

Introducción a la programación con Python

Funciones y Bucles

Alexis Rodríguez

Marcel Morán C

Esquema

- ¿Que es una función?
- Terminología y sintaxis de una función
- Scope local vs Scope global
- El stack de llamadas
- ¿Que es un bucle?
- Terminología y sintaxis de un bucle

Reporte de acciones

Google = 114.77, 115.07, 118.12

- Reportar la sumatoria total
- Reportar la media
- Reportar el día con la mayor ganancia

Bloomberg



Reporte de acciones

Google = 114.77, 115.07, 118.12

- Reportar la sumatoria total
- Reportar la media
- Reportar el dia con la mayor ganancia

media

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i = \frac{x_1 + x_2 + \dots + x_n}{n}$$

```
primer_dia = 114.77
segundo_dia = 115.07
tercer_dia = 118.12
sum_dias = primer_dia + segundo_dia + tercer_dia
media = sum_dias/3
print("la sumatoria total es: " + str(sum_dias))
print("la media es: " + str(media))
if (primer_dia >= segundo_dia and primer_dia >= tercer_dia):
    print("El primer dia fue donde hubo mayor ganancia")
elif(segundo_dia >= primer_dia and segundo_dia >= tercer_dia):
    print("El segundo dia fue donde hubo mayor ganancia")
else:
    print("El tercer dia fue donde hubo mayor ganancia")
```

Resultado

la sumatoria total es: 347.96
la media es: 115.98666666666666
El tercer dia fue donde hubo
mayor ganancia

Reporte de acciones



Google = 114.77, 115.07, 118.12 , Amazon = 138.23, 133.22, 133.62

- Reportar la sumatoria total
- Reportar la media
- Reportar el dia con la mayor ganancia

Resultado

Acciones de Google resumen:

la sumatoria total es: 347.96

la media es: 115.98666666666666

El tercer dia fue donde hubo mayor ganancia

Acciones de Amazon resumen:

la sumatoria total es: 405.07

la media es: 135.02333333333334

El primer dia fue donde hubo mayor ganancia

```
primer_dia = 114.77
segundo_dia = 115.07
tercer_dia = 118.12
sum_dias = primer_dia + segundo_dia + tercer_dia
media = sum_dias/3
print("Acciones de Google resumen:")
print("-----")
print("la sumatoria total es: " + str(sum_dias))
print("la media es: " + str(media))
if (primer_dia >= segundo_dia and primer_dia >= tercer_dia):
    print("El primer dia fue donde hubo mayor ganancia")
elif(segundo_dia >= primer_dia and segundo_dia >= tercer_dia):
    print("El segundo dia fue donde hubo mayor ganancia")
else:
    print("El tercer dia fue donde hubo mayor ganancia")
print("-----")
```

```
print("Acciones de Amazon resumen:")
print("-----")
primer_dia = 138.23
segundo_dia = 133.22
tercer_dia = 133.62
sum_dias = primer_dia + segundo_dia + tercer_dia
media = sum_dias/3
print("la sumatoria total es: " + str(sum_dias))
print("la media es: " + str(media))
if (primer_dia >= segundo_dia and primer_dia >= tercer_dia):
    print("El primer dia fue donde hubo mayor ganancia")
elif(segundo_dia >= primer_dia and segundo_dia >= tercer_dia):
    print("El segundo dia fue donde hubo mayor ganancia")
else:
    print("El tercer dia fue donde hubo mayor ganancia")
print("-----")
```

Reporte de acciones

Google, Amazon, Apple , Microsoft, Moran&RodriguezInc,

- Reportar la sumatoria total
- Reportar la media
- Reportar el dia con la mayor ganancia



```
primer_dia = 118.27
segundo_dia = 119.87
tercer_dia = 121.87
sum_dias = primer_dia + segundo_dia + tercer_dia
media = sum_dias/3

print("Acciones de Google resumen:")
print("-----")
print("La sumatoria total es: " + str(sum_dias))
print("La media es: " + str(media))
if primer_dia < segundo_dia and primer_dia < tercer_dia:
    print("El primer dia fue donde hubo mayor ganancia")
elif segundo_dia < primer_dia and segundo_dia < tercer_dia:
    print("El segundo dia fue donde hubo mayor ganancia")
else:
    print("El tercer dia fue donde hubo mayor ganancia")
print("-----")

primer_dia = 118
segundo_dia = 119.22
tercer_dia = 121.82
sum_dias = primer_dia + segundo_dia + tercer_dia
media = sum_dias/3

print("Acciones de Amazon resumen:")
print("-----")
print("La sumatoria total es: " + str(sum_dias))
print("La media es: " + str(media))
if primer_dia < segundo_dia and
    primer_dia < tercer_dia:
    print("El primer dia fue donde hubo mayor ganancia")
elif segundo_dia < primer_dia and segundo_dia < tercer_dia:
    print("El segundo dia fue donde hubo mayor ganancia")
else:
    print("El tercer dia fue donde hubo mayor ganancia")
print("-----")

primer_dia = 118
segundo_dia = 119.22
tercer_dia = 121.82
sum_dias = primer_dia + segundo_dia + tercer_dia
media = sum_dias/3

print("Acciones de Apple resumen:")
print("-----")
print("La sumatoria total es: " + str(sum_dias))
print("La media es: " + str(media))
if primer_dia < segundo_dia and
    primer_dia < tercer_dia:
    print("El primer dia fue donde hubo mayor ganancia")
elif segundo_dia < primer_dia and segundo_dia < tercer_dia:
    print("El segundo dia fue donde hubo mayor ganancia")
else:
    print("El tercer dia fue donde hubo mayor ganancia")
print("-----")

primer_dia = 118
segundo_dia = 119
tercer_dia = 120
sum_dias = primer_dia + segundo_dia + tercer_dia
media = sum_dias/3

print("Acciones de Microsoft resumen:")
print("-----")
print("La sumatoria total es: " + str(sum_dias))
print("La media es: " + str(media))
if primer_dia < segundo_dia and primer_dia < tercer_dia:
    print("El primer dia fue donde hubo mayor ganancia")
elif segundo_dia < primer_dia and segundo_dia < tercer_dia:
    print("El segundo dia fue donde hubo mayor ganancia")
else:
    print("El tercer dia fue donde hubo mayor ganancia")
print("-----")

primer_dia = 2000
segundo_dia = 2000
tercer_dia = 2000
sum_dias = primer_dia + segundo_dia + tercer_dia
media = sum_dias/3

print("Acciones de MoranRodriguezInc resumen:")
print("-----")
print("La sumatoria total es: " + str(sum_dias))
if primer_dia < segundo_dia and primer_dia < tercer_dia:
    print("El primer dia fue donde hubo mayor ganancia")
elif segundo_dia < primer_dia and segundo_dia < tercer_dia:
    print("El segundo dia fue donde hubo mayor ganancia")
else:
    print("El tercer dia fue donde hubo mayor ganancia")
print("-----")
```

Reporte de acciones

Google, Apple , Microsoft, Moran&RodriguezInc, AZ Technology Inc.,Behance LLC,Advanced Players,Blue Space,Centric Services,CO-AX Inc.,Cyber Group,Golden Section,High Tech Fabrication,Kingston Ttechnology Co Inc.,Micron Technology Inc.,Optionscity Software Inc.,Nationwide Group,Riverbed Technology,S4 Technologies,SilkRoad,Seaco Technologies,SWC Technology Partners,Technology Informer,Trade Technologies,Warrior,Woodlands Corp.,Worldwide Tech Services,Techcess Group,Target Technology Co,Sky High Tech,Seppi Technology Associates,Root Level,Percento,Lighthouse

- Reportar la sumatoria total
- Reportar la media
- Reportar el día con la mayor ganancia
- Reportar la varianza
- Reportar la desviación estándar
- Reportar la compañía con la menor desviación



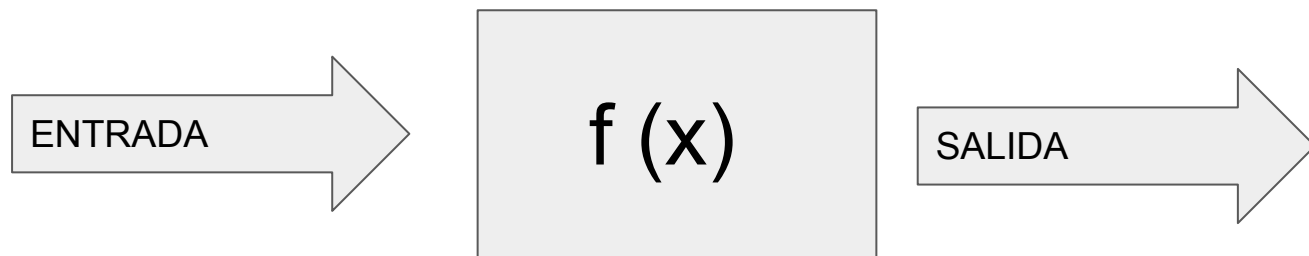
Reporte de acciones

Google, Apple , Microsoft, Moran&RodriguezInc, AZ Technology Inc.,Behance LLC,Advanced Players,Blue Space,Centric Services,CO-AX Inc.,Cyber Group,Golden Section,High Tech Fabrication,Kingston Ttechnology Co Inc.,Micron Technology Inc.,Optionscity Software Inc.,Nationwide Group,Riverbed Technology,S4 Technologies,SilkRoad,Seaco Technologies,SWC Technology Partners,Technology Informer,Trade Technologies,Warrior,Woodlands Corp.,Worldwide Tech Services,Techcess Group,Target Technology Co,Sky High Tech,Seppi Technology Associates,Root Level,Percento,Lighthorse

- Reportar la sumatoria total
- Reportar la media
- Reportar el día con la mayor ganancia
- Reportar la varianza
- Reportar la desviación estándar
- Reportar la compañía con la menor desviación
- **Información de los últimos 4 días**



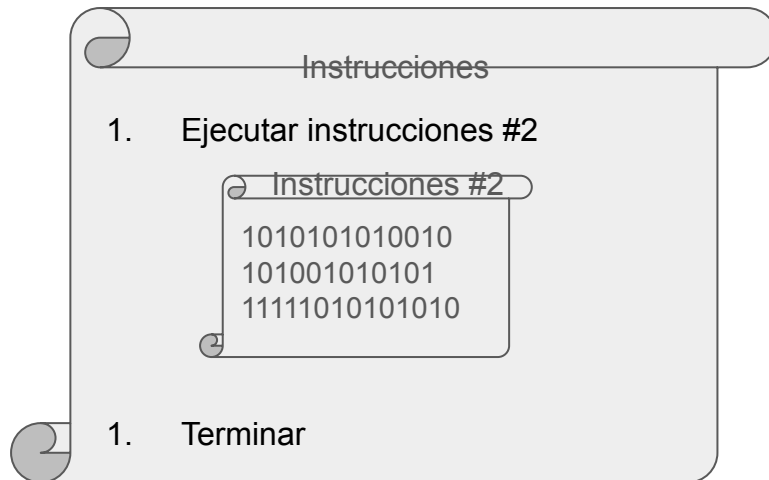
¿Que es una función?



X	X!	Y
1	$1! = 1$	1
2	$2! = 2 \times 1!$	2
3	$3! = 3 \times 2! = 3 \times 2 \times 1!$	6

¿Que es una función?

- Una función es como un miniprograma
- `Input()`, `print()` son ejemplos de funciones definidas en python
- Funciones pueden ser creadas
- Las funciones nos ayudan a descomponer nuestras instrucciones o código
- Facilita el reuso de código y depuración



¿Que es una función?

- Una función es como un miniprograma
- **Input()**, **print()** son ejemplos de funciones definidas en python
- Funciones pueden ser creadas
- Las funciones nos ayudan a descomponer nuestras instrucciones o código
- Facilita el reuso de código y depuración

Resultado

Acciones de Google resumen:

```
-----  
la sumatoria total es: 347.96  
la media es: 115.98666666666666  
El tercer dia fue donde hubo mayor ganancia  
-----
```

Acciones de Amazon resumen:

```
-----  
la sumatoria total es: 405.07  
la media es: 135.02333333333334  
El primer dia fue donde hubo mayor ganancia  
-----
```

```
primer_dia = 114.77  
segundo_dia = 115.07  
tercer_dia = 118.12  
sum_dias = primer_dia + segundo_dia + tercer_dia  
media = sum_dias/3  
print("Acciones de Google resumen:")  
print("-----")  
print("la sumatoria total es: " + str(sum_dias))  
print("la media es: " + str(media))  
if (primer_dia >= segundo_dia and primer_dia >= tercer_dia):  
    print("El primer dia fue donde hubo mayor ganancia")  
elif(segundo_dia >= primer_dia and segundo_dia >= tercer_dia):  
    print("El segundo dia fue donde hubo mayor ganancia")  
else:  
    print("El tercer dia fue donde hubo mayor ganancia")  
print("-----")  
print("Acciones de Amazon resumen:")  
print("-----")  
primer_dia = 138.23  
segundo_dia = 133.22  
tercer_dia = 133.62  
sum_dias = primer_dia + segundo_dia + tercer_dia  
media = sum_dias/3  
print("la sumatoria total es: " + str(sum_dias))  
print("la media es: " + str(media))  
if (primer_dia >= segundo_dia and primer_dia >= tercer_dia):  
    print("El primer dia fue donde hubo mayor ganancia")  
elif(segundo_dia >= primer_dia and segundo_dia >= tercer_dia):  
    print("El segundo dia fue donde hubo mayor ganancia")  
else:  
    print("El tercer dia fue donde hubo mayor ganancia")  
print("-----")
```

¿Que es una función?

- Una función es como un miniprograma
- `Input()`, `print()` son ejemplos de funciones definidas en python
- Funciones pueden ser creadas
- Las funciones nos ayudan a descomponer nuestras instrucciones o código
- Facilita el reuso de código y depuración

Resultado

Acciones de Google resumen:

la sumatoria total es: 347.96

la media es: 115.98666666666666

El tercer dia fue donde hubo mayor ganancia

Acciones de Amazon resumen:

la sumatoria total es: 405.07

la media es: 135.02333333333334

El primer dia fue donde hubo mayor ganancia

```
def operaciones(primer_dia, segundo_dia, tercer_dia, compania):  
    sum_dias = primer_dia + segundo_dia + tercer_dia  
    media = sum_dias/3  
    print("Acciones de " + compania + " resumen:")  
    print("-----")  
    print("la sumatoria total es: " + str(sum_dias))  
    print("la media es: " + str(media))  
    if (primer_dia >= segundo_dia and primer_dia >= tercer_dia):  
        print("El primer dia fue donde hubo mayor ganancia")  
    elif(segundo_dia >= primer_dia and segundo_dia >= tercer_dia):  
        print("El segundo dia fue donde hubo mayor ganancia")  
    else:  
        print("El tercer dia fue donde hubo mayor ganancia")  
    print("-----")  
  
operaciones(114.77, 115.07, 118.12, "Google")  
operaciones(138.23, 133.22, 133.62, "Amazon")
```

Terminología y sintaxis de una función

- **Palabra def para definir una función**
- Nombre de la función
- Cuerpo
- Palabra para regresar un resultado
- El valor a regresar
- Llamar
- Argumentos
- Parametros
- Pasar

```
def es_numero_par(num):  
    print("Dentro del cuerpo")  
    return num % 2 == 0
```

```
entero = int(input("ingrese numero"))
```

```
if(es_numero_par(entero)):  
    print("Numero es par")  
else:  
    print("Numero es par")
```

Terminología y sintaxis de una función

- Palabra def para definir una función
- **Nombre de la función**
- Cuerpo
- Palabra para regresar un resultado
- El valor a regresar
- Llamar
- Argumentos
- Parametros
- Pasar

```
def es_numero_par(num):  
    print("Dentro del cuerpo")  
    return num % 2 == 0
```

```
entero = int(input("ingrese numero"))
```

```
if(es_numero_par(entero)):  
    print("Numero es par")  
else:  
    print("Numero es par")
```

Terminología y sintaxis de una función

- Palabra def para definir una función
- Nombre de la función
- **Cuerpo**
- Palabra para regresar un resultado
- El valor a regresar
- Llamar
- Argumentos
- Parametros
- Pasar

```
def es_numero_par(num):
```

```
    print("Dentro del cuerpo")
```

```
    return num % 2 == 0
```

```
entero = int(input("ingrese numero"))
```

```
if(es_numero_par(entero)):
```

```
    print("Numero es par")
```

```
else:
```

```
    print("Numero es par")
```

Terminología y sintaxis de una función

- Palabra def para definir una función
- Nombre de la función
- Cuerpo
- **Palabra para regresar un resultado**
- El valor a regresar
- Llamar
- Argumentos
- Parametros
- Pasar

```
def es_numero_par(num):
```

```
    print("Dentro del cuerpo")
```

```
    return num % 2 == 0
```

```
entero = int(input("ingrese numero"))
```

```
if(es_numero_par(entero)):
```

```
    print("Numero es par")
```

```
else:
```

```
    print("Numero es par")
```


Terminología y sintaxis de una función

- Palabra def para definir una función
- Nombre de la función
- Cuerpo
- Palabra para regresar un resultado
- **El valor a regresar**
- Llamar
- Argumentos
- Parametros
- Pasar

```
def es_numero_par(num):  
    print("Dentro del cuerpo")  
    return num % 2 == 0
```

```
entero = int(input("ingrese numero"))
```

```
if(es_numero_par(entero)):  
    print("Numero es par")  
else:  
    print("Numero es par")
```

Terminología y sintaxis de una función

- Palabra def para definir una función
- Nombre de la función
- Cuerpo
- Palabra para regresar un resultado
- El valor a regresar
- **Llamar**
- Argumentos
- Parametros
- Pasar

```
def es_numero_par(num):  
    print("Dentro del cuerpo")  
    return num % 2 == 0
```

```
entero = int(input("ingrese numero"))
```

```
if es_numero_par(entero):  
    print("Numero es par")  
else:  
    print("Numero es par")
```

Terminología y sintaxis de una función

- Palabra def para definir una función
- Nombre de la función
- Cuerpo
- Palabra para regresar un resultado
- El valor a regresar
- Llamar
- **Argumentos**
- **Parametros**
- Pasar

```
def es_numero_par(num):
```

```
    print("Dentro del cuerpo")
```

```
    return num % 2 == 0
```

```
entero = int(input("ingrese numero"))
```

```
if es_numero_par(entero):
```

```
    print("Numero es par")
```

```
else:
```

```
    print("Numero es par")
```

None value

- Tipo de dato None
- Especifica la ausencia de un valor
- Se usa para evitar que algo se confunda con una variable real
- Cualquier función que no especifique la instrucción return regresa un None

```
contenedor = print('Hola')  
>>> 'Hola'  
print(contenedor)  
>>> None
```



0 | NONE

Scope local vs Scope global

- Scope local : Parámetros y variables en una función llamada viven dentro de la función
- Scope global: Variables asignadas fuera la función

```
def funct_cambiar():  
    variable_local = 0  
  
variable_global = 10
```

Scope local vs Scope global

- Scope local : Parámetros y variables en una función llamada viven dentro de la función
- Scope global: Variables asignadas fuera la función
- **Variables tienen que ser globales o locales**
- **Código en el scope global no puede usar variables locales**

```
def funct_cambiar():  
    variable_local = 0
```

```
funct_cambiar()  
print(variable_local)
```

```
NameError: name 'variable_local' is not defined
```

Scope local vs Scope global

- Scope local : Parámetros y variables en una función llamada viven dentro de la función
- Scope global: Variables asignadas fuera la función
- **Variables tienen que ser globales o locales**
- **Código en el scope global no puede usar variables locales**
- **Código en scope local puede usar variables globales**

```
def funct_cambiar():  
    print('funcion llamada')  
    variable = 0
```

```
variable = 10  
print(variable)  
funct_cambiar()  
print(variable)  
10  
funcion llamada  
10
```

Scope local vs Scope global

- Scope local : Parámetros y variables en una función llamada viven dentro de la función
- Scope global: Variables asignadas fuera la función
- **Variables tienen que ser globales o locales**
- **Código en el scope global no puede usar variables locales**
- **Código en scope local puede usar variables globales**

```
def funct_cambiar():  
    global variable  
    print('funcion llamada')  
    variable = 0
```

```
variable = 10  
print(variable)  
funct_cambiar()  
print(variable)  
10  
funcion llamada  
0
```


Scope local vs Scope global

- Scope local : Parámetros y variables en una función llamada viven dentro de la función
- Scope global: Variables asignadas fuera la función
- **Variables tienen que ser globales o locales**
- **Código en el scope global no puede usar variables locales**
- **Código en scope local puede usar variables globales**
- **Código de scope local de una función no puede ser variables de otros scope locales**

```
def funct_cambiar():  
    variable_local = 0  
  
def funct_referencia():  
    ref_var_local = variable_local  
  
funct_cambiar()  
funct_referencia()  
NameError: name 'variable_local' is not defined
```

Scope local vs Scope global

- Scope local : Parámetros y variables en una función llamada viven dentro de la función
- Scope global: Variables asignadas fuera la función
- **Variables tienen que ser globales o locales**
- **Código en el scope global no puede usar variables locales**
- **Código en scope local puede usar variables globales**
- **Código de scope local de una función no puede ser variables de otros scope locales**
- **Variables pueden usar el mismo símbolo si viven en diferentes scopes**

```
def funct_cambiar():  
    variable_local = 0
```

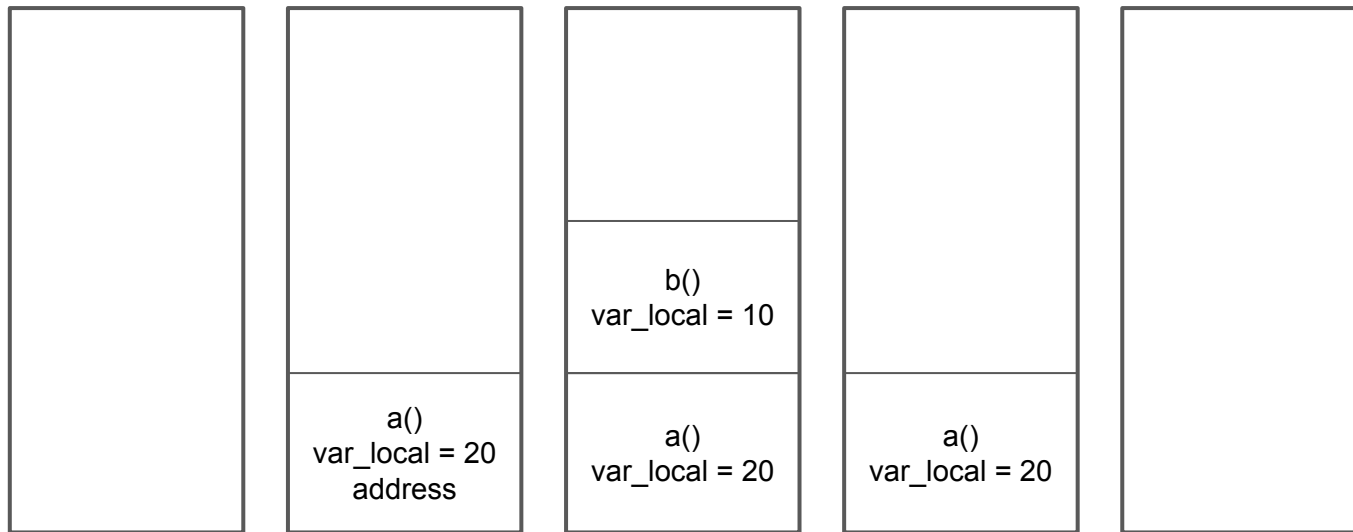
```
def funct_referencia():  
    variable_local = 5
```

```
variable_local = 10
```

La pila (Stack) de llamadas

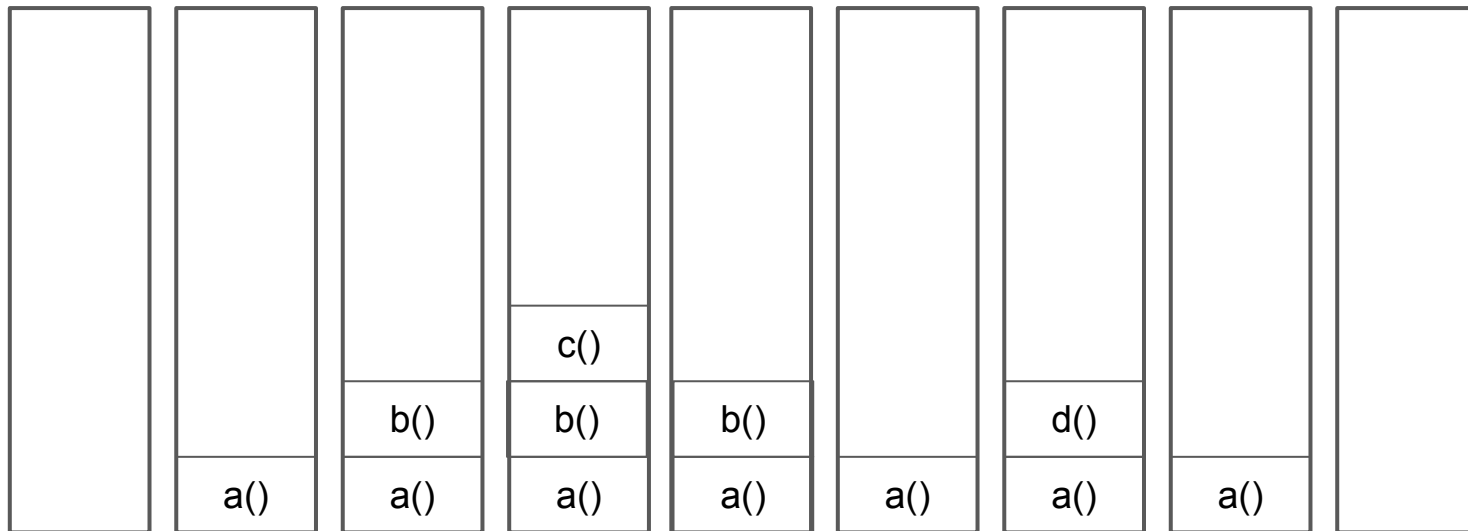
- Una función puede llamar a otra función
- La información de una función que llama a otra se añade a la pila de llamadas

```
def a():  
    var_local = 10  
    b()  
def b():  
    var_local = 20  
    c()  
def c():  
    pass  
a()
```



La pila (Stack) de llamadas

```
def a():  
    b()  
    d()  
def b():  
    c()  
def c():  
    pass  
def d():  
    pass  
  
a()
```



La pila (Stack) de llamadas

```
def primera_funcion():  
    print('Estamos dentro de la primera funcion')  
    segunda_funcion()  
  
def segunda_funcion():  
    print('Estamos dentro de la segunda funcion')  
    tercera_funcion()  
  
def tercera_funcion():  
    print('Estamos dentro de la tercera funcion')  
    funcion_que_genera_excepcion()  
  
def funcion_que_genera_excepcion():  
    print('Estamos al inicio de la funcion que genera excepcion')  
    resultado = 25 / 0  
    print('Estamos al final de la funcion que genera excepcion')
```

primera_funcion()

```
C:\Users\alexi\anaconda3\envs\adaexam\python.exe D:/Home/teaching/semester_2/cursos_de_python/clases/ejemplo_stack.py  
Traceback (most recent call last):  
  File "D:/Home/teaching/semester_2/cursos_de_python/clases/ejemplo_stack.py", line 18, in <module>  
    primera_funcion()  
  File "D:/Home/teaching/semester_2/cursos_de_python/clases/ejemplo_stack.py", line 3, in primera_funcion  
    segunda_funcion()  
  File "D:/Home/teaching/semester_2/cursos_de_python/clases/ejemplo_stack.py", line 7, in segunda_funcion  
    tercera_funcion()  
  File "D:/Home/teaching/semester_2/cursos_de_python/clases/ejemplo_stack.py", line 11, in tercera_funcion  
    funcion_que_genera_excepcion()  
  File "D:/Home/teaching/semester_2/cursos_de_python/clases/ejemplo_stack.py", line 15, in funcion_que_genera_excepcion  
    resultado = 25 / 0  
ZeroDivisionError: division by zero  
Estamos dentro de la primera funcion  
Estamos dentro de la segunda funcion  
Estamos dentro de la tercera funcion  
Estamos al inicio de la funcion que genera excepcion
```

¿Que es un bucle?

Buenos dias

```
>>> nombre = input('Cual es tu nombre?: ')
>>> print('Hola', nombre)
>>> print('Hola', nombre, 'de nuevo!')
>>> print('Hola', nombre, 'de nuevo!')
```

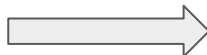


Buenos días

Hola Marcel
Hola Marcel de nuevo!
~~Hola Marcel de nuevo!~~

Buenos dias

```
>>> nombre = input('Cual es tu nombre?: ')
>>> print('Hola', nombre)
>>> print('Hola', nombre, 'de nuevo!')
>>> print('Hola', nombre, 'de nuevo!')
>>> print('Hola', nombre, 'de nuevo!')
>>> print('Hola', nombre, 'de nuevo!')
```

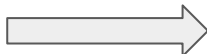


Buenos dias

Hola Marcel
 Hola Marcel de nuevo!
 Hola Marcel de nuevo!
 Hola Marcel de nuevo!
 Hola Marcel de nuevo!

Buenos días

Hola Marcel
Hola Marcel de nuevo!
~~Hola Marcel de nuevo!~~
Hola Marcel de nuevo!
Hola Marcel de nuevo!
Hola Marcel de nuevo!
Hola Marcel de nuevo!
Hola Marcel de nuevo!
Hola Marcel de nuevo!



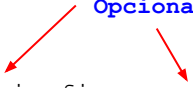
Terminología y sintaxis de un bucle

- **for loop**

Nos permite ejecutar un bloque de código tantas veces como sean especificadas con anticipacion.

```
for <variable> in range(<algun_numero>):  
    <expresion>  
    <expresion>  
    . . .
```

Opcionales



`range(inicio, fin, aumento)`

Nota: fin no está incluido

Ejemplos:

```
valores = ''  
for indice in range(10):  
    valores += str(indice) + ' '  
print('Valores:', valores)
```

Valores: 0 1 2 3 4 5 6 7 8 9

```
valores = ''  
for indice in range(5, 10):  
    valores += str(indice) + ' '  
print('Valores:', valores)
```

Valores: 5 6 7 8 9

```
valores = ''  
for indice in range(5, 10, 2):  
    valores += str(indice) + ' '  
print('Valores:', valores)
```

Valores: 5 7 9

Terminología y sintaxis de un bucle

```
nombre = input('Cual es tu nombre?')
repeticiones = 10
for index in range(repeticiones):
    if index == 0:
        print('Buenos dias')
    else:
        print('Hola', nombre, 'de nuevo!')
```

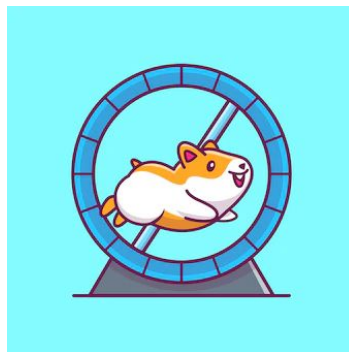

Terminología y sintaxis de un bucle

- **while** loop

Nos permite ejecutar un bloque de código mientras una condición sea verdadera.

```
while <condicion>:  
    <expresion>  
    <expresion>  
    . . .
```

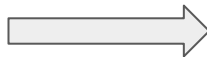
Boolean



Terminología y sintaxis de un bucle

- **while** loop

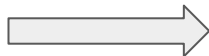
```
nombre = input('Cual es tu nombre?')
repeticion = 0
while repeticion < 10:
    if repeticion == 0:
        print('Hola', nombre)
    else:
        print('Hola', nombre, 'de nuevo!')
    repeticion += 1
```

[illegible]

Terminología y sintaxis de un bucle

- **while** loop

```
acumulador = 0
while True:
    operador = input('Operador: ')
    valor = float(input('Valor: '))
    if operador == '+':
        acumulador += valor
    elif operador == '-':
        acumulador -= valor
    elif operador == '*':
        acumulador *= valor
    elif operador == '/':
        acumulador /= valor
    else:
        print('Operador invalido! Terminando el programa')
        print('Resultado: ', acumulador)
        break
```



Operador invalido!
Terminando el programa
Resultado: -4.0

Conclusión

- Una función es como un mini programa, facilita el reuso de código y la depuración
- Una función se crea con la instrucción *def nombre_fun()* :
- Una función puede regresar un valor con la instrucción *return*
- Parámetros no son lo mismo que argumentos
- *None* es un tipo de dato que especifica la ausencia de un valor
- Toda función que no especifique *return*, regresara un tipo *None*
- Las reglas de local scope y global scope deben ser respetadas para evitar problemas
- La pila de llamadas guarda un cuadro(frame) cuando una función es llamada
- Un bucle facilita el reuso de código y la depuración

Retroalimentación

- Para retroalimentación dirigirse al siguiente enlace <https://forms.gle/d3xaTuzbAcdZZBh59> .
- Déjanos saber qué podemos hacer para mejorar el curso

