Fin del bucle
```

Por norma general, y salvo casos muy concretos, si ves un while True, es probable que haya un break dentro del bucle.

Sentencia Continue

Dejanos tu feedback! 👍

Contenido de la clase

el código restante en la iteración actual y vuelve al principio en el caso de que aún queden iteraciones por completar.
La diferencia con la sentencia break es que el continue no rompe el bucle, si no que pasa a la siguiente iteración saltando el código pendiente.
En el siguiente ejemplo vemos como al encontrar la letra P se llama al continue, lo que hace que se salte el print(). Es por ello por lo que no vemos la letra P impresa en pantalla.

```
>>> cadena = 'Python'
>>> for letra in cadena:
>>>     if letra == 'P':
>>>         continue
>>>     print(letra)
y
t
h
o
n
```

Homework

Completa la tarea descrita en el archivo **README**

Si tienes dudas sobre este tema, puedes consultarlas en el canal #python de Slack

Clase en vivo de Resolución de ejercicios.

Los martes/jueves (segun corresponda) a las 18HS ARG cada dos semanas hacemos una clase de

Dejanos tu feedback! 👍

Hecho con 💛 por alumnos de Henry

Grabación de la Clase 3

Principales Objetivos de Aprendizaje para esta Clase

Flujos de Control

- Condicionales
- Ciclos Iterativos o Loops
- Sentencia Break
- Sentencia Continue

Homework

Clase en vivo de Resolución de ejercicios.

Dejanos tu feedback! 👍