



UNIVERSIDAD ADOLFO IBÁÑEZ
FACULTAD DE INGENIERÍA Y CIENCIAS

INTRODUCCIÓN A PYTHON NUMPY (2)

Background designed by kjpargeter / Freepik

Miguel Carrasco
Junio 2019

- ▶ Estructuras de control
- ▶ Listas
- ▶ Funciones integradas
- ▶ Funciones



```
#Read item in dictionary
for key, value in item.FidValue.items():
    typeOfFID = mapFIDType[key]
    if(typeOfFID == "DATE"):
        d = datetime.datetime.strptime(value, "%Y-%m-%d")
        dataCal = datetime.date(d.year, d.month, d.day)
        FidAndValue = FidValue + [(key, dataCal)]
    else:FidAndValue = FidValue + [(key, value)]
```

```
try:
    start = date(int(self.start_year.get()),
                 self.months.index(self.start_month.get()),
                 int(self.start_day.get()))
    end = date(int(self.end_year.get()),
               self.months.index(self.end_month.get()),
               int(self.end_day.get()))
```





Tiempo : 10 minutos

El salario se calcula según las horas semanales trabajadas y el valor de la hora. Si trabaja más de 44 semanales, cada hora adicional se paga con un 1,5 % más.

Problema: Calcular el salario semanal de un empleado.

Datos de entrada: Nombre del empleado, horas semanales trabajadas, valor de la hora normal

Datos de salida: Sueldo a pagar

Restricciones: Las instrucciones disponibles en Python



Tiempo : 10 minutos

Calcular el salario semanal de un empleado. El salario se calcula según las horas semanales trabajadas y el valor de la hora. Si trabaja más de 44 semanales, cada hora adicional se paga con un 1,5 % más.

```
print("Ingresa tu nombre ")
nombre = input()
print("Ingresa horas semanales trabajadas")
horas = int(input())
print("Ingresa tu valor de hora ")
val_h = float(input("Ingresa tu valor de hora "))

if (horas > 44):
    sueldo = 44 * val_h + (horas - 44)*val_h*1.015
else:
    sueldo = horas * val_h

print(nombre," tu sueldo es ", sueldo)
```

▼ Python

funciones

```
print("Ingresa tu nombre ")
nombre = input()
```

Obtener
nombre del empleado

```
print("Ingresa horas semanales trabajadas")
horas = int(input())
```

Obtener
horas trabajadas

```
print("Ingresa tu valor de hora ")
val_h = float(input())
```

Obtener
nombre del empleado

```
if (horas > 44):
    sueldo = 44 * val_h + (horas - 44)*val_h*1.015
else:
    sueldo = horas * val_hora
```

Calcular
sueldo

```
print(nombre," tu sueldo es ", sueldo)
```

Mostrar
nombre del empleado



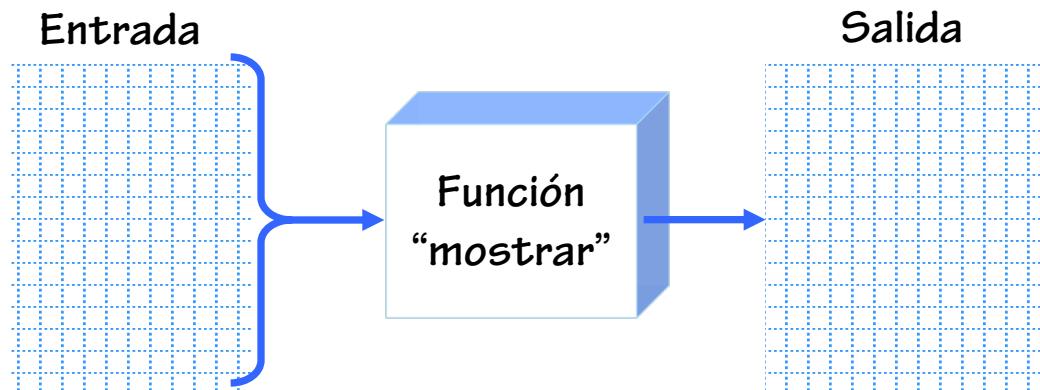
Las funciones son pequeños partes de código que pueden ser reutilizados nuevamente. Son independientes de nuestro programa.



Es un subprograma que realiza una tarea específica.



Puede recibir cero o más valores del programa que los llama y devolver 0 o 1 valor a dicho programa.





Las funciones se definen antes del código principal.
La función más sencilla no recibe ni retorna ningún valor.

Función

```
def nombreFuncion():
    <instrucciones>
    return
```

```
def holaMundo():
    print("Nuestra primera funcion")
```



Las funciones se definen antes del código principal.

La función más sencilla no recibe ni retorna ningún valor.

Una función se puede invocar desde cualquier parte, simplemente escribiendo su nombre como una instrucción

Función

```
def nombreFuncion():
    <instrucciones>
    return
```



invocación
o llamado



```
def holaMundo():
    print("Nuestra primera funcion")
```

```
holaMundo()
```

Solución con una función

función

```
def calculaSueldo(h_trabajadas , val_h):
    if (h_trabajadas > 44):
        sueldoFinal = 44 * val_h + (h_trabajadas - 44)* val_h *1.015
    else:
        sueldoFinal = h_trabajadas * val_h
    return sueldoFinal
```

programa ppal.

```
print("Ingresa tu nombre ")
nombre = input()
print("Ingresa horas semanales trabajadas")
horas = int(input())
print("Ingresa tu valor de hora ")
val_hora = float(input())

sueldo = calculaSueldo(horas , val_hora)
print( nombre, "tu sueldo es", sueldo )
```



Una función puede recibir uno o más parametros (valores). Estos valores son copiados dentro de la función y pueden ser empleados dentro de ella. Solo la función los puede emplear.

Función

```
def nombreFuncion(<var1>,..., <varn>):  
    <instrucciones>  
    return
```



invocación
o llamado



```
def mostrarPantalla(input):  
    print (input)  
    return
```



```
mostrarPantalla("holaMundo")
```



La función puede finalizar en cualquier parte de su código.

Función

```
def nombreFuncion(<var1>,..., <varn>):  
    <instrucciones>  
    return
```

```
def numPrimo(num):  
    for i in range(2,num,1):  
        if (num%i==0):  
            print ("NO primo")  
            return  
        print ("SI primo")  
    return
```



invocación
o llamado

numPrimo(21)



Una función puede devolver un **valor** finalizando la ejecución de la función

Función

```
def nombreFuncion(<var1>,..., <varn>):  
    <instrucciones>  
    return <vari/valor>
```



invocación
o llamado

```
if (numPrimo(21)==0):  
    print ("NO primo")  
else:  
    print ("SI primo")
```



Tiempo : 5 minutos

Cree una función que reciba un texto (varTexto) y un número (varNum) como parámetros y muestre por pantalla el texto un número varNum de veces.

```
def repetirTexto(varTexto, varNum):  
    for i in range(1,varNum+1,1)  
        print (varTexto)
```



Tiempo : 10 minutos ¿Qué realizan las siguientes funciones?

```
def F1(var1, var2 , var3):  
    if (var1<var2):  
        var1=var2  
    if (var1<var3):  
        var1=var3  
    print (var1)  
    return
```

Muestra el mayor de los 3 números

```
def F2(var1, var2 , var3):  
    flag=1  
    if (var1>var2):  
        flag=0  
    else:  
        if  
(var2>var3):  
            flag=0  
    return flag
```

retorna 1 si los números ingresados están ordenados de menor a mayor, 0 en caso contrario.

Variables locales



Cada función tiene sus propias variables (**no las comparte con ninguna otra función**).



Los parámetros también son variables propias de la función.



invocación
o llamado

```
def F1(var1, var2 , var3):  
    print( "en la función", var1+var2 +var3 )
```

F1(4,5,6)

```
print( "en algoritmo", var1+var2 +var3 )
```



*Estas variables no han sido definidas
en el código principal, por lo cual se
produce error*



Tiempo : 5 minutos ¿Qué valores se muestran por pantalla?

```
def suma(var1, var2):
    var1 = var1+var2
    return (var1)

var1 = 10
var2 = 5
var2 = suma(var1, var2)
print("var1 es ", var1, " y var2 es ", var2 )
```

El programa mostrará que var1 es 10 y var2 es 15.



Se denomina a la forma por defecto en que un valor es traspasado del código principal o función a otra función.

```
def F1(var1, var2):  
    var1 = var1+var2  
    return (var1)
```

```
var1 = 10  
var2 = 5  
var2 = F1(var1, var2)  
print("var1 es ", var1, " y var2 es ", var2 )
```



A la variable var1 de F1 se le asigna el valor 5, mientras que a la variable var2 de F1 se le asigna el valor 10.

El valor de var1 (10) y var2 (5) se envían a la función F1, no la variable



Como se puede observar, se traspasan los valores de las variables **NO las variables en sí**.



Tiempo : 10 minutos

```
def cambios(x, y)
    3->
        x = x+1
    4->
        y = 20
    return y

a = 10
1->
b = 5
2->
b= cambios(b, a)
5->
```

Las flechas numeradas identifican instantes de tiempo de ejecución del siguiente código. Por ejemplo: 1-> señala el instante inmediatamente después de ejecutar la instrucción `a=10`. Completa la tabla con los valores de las variables en los instantes de tiempo señalados. Ponga XX en los bloques en los que la variable está fuera de su ámbito y -- si el valor de la variable no ha sido inicializado.

	a	b	x	y
1->	10	--	XX	XX
2->				
3->				
4->				
5->				



Para obtener información de una biblioteca en particular, por ejemplo las funciones existentes, vaya al terminal, importe la biblioteca y llame a la función: **help(<nombreBiblioteca>)**

Ejemplo:
import math
help(math)

```
>>> help(math)
Help on module math:

NAME
    math

MODULE REFERENCE
    https://docs.python.org/3.6/library/math

The following documentation is automatically generated from the Python
source files. It may be incomplete, incorrect or include features that
are considered implementation detail and may vary between Python
implementations. When in doubt, consult the module reference at the
location listed above.

DESCRIPTION
    This module is always available. It provides access to the
    mathematical functions defined by the C standard.

FUNCTIONS
    acos(...)
        acos(x)

        Return the arc cosine (measured in radians) of x.
```



Veamos algunas funciones importantes del módulo **math**

math.ceil(x) : retorna el entero más pequeño que sea igual o mayor que x

math.exp(x) : retorna el valor exponencial de x

math.fabs(x) : retorna el valor absoluto de x

math.factorial(x) : retorna el factorial de x

math.floor(x) : retorna el entero más grande que sea igual o menor que x

math.log(x,baseLog) : retorna el logaritmo de x en base baseLog

math.pow(x,y) : retorna x elevado y

math.sqrt(x) : retorna la raíz cuadrada de x



Tiempo : 10 minutos

Cree un programa que le **solicite** al usuario un número z y **un número de iteraciones numIter** por pantalla y calcule la sumatoria:

$$\sum_{k=0}^{numIter} \frac{z^k}{k!}$$



importamos
la librería



math.factorial(x) : retorna el factorial de x:

math.pow(x, y) : retorna x elevado y

```
import math
z = int(input("ingrese numero: "))
numIter = int(input("ingrese numero iteraciones: "))
suma = 0
for k in range(0,numIter+1):
    suma = suma + math.pow(z,k) / math.factorial(k)
print("el valor es ",suma)
```



Veamos algunas funciones importantes de la biblioteca **time**

- `time.sleep(x)` : el programa se pausa por x segundos
- `time.time()` : devuelve el número de segundos desde el 1 de Enero de 1970 (año que se utiliza como valor inicial en Unix)
- `time.ctime(x)` : convierte el tiempo expresado en segundos en el formato de la fecha actual: Día de la semana, Mes, día, hora, año.



Veamos algunas funciones importantes de la biblioteca **random**

- `random.randint (x,y)` : retorna un número aleatorio entero entre x e y.
- `random.random ()` : retorna un número aleatorio entre 0 y 1.
- `random.uniform (x,y)` : retorna un número aleatorio uniforme entre x e y.



Tiempo : 10 minutos

Cree un programa que genere una multiplicación entre dos números en forma aleatoria y le pida al usuario resolverla. Si el usuario se demoró más de 10 segundos en resolverla, despliegue el mensaje “eres muy lento, el resultado era X”. Caso contrario, verifique si el usuario pudo resolver la multiplicación en forma correcta, felicitándolo o mostrando la solución de la multiplicación.

```
import time
import random

time1 = time.time()
num1 = random.randint(1,100)
num2 = random.randint(1,100)
print("¿Cuánto es ",num1," por ",num2,"?",end="")
res = int(input())
time2 = time.time()

if (time2-time1>10):
    print("Eres muy lento, el resultado era ",num1*num2)
else:
    if (num1*num2==res):
        print("FELICITACIONES")
    else:
        print("Te equivocaste, el resultado era ",num1*num2)
```

time.time(): devuelve el número de segundos desde el 1 de Enero de 1970 (año que se utiliza como valor inicial en Unix)

random.randint (x,y): retorna un número aleatorio entero entre x e y.