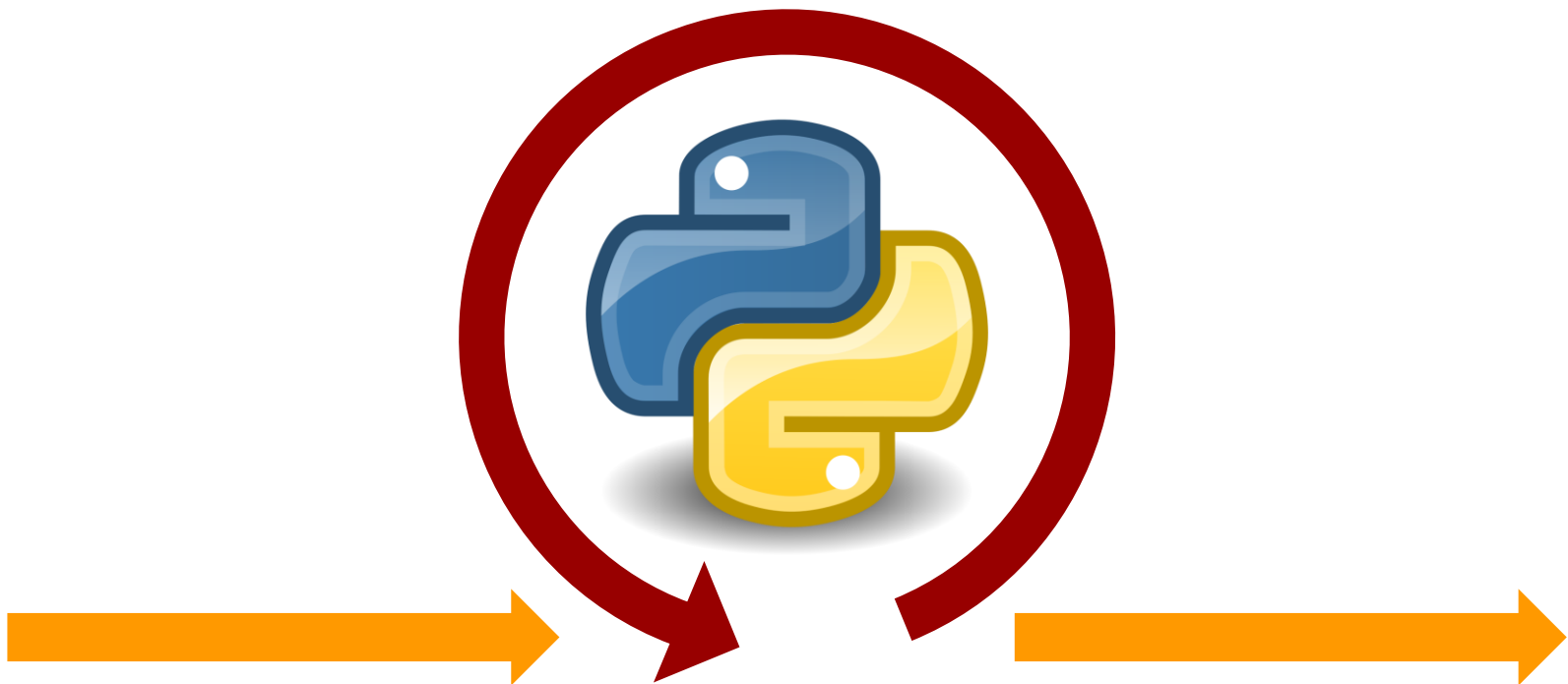


# Python (IV): Ciclo for

Programación (tics 100)  
Semestre 01/2020

# ¿Qué veremos hoy?

Ciclos en Python... segunda iteración :)



# Ciclos en Python

La clase anterior vimos que habían dos posibles tipos de ciclo:

- while: basados en una condición
- for: basados en una cantidad fija de repeticiones

# Ciclo while

## ▼ Algoritmo

Imprimir los números enteros del 1 al 10:

1. Comenzar con número igual a 1
2. Decir el número
3. Sumar 1 al número
4. Si número es menor a 11 repetir desde el paso 2

## ▼ Python

```
numero = 1
while numero < 11:
    print(numero)
    numero = numero + 1
```

# Ciclo for

En realidad un ciclo for en Python no se basa en repetir una cantidad fija de veces...

Lo que hace es “recorrer” los elementos de una colección

# Ciclo for

Lo que hace **for** es repetir una serie de instrucciones por cada elemento.

Supongamos que tenemos un conjunto:

`frutas = {Pera, Manzana, Naranja}`

Y queremos, para cada una de las frutas,

- Lavarla
- Pelarla
- Trozarla

# Ciclo for

for fruta in frutas:

    lavar fruta

    pelar fruta

    trozar fruta

Esto hará que:

- 1ero lavemos la Pera, pelemos la Pera y trocemos la Pera
- 2do lavemos, pelemos y trocemos la Manzana
- 3ero lavemos, pelemos y trocemos la Naranja

Y terminamos

# Ciclo for

```
for fruta in frutas:  
    lavar fruta  
    pelar fruta  
    trozar fruta
```

Entonces, las instrucciones se repiten **3 veces**,  
pues frutas tiene **3 elementos**



Entonces

**¿Cómo repetimos  
instrucciones un número de  
veces?**

# Función range()

```
range([start], stop, [step])
```

La función range nos entrega una **colección de elementos** que van desde el valor inicial start (o 0 si no está definido), hasta el valor stop - 1 (stop no incluido), avanzando con paso step (o 1 si no está definido)

▼ Python

```
numeros = range(1,11)
print(list(numeros))
# imprime [1, 2, 3, 4, 5, 6, 7, 8, 9 ,10]

impares = range(1, 11, 2)
print(list(impares))
# imprime [1, 3, 5, 7, 9]

con_el_cero = range(11)
print(list(con_el_cero))
# imprime [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

¿Y qué tiene que ver la función range() con un ciclo for?



# Ciclo for

## ▼ Algoritmo

Imprimir los números enteros del 1 al 10:

1. Comenzar con número igual a 1
2. Decir el número
3. Sumar 1 al número
4. Si número es menor a 11  
repetir desde el paso 2



## ▼ Python

```
numero = 1
for _ in range(10):
    print(numero)
    numero = numero + 1
```



# Ciclo for

## ▼ Algoritmo

Imprimir los números enteros del 1 al 10:

1. Comenzar con número igual a 1
2. Decir el número
3. Sumar 1 al número
4. Si número es menor a 11  
repetir desde el paso 2



## ▼ Python

```
for numero in range(10):  
    print(numero + 1)
```

Another Way

# Ciclo for

## ▼ Algoritmo

Imprimir los números enteros del 1 al 10:

1. Comenzar con número igual a 1
2. Decir el número
3. Sumar 1 al número
4. Si número es menor a 11 repetir desde el paso 2



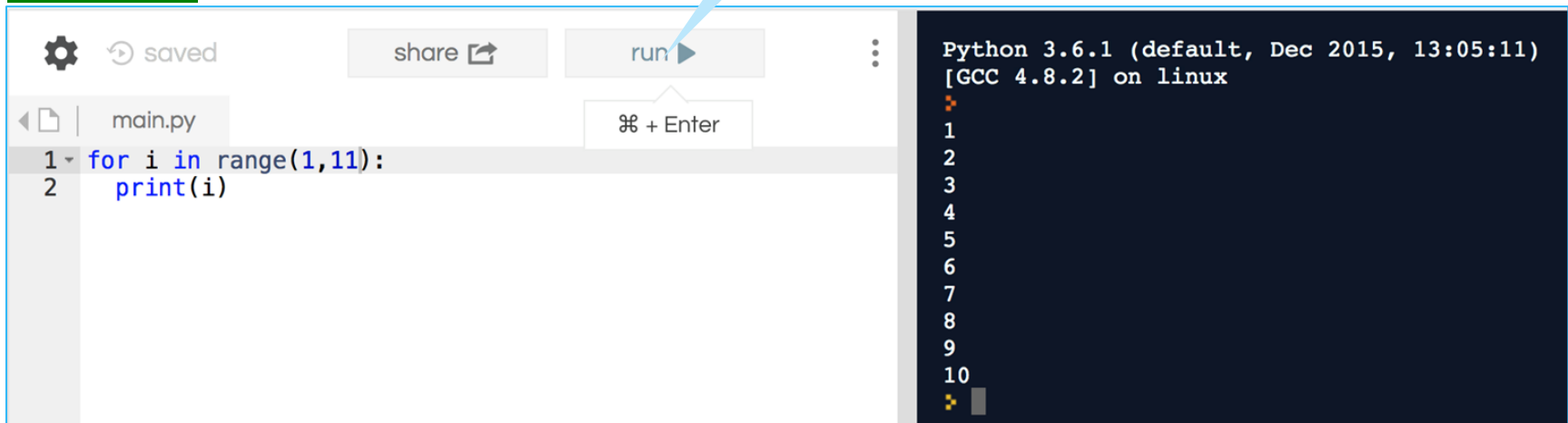
## ▼ Python

```
for i in range(1,11):
    print(i)
```



En la instrucción: `for i in range(a,b)`, el valor *i* toma los valores: *a* , *a*+1, *a*+2, ..., *b*-1.

## ▼ repl.it



The screenshot shows the repl.it interface. On the left, the code editor displays the Python code: `1 for i in range(1,11):` and `2 print(i)`. Above the editor are buttons for settings, saved status, share, and a 'run' button with a play icon. A tooltip indicates that pressing '⌘ + Enter' will run the code. On the right, the output terminal shows the execution results: 'Python 3.6.1 (default, Dec 2015, 13:05:11) [GCC 4.8.2] on linux' followed by the numbers 1 through 10, each on a new line.

# Problemos



*Tiempo : 1  
minuto*

¿Qué aparece en pantalla al ejecutar el siguiente programa en Python?

```
for i in range(2,14,3):  
    print(i)
```


# Probemos






Tiempo : 1  
minuto


¿Qué aparece en pantalla al ejecutar el siguiente programa en Python?


```
for i in range(2,14,3):
    print(i)
```






saved

share


run





main.py

1
for i in range(2,14,3):

2
 print(i)

⌘ + Enter

```

Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux

2
5
8
11

```

# ¡Ahora tú!



Tiempo : 5  
minutos

Usando ciclo for, escribe un programa en Python que pida un número al usuario (N) e imprima un rectángulo relleno de símbolos #. El ancho del rectángulo es de 6 símbolos # y la altura es de N símbolos # (se imprime un total de  $6 \cdot N$  símbolos). Ejemplo:

```
Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
>
¿Cuántas filas? 4
#####
#####
#####
#####
>
```



# ¡Ahora tú!



Tiempo : 5  
minutos

Usando ciclo for, escribe un programa en Python que pida un número al usuario (N) e imprima un rectángulo relleno de símbolos #. El ancho del rectángulo es de 6 símbolos # y la altura es de N símbolos # (se imprime un total de  $6 \cdot N$  símbolos). Ejemplo:

```
N = int(input('¿Cuántas filas? '))
for _ in range(N):
    print('#####')
```

# ¡Ahora tú!



Tiempo : 5  
minutos

Usando ciclo for, escribe un programa en Python que muestre los divisores del número que el usuario ingresa por teclado. Ejemplo:

```
Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
>
Ingresa un número 15
Los divisores del número 15 son:
1
3
5
15
>
```

# ¡Ahora tú!



Tiempo : 5  
minutos

Usando ciclo for, escribe un programa en Python que muestre los divisores del número que el usuario ingresa por teclado. Ejemplo:

```
N = int(input('Ingresa un número: '))
print(f'Los divisores del {N} son:')
for i in range(1, N + 1):
    if N%i == 0:
        print(i)
```

# Resumen

```
for i in coleccion:  
    instrucciones de ciclo for
```

El ciclo **for** recorre los elementos que forman parte de la colección **uno a uno**.  
Una colección puede ser:

1. un texto (recorrerá cada letra),
2. un conjunto de elementos (como entrega range) o
3. una lista

▼ Python

```
for i in 'hola':  
    print(i) #imprime cada letra de hola
```

▼ Python

```
for i in range(1,11):  
    print(i) #imprime los nros del 1 al 10
```

Espera un momento...

**... ¿Qué es una lista?  
(continuará)**

# Python (IV): Ciclo for

Programación (tics 100)  
Semestre 01/2020

## Ejercicio en parejas o tríos

- ▶ Escriba un programa que permita calcular la suma de los  $n$  primeros números de Fibonacci.
- ▶ La serie de Fibonacci parte con las cifras 0 y 1. La siguiente cifra siempre será la suma de las dos anteriores.
- ▶ Las cinco primeras  $n$  cifras de la serie de Fibonacci cuando  $n = 5$  son:  
0, 1, 1, 2, 3, 5
- ▶ Es decir  $n = (n-1) + (n-2)$
- ▶ Y su suma de los primeros 5 números de la serie de Fibonacci es  $0+1+1+2+3+5 = 12$

## Ejercicio en parejas o tríos

- ▶ El programa debe mostrar en pantalla las  $n$  cifras de la serie, donde el valor de  $n$  es indicado por el usuario del programa.
- ▶ Además, se debe indicar la suma de estos valores.