TAREA NO. 3

Funciones en Python.

Una función en general, es un bloque de código que se encarga de realizar una tarea específica y que podemos reutilizarla cuando se desee.

En Python las definimos con la siguiente sintaxis.

```
def nombre_funcion():
    #bloque de código
```

Caso de Uso de una función.

Siempre que se desee aislar una tarea específica que usaremos en varias ocaciones, por ejemplo operaciones aritméticas sobre dos enteros, decimales, etc. (calcular el total de un producto, según su precio y cantidad comprada).

Parámetros posicionales de funciones.

Los parámetros posicionales en una función en Python, se refiere a que el orden de los parámetros siguen el orden en el que se definen en la función, al momento de llamarla.

Por ejemplo.

```
def nombre_funcion(par1, par2):
    print(par1,par2)
```

y luego la llamamos.

```
nombre_funcion(2,1)
```

se imprime como resultado el 2 y el 1.

Caso de uso de parámetros posicionales.

Se utilizan cuando definamos funciones cuyo resultado de retorno el orden no afecta el resultado, por ejemplo la multiplicación de dos números.

Parámetros nombrados.

Los parámetros nombrados en funciones de Python nos permite enviarle estos en cualquier orden, siempre indicando el nombre del parámetro y el valor a enviarle.

Por ejemplo.

```
def nombre_funcion(par1,par2):
    print(par1,par2)
```

y luego llamamos a la función.

```
nombre_funcion(par2=2,par1=1)
```

el resultado será 1,2.

Caso de uso de parámetros nombrados.

Estos parámetros se utilizan cuando queremos asegurar el orden al enviarlos y no afectar el resultado de la función, por ejemplo la división de dos numeros.

Retorno de múltiples valores.

Es posible retornar multiples valores en una función de Python siempre que se necesite.

Por ejemplo.

```
def hora_completa(h,m,s):
    return (h,ms)
```

la función devuelve una n-upla de tamaño adecuado al retorno del resultado.

Caso de uso de retorno múltiple.

Se utiliza cuando se requiere devoler multiples valores, por ejemplo devolver hora, devolver un objeto con datos de una persona, nombre, edad, apellido, etc.

Funciones como objetos y como parámetros de otras funciones.

En Python es posible pasar funciones como parámetros a otras funciones cuando sea necesario.

Por ejemplo definimos una función que calcula la suma de dos enteros.

```
def sumar_enteros(a,b):
    return a + b
```

Ahora definimos otra función que nos permita obtener una tercera suma en base al resultado de la sumar_enteros.

```
def sumar_total(funcion,a,b,c):
    return c + function(a,b)
```

Llamamos a la función.

```
Suma_total(sumar_enteros,1,2)
```

Caso de uso funciones como parámetros.

Por ejemplo si tenemos funciones cuyo resultado de retorno depende del resultado de otras funciones, se hace mas sencillo parar el nombre de la función como parámetro.

Funciones anónimas o lambda.

Son funciones en Python que no tienen nombres, se utilizan para realizar acciones sencillas y especificas.

Por ejemplo. para sumar dos números

```
r = lamda a,b: a+b
```

Nos devuelve como resultado la suma de a y b.

Caso de uso de funcione anónimas.

Se utiliza cuando queremos realizar tareas simples que no requieran de mayor código, por ejemplo, revertir una cadena, suma de dos números, etc.