Desarrollo guiado por pruebas

Autor: Victor Simo Lozano

Actividad 2

En base al archivo .py de la actividad anterior, ejecutar una serie de test a las funciones del mismo.

Nota: *Para la realización de los test, se ha modificado el archivo original de modo que este incluya las funciones necesarias para realizar la práctica.*

CONTENIDO

Implementar un conjunto de test unitarios para validar las funciones que implementó en la actividad anterior.
Para desarrollar este conjunto de tests puede hacer uso de la herramienta unittest o pytest.

SOLUCION

1.-

Importamos la libreria para los test asi como las funciones de la práctica 1 a evaluar.

In [1]:
```
import os
import sys

funcPath = os.path.abspath(os.path.join('..'))
if funcPath not in sys.path:
    sys.path.append(funcPath)
```

In [2]:
```
import unittest

# Try en la importacion al tratarse de un modulo susceptible de errores en .csv que este trabaja.
try:
```

```
    from Actividad_Control_Gastos import MayorGasto, MayorAhorro, \
        CheckDataLength, CheckEmpty, CheckValues
except Exception as e:
    print(f'Import Error: {e}')
```

```
COMPROBACIONES:
En curso...
CheckDataLength...
... finished.

CheckEmpty...
... finished.

CheckValues...
Atención! El valor "'-541'" del mes de Enero no se ha convertido a valor numerico.
Introduzca un nuevo valor o 0 si lo descarta: -541


Atención! El valor "'-602'" del mes de Julio no se ha convertido a valor numerico.
Introduzca un nuevo valor o 0 si lo descarta: -602


Atención! El valor "bug" del mes de Septiembre no se ha convertido a valor numerico.
Introduzca un nuevo valor o 0 si lo descarta: 0


Atención! El valor "ups" del mes de Octubre no se ha convertido a valor numerico.
Introduzca un nuevo valor o 0 si lo descarta: 0


Atención! El valor "'00'" del mes de Noviembre no se ha convertido a valor numerico.
Introduzca un nuevo valor o 0 si lo descarta: 0


... finished.

... COMPROBACIONES DONE.
```

Nota:

La libreria escogida para el desarrollo de test es "unittest" como se observa.

*Debido a que el modulo de la actividad 1 ejecuta una serie de checks debido a que es susceptible al contenido del .csv que este emplea, durante su importacion se observa dichos checks.**No se trata de ninguna clase de test***

2.-

Creacion de la clase test y definicion de estos.

In [3]:
```
# De acuerdo con documentación de unittest, creacion de la clase de test que hereda de unittest.TestCase

# Dada la posibilidad de error de importación, ejecutamos un try en cada evaluacion de test
# para evitar una interrupción en nuestro código si nuestra funcion no existe mediante la excepción "NameError".
# En el caso de excepción, decimos al usuario que hay un error y hacemos fallar nuestro test.


class TestClass(unittest.TestCase):
    def test_Gasto(self):
        ''' Test para comprobar que el mes de mayor gasto.'''
        try:
            # Evaluacion de igualdad entre nuestra función y el resultado que esperamos "Abril"
            self.assertEqual(MayorGasto(), 'Abril')
        except NameError:
            print('Error testing function.')
            self.fail()

    def test_Ahorro(self):
        '''Test para comprobar el mes de mayor ahorro.'''
        try:
            # Evualuacion de igualdad entre nuestra función y el resultado que esperamos "Enero"
            self.assertEqual(MayorAhorro(), 'Enero')
        except NameError:
            print('Error testing function.')
            self.fail()

    def test_CheckLength(self):
        '''Test para comprobar el numero de columnas de nuestro set de datos.'''
        try:
            # Evaluación de igualdad entre el numero de columnas de nuestro set y el numero de meses.
            self.assertEqual(CheckDataLength(), 12)
        except NameError:
            print('Error testing function.')
            self.fail()

    def test_CheckEmpty(self):
        '''Test para comprobar que todas las columnas poseen datos.'''
        try:
            # Evaluación de la diferencia entre nuestra función y el resultado esperado. Si nuestra función no retorna
```

```
            # "Sucess" saltará una excepción
            self.assertFalse(CheckEmpty() != 'Success')
        except NameError:
            print('Error testing function.')
            self.fail()

    def test_Values(self):
        try:
            self.assertFalse(CheckValues() != 'Success')
        except NameError:
            print('Error testing function.')
            self.fail()


unittest.main(argv=[''], verbosity=2, exit=False)
```

```
test_Ahorro (__main__.TestClass)
Test para comprobar el mes de mayor ahorro. ... ok
test_CheckEmpty (__main__.TestClass)
Test para comprobar que todas las columnas poseen datos. ... ok
test_CheckLength (__main__.TestClass)
Test para comprobar el numero de columnas de nuestro set de datos. ... ok
test_Gasto (__main__.TestClass)
Test para comprobar que el mes de mayor gasto. ... ok
test_Values (__main__.TestClass) ... ok

----------------------------------------------------------------------
Ran 5 tests in 0.082s

OK
```

Out[3]:
```
<unittest.main.TestProgram at 0x221f6f6d610>
```