Creado por:

Isabel Maniega

Funciones

Motivos de uso:

- Algo que no podemos repetir (mucha repetición)
- Requerimos automatizar para no repetir el código muchas veces

In [1]:
```
x = 1
y = x + 3
y
```

Out[1]: 4

In [2]:
```
x = 2
y = x + 3
y
```

Out[2]: 5

In [3]:
```
x = 3
y = x + 3
y
```

Out[3]: 6

Código repetido...

declaramos una función:

In [4]:
```
def suma(x):
    # print(x  + 3)
    return x + 3

resultado = suma(1)
resultado
```

Out[4]: 4

In [5]:
```
resultado = suma(2)
resultado
```

Out[5]: 5

In [6]:
```
resultado = suma(3)
resultado
```

Out[6]: 6

Realizamos un bucle for para automatizarlo

In [7]:
```
for x in range(1, 6):
    print("valor de x:", x)
    print(suma(x))
```

```
valor de x: 1
4
valor de x: 2
5
valor de x: 3
6
valor de x: 4
7
valor de x: 5
8
```

Función: lambda

In [8]:
```
def funcion(x):
    return x + 1
```

In [9]:
```
funcion(1)
```

Out[9]: 2

In [10]:
```
(lambda x: x + 1)(3)
```

Out[10]: 4

In [11]:
```
f = lambda x: x + 1
f(3)
```

Out[11]: 4

Función: Creación y llamada

In [12]:
```
def funcion():
    print("Hola mundo")
```

In [13]:
```
funcion()
```

```
Hola mundo
```

In [14]:
```
def funcion():
    return "Hola Mundo"
```

In [15]:
```
funcion()
```

Out[15]: 'Hola Mundo'

Función que recibe 2 variables y retorne 2 variables

In [16]:
```
def variasOpciones(x, y):
    suma = x + y
    producto = x * y
    return suma, producto
```

In [17]:
```
variasOpciones(3, 2)
```

Out[17]: (5, 6)

In [18]:
```
SUMA, PRODUCTO = variasOpciones(3, 2)
SUMA
```

Out[18]: 5

In [19]:
```
PRODUCTO
```

Out[19]: 6

In [20]:
```
SUMA, PRODUCTO
```

Out[20]: (5, 6)

In [ ]:
```
# OJO con el orden que retornamos las variables
# return suma, producto
# al mostrar la variable deben seguir el mismo orden suma, producto = var
```

In [21]:
```
def variasOpciones(x, y):
    suma = x + y
    producto = x * y
    return producto, suma
```

In [22]:
```
SUMA, PRODUCTO = variasOpciones(3, 2)
SUMA, PRODUCTO
```

Out[22]: (6, 5)

Error al mostrar la información!!!

Funciones Recursivas

Es una técnica donde una función se invoca a sí misma.

La serie fibonnacci es un claro ejemplo de recursividad:

Fib i = Fib i-1 + Fib i-2

El número i se refiere al número de i -1, y asi sucesivamente hasta llegar a los primeros dos.

Se puede crear una funcion Fib() para usar la recursividad:

In [23]:
```
def fib(n):
    if n < 1:
```

```
        return None
    if n < 3:
        return 1
    return fib(n-1) + fib (n-2)
```

Este programa necesita de una condición que detenga el bucle infinito, esto ocasiona un consumo alto en memoria y por lo tanto pueden ser en ocasiones ineficientes.

Variables Locales y Globales

In [25]:
```
# Podemos cambiar el valor de una variable
```

In [26]:
```
x = 6
print(x)
```

```
6
```

In [27]:
```
# volvemos a definir el valor de x, el valor de x pasa a valer 5
x = 5
print(x)
```

```
5
```

In [28]:
```
def funcion_cambiar_x():
    x = 6
    #print(x)
    return x
```

In [29]:
```
funcion_cambiar_x()
```

Out[29]: 6

In [30]:
```
print(x)
```

```
5
```

In [31]:
```
y = 6
print(y)
y = 5
print(y)

def cambiar_y():
    y = 3
    return y

print(cambiar_y())

print(y)
```

```
6
5
3
5
```

In [32]:
```
def funcion_cambiar_x():
    global x
    x = 6
```

```
    # print(x)
    return x
```

In [33]:
```
funcion_cambiar_x()
```

Out[33]: 6

In [34]:
```
print(x)
```

```
6
```

In [39]:
```
y = 6
print(y)
y = 5
print(y)

def cambiar_y():
    global y
    y = 3
    return y

print(cambiar_y())

print(y)
```

```
6
5
3
3
```

Break, continue, pass - For -

- **BREAK**

In [4]:
```
L = [5, 10, 15, 20, 25, 30, 35]
L
```

Out[4]: [5, 10, 15, 20, 25, 30, 35]

In [5]:
```
for numero in L:
    if numero == 20:
        print("\n")
        break
    else:
        print(numero) # mostrar: 5, 10, 15
print("hemos llegado al 20, y salida del bucle FOR")
```

```
5
10
15


hemos llegado al 20, y salida del bucle FOR
```

- **CONTINUE**

In [6]:
```
L = [5, 10, 15, 20, 25, 30, 35]
L
for numero in L:
    if numero == 20:
        print("hemos llegado al valor 20, y CONTINUO (SIN IMPRIMIRLE)")
        continue
    else:
        print(numero) # mostrar: 5, 10, 15, 25, 30, 35
```

```
5
10
15
hemos llegado al valor 20, y CONTINUO (SIN IMPRIMIRLE)
25
30
35
```

- **PASS**

In [14]:
```
def funcion():
    # TODO: funcion de suma de variables
    pass
    # pendiente de describir la actividad de la función
funcion()
```

In [10]:
```
L = [5, 10, 15, 20, 25, 30, 35]
L
for numero in L:
    if numero == 20:
        print("hemos llegado al valor 20, y CONTINUO (SIN IMPRIMIRLE)")
        pass
    else:
        print(numero) # mostrar: 5, 10, 15, 25, 30, 35
```

```
5
10
15
hemos llegado al valor 20, y CONTINUO (SIN IMPRIMIRLE)
25
30
35
```

Menús

In [15]:
```
L = []

def insertar(elemento):
    L.append(elemento)

def eliminar():
    L.remove(L[-1])

def consultar():
    print("\n")
    print("Los numeros que tiene en este momento son: ")
    print(L)
    print("\n")
```

```
while True:
    print("\n")
    print("**************** MENU ******************")
    print("****************************************")
    print("*****  1. Insertar (nuevo elemento) *****")
    print("*****  2. Eliminar (último elemento) ****")
    print("*****  3. Consultar (toda la lista)  ****")
    print("*****  99. Salir (del menú)        ******")
    print("****************************************")

    print("\n")

    opcion = int(input("Inserte su opción: "))

    if opcion == 1:
        # recoger el valor a insertar con elemento
        elemento = input("Inserte el nuevo número: ")
        # ir a la funcion insertar
        insertar(elemento)
    elif opcion == 2:
        if len(L) != 0:
            # ir a la funcion eliminar
            eliminar()
        else:
            print("\n")
            print("no tiene elementos para eliminar")
            print("\n")
    elif opcion == 3:
        # ir a la funcion consultar
        consultar()
    elif opcion == 99:
        break
    else:
        print("por favor, escriba una opción correcta. ")
        print("\n")
```

```
**************** MENU ******************
****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
****************************************
```

```
**************** MENU ******************
****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
****************************************
```

```
Los numeros que tiene en este momento son:
['2']
```

```
*************** MENU ******************
*****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
*****************************************
```

```
*************** MENU ******************
*****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
*****************************************
```

```
Los numeros que tiene en este momento son:
['2', '3']
```

```
*************** MENU ******************
*****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
*****************************************
```

```
*************** MENU ******************
*****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
*****************************************
```

```
Los numeros que tiene en este momento son:
['2']



**************** MENU ******************
****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
****************************************



**************** MENU ******************
****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
****************************************


Los numeros que tiene en este momento son:
[]



**************** MENU ******************
****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
****************************************


no tiene elementos para eliminar



**************** MENU ******************
****************************************
*****  1. Insertar (nuevo elemento) *****
*****  2. Eliminar (último elemento) ****
*****  3. Consultar (toda la lista)  ****
*****  99. Salir (del menú)        ******
****************************************
```

Main en python

In [ ]:
```
# sirve para simular el int main() de otros lenguajes de programación
# un ejemplo de Main en otros lenguajes como C, sería el siguiente:
```

In [16]:
```
from IPython.display import Image
Image("ejemploMainC.png")
```

Out[16]:
```
//FACTORIAL CON FUNCIONES

#include <stdio.h>
//Variables globales
//deben ser reconocidas en ambas funciones
int num;
int y=0; //simplemente recoge el valor de la variable
int factorial=1;

int main()
{
    printf("\nIntroduzca un numero y le diré su factorial:\n");
    scanf("%d", &num);
    y=num;

    //AQUI LLAMA A LA FUNCION CREADA
    funcion_factorial();
    //UNA VEZ AQUI EJECUTA SIN MAS
    printf("El factorial de %d es %d\n", y, factorial);
}

int funcion_factorial()
{
    for(num=num;num>=1;num--)
    {
        factorial=factorial*num;
    }
}
```

Ejemplo 1

In [17]:
```
def main():
    print("Estamos en la funcion main()")

if __name__ == "__main__":
    main()
```

```
Estamos en la funcion main()
```

Ejemplo 2

In [18]:
```
def cuadrado(x):
    return x * x

def main():
    print("Estamos en la funcion main()")

def funcion():
    print("test")

if __name__ == "__main__":
```

```
    print(cuadrado(8))
    main()
```

```
64
Estamos en la funcion main()
```

Try - Except

In [ ]:
```
# Ejecutamos todo el código de una sola pasada
# para ver como funciona el except

# Sirve para testear errores en el código

# de tal manera que no para todo el programa al detectar un error
```

In [ ]:
```
# Asumimos que teniamos:
# una variable "x" que apareció anteriormente
# una variable "w" que no apareció previamente (NO DECLARADA)
```

In [19]:
```
x = [10, 20, 30, 40]
x
```

Out[19]:
```
[10, 20, 30, 40]
```

In [20]:
```
try:
    print(s)
except Exception as e:
    print("Error: %s" % str(e))
    print(type(e))
```

```
Error: name 's' is not defined
<class 'NameError'>
```

In [21]:
```
# Excepciones según error
try:
    print(s)
except NameError:
    print("error en el nombre no definido")
except Exception as e:
    print("Error: %s" % str(e))
    print(type(e))
```

```
error en el nombre no definido
```

In [22]:
```
w = 25
```

In [23]:
```
try:
    print(w)
except Exception as e:
    print("Error: %s" % str(e))
```

```
25
```

In [24]:
```
try:
    print(x)
except Exception as e:
    print("Error: %s" % str(e))
```

```
[10, 20, 30, 40]
```

Creado por:

Isabel Maniega