Funciones

Funciones como **print()**, **input()** y **len()** ya vienen incorporadas con el lenguaje Python. Una función es como un mini-programa dentro de un programa.
Para que se entienda fácilmente crearemos una función desde cero en Python:

```
def hola():
        print('Hola!')
        print('Hola!!!')
        print('Hola mundo!')
```

La primera línea de una función comienza con la palabra reservada **def**, seguido del **nombre** de nuestra función, **paréntesis**, **dos puntos** y una nueva línea con indentación de 4 líneas que delimita el comienzo del **bloque de código** de la función.
Una vez definida nuestra función podemos utilizarla igual que como venimos haciendo con las otras funciones.

```
hola()
hola()
hola()
```

Como en este programa llamamos al a función 3 veces, se ejecutará nuestra función 3 veces.

Definición de Función con Parámetros.

Cuando utilizamos la función **print()** o **len()** , les pasamos valores llamados **argumentos** que van escritos entre los paréntesis de la función. Cuando definimos funciones también podemos hacer que acepten **argumentos** al definir **parámetros** dentro de los paréntesis.

```
def hola(nombre):
   print('Hola ' + nombre)
hola('Juan')
```

Un detalle a tener en cuenta es que, al igual que cuando termina o cerramos un programa, cuando una función termina los valores de sus parámetros se pierden.

Devolver Valor o Sentencia

Cuando llamamos a la función **len()** y le pasamos un **argumento** como 'Hola', la función evalúa la cadena y nos devuelve el valor 5. El valor que devuelve una función puede llamarse **valor de retorno** o **Return Value**.

Cuando creamos una función podemos definir cuál será el valor de retorno utilizando la palabra reservada **return.**

```
import random
def divinacion(numRespuesta):
if numRespuesta == 1:
        return 'Probablemente'
elif numRespuesta == 2:
        return 'Tal vez'
elif numRespuesta == 3:
        return 'Si'
elif numRespuesta == 4:
        return 'Pregunta de nuevo más tarde'
elif numRespuesta == 5:
        return 'No entendí la pregunta'
elif numRespuesta == 6:
        return 'No'
elif numRespuesta == 7:
        return 'Lo dudo mucho'
r = random.randint(1, 7)
fortuna = divinacion(r)
print(fortuna)
```

El Valor Nulo o None

En Python existe un valor llamado None, que representa la ausencia de valores. None es el único valor del tipo NoneType. (Otros lenguajes de programación lo llaman null, nil, o undefined) Al igual que los booleanos True y False, None debe ser tipeado con una N mayúscula.
Cuando creamos una función y no definimos un valor de retorno utilizando **return**, Python automáticamente agrega un *return None* al final de nuestra función. Lo mismo ocurrirá si escribimos return y sin un valor retorno.

Variables Locales y ámbito Globales

Las variables que utilizamos dentro una función se conocen como Variables Locales y no tendrán un alcance global. Ejemplo:

```
def spam():
        huevos = 31337
        spam()
print(huevos)
```

Otros ámbitos Locales no pueden utilizar Variables Locales en otros ámbitos Locales. Ejemplo:

```
def spam():
        huevos = 99
        cerdo()
        print(huevos)
def cerdo():
        jamon = 101
        huevos = 0
spam()
```

Variables Globales

Las variables que utilizamos fuera de las funciones se conocen como Variables Globales. En el caso de necesitar cambiar una variable global, utilizando una función, podremos escribir **global** seguido de la variable global que vayamos a utilizar en la función.

```
def spam():
        global huevos
        huevos = 'spam'
huevos = 'global'
spam()
print(huevos)
```

Manejo de Excepciones o Errores

```
def spam(dividirPor):
        return 42 / dividirPor
print(spam(2))
print(spam(12))
print(spam(0))
print(spam(1))
```

```
def spam(dividirPor):
        try:
                return 42 / dividirPor
        except ZeroDivisionError:
                print('Error: Invalid argument.')

print(spam(2))
print(spam(0))
```