EXAMEN TEST UNITAIRE PYTHON
ribeiro benjamin

Exercice 1 :

Voici la function nommer nb_premier qui va être utilisé pour les test (j'ai juste renommer car c'est plus compréhensible pour moi) :

f1.py > nb_premier

```python
import math

def nb_premier(nombre):
    if nombre < 2:
        return False
    for i in range(2, int(math.sqrt(nombre)) + 1):
        if nombre % i == 0:
            return False
    return True
```

(Voir les tests dans le repo git) J'ai mis en place plusieurs tests par exemple sur les nombre négatifs , non premier , grands nombre , 0 et 1 .

Voici les résultats :

```
PS C:\Examen>  c:; cd 'c:\Examen'; & 'c:\Users\DÉVELOPPEUR\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\DÉVELOPPEUR\.vscode\extensions\ms-python.debugpy-2024.0.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '53780' '--' 'C:\Examen\test_f1.py'
Résultat test grands nombres non premiers :  False
.Résultat test grands nombres premiers :  True
.Résultat test nombres négatifs :  False
Résultat test nombres négatifs :  False
Résultat test nombres négatifs :  False
.Résultat test nombres non premiers :  False
Résultat test nombres non premiers :  False
Résultat test nombres non premiers :  False
Résultat test nombres non premiers :  False
Résultat test nombres non premiers :  False
Résultat test nombres non premiers :  False
.Résultat test nombres premiers :  True
Résultat test nombres premiers :  True
Résultat test nombres premiers :  True
Résultat test nombres premiers :  True
Résultat test nombres premiers :  True
.Résultat test zéro et un :  False
Résultat test zéro et un :  False
.
----------------------------------------------------------------------
Ran 6 tests in 0.007s

OK
```

On peut constater que tout les tests passe et que ceux qui ne sont pas premier sont False et ceux qui le sont en True .

Exercice 2 :

Voici la fonction compter_nombre :

```python
def compter_mots(phrase):
    # phrase vide
    if not phrase:
        return 0


    mots = phrase.strip().split()

    #  nombre de mots
    return len(mots)
```

(Voir repo pour test) J'ai mis un test avec une phrase sans mots , avec 1 mot , 3 mots et 4mots avec un caractère spécial .

Résultat :

```
Test espaces supplémentaires :
Résultat attendu : 3
Résultat obtenu : 3
.Test mots multiples :
Résultat attendu : 4
Résultat obtenu : 4
.Test phrase vide :
Résultat attendu : 0
Résultat obtenu : 0
.Test une seule mot :
Résultat attendu : 1
Résultat obtenu : 1
.
----------------------------------------
Ran 4 tests in 0.005s

OK
PS C:\Examen>
```

Tout les tests passe bien .

Exercice 3 :

Voici la fonction compte_bancaire :

```python
class CompteBancaire:
    def __init__(self, solde_initial):
        self.solde = solde_initial
    def deposer(self, montant):
        self.solde += montant
    def retirer(self, montant):
        if self.solde < montant:
            raise ValueError("Solde insuffisant")
        self.solde -= montant
    def obtenir_solde(self):
        return self.solde
```

(Voir le repo pour les test et la fonction) J'ai ajouter donc les test demander , dépôt , retirer et voir le solde .
Dans le test dépot j'ai initialiser 100 euros dans le compte bancaire et l'on fait un dépôt de 50 ce qui donne 150 .
Dans le test retirer plus ou moins la meme chose mais l'on retire la somme
et dans obtenir le solde j'affiche juste le solde avec self.solde

Résultat :

```
PS C:\Examen> c:; cd 'c:\Examen'; & 'C:\Users\DEVELOPPEUR\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\DEVELOPPEUR\.vscode\extensions\ms-python.debugpy-2024.0.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '54260' '--' 'C:\Examen\f3\test_compte_bancaire.py'
Solde attendu après dépôt : 150
Solde obtenu après dépôt : 150
.Solde attendu : 100
Solde obtenu : 100
.Solde attendu après retrait : 50
Solde obtenu après retrait : 50
.
----------------------------------------------------------------------
Ran 3 tests in 0.003s

OK
PS C:\Examen>
```

Exercice 4 :

Voici la fonction somme_liste :

```python
def somme_liste(liste):
    somme = 0
    for element in liste:
        somme += element
    return somme
```

Pour les test j'ai tester une liste vide , une liste de négatifs et une liste de positifs.

Résultat :

```
PS C:\Examen>  c:; cd 'c:\Examen'; & 'c:\Users\DÉVELOPPEUR\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\DÉVELOPPEUR\.vscode\extensions\ms-python.debugpy-2024.0.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '54408' '--' 'C:\Examen\f4\test_somme_liste.py'
Résultat attendu pour une liste de négatifs : -15
Résultat obtenu pour une liste de négatifs : -15
.Résultat attendu pour une liste de positifs : 15
Résultat obtenu pour une liste de positifs : 15
.Résultat attendu pour une liste vide : 0
Résultat obtenu pour une liste vide : 0
.
----------------------------------------------------------------------
Ran 3 tests in 0.005s

OK
```

Exercice 5 :

Voici la fonction Rectangle :

```python
class Rectangle:
    def __init__(self, longueur, largeur):
        self.longueur = longueur
        self.largeur = largeur
    def calculer_perimetre(self):
        return 2 * (self.longueur + self.largeur)
    def calculer_surface(self):
        return self.longueur * self.largeur
```

Pour les test j'ai calculer la surface et le périmètre

Résultat :

```
PS C:\Examen>  c:; cd 'c:\Examen'; & 'c:\Users\DÉVELOPPEUR\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\DÉVELOPPEUR\.vscode\extensions\ms-python.debugpy-2024.0.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '54520' '--' 'C:\Examen\f5\test_rectangle.py'
Périmètre attendu : 30
Périmètre obtenu : 30
.Surface attendue : 50
Surface obtenue : 50
.
----------------------------------------------------------------------
Ran 2 tests in 0.003s

OK
```

Exercice 6 :

Voici la function est_palindrome (que j'ai un peu modifié pour ajouter les espaces) :

```python
def est_palindrome(chaine):
    chaine = chaine.replace(" ", "").lower()
    chaine_inverse = chaine[::-1]
    return chaine == chaine_inverse
```

Pour les test j'ai fait un test avec des espaces vide , des mots , palindrome et non palindrome .

Résultat :

```
Test non palindrome :
Résultat attendu pour 'hello' : False
Résultat obtenu : False
Résultat attendu pour 'world' : False
Résultat obtenu : False
.Test palindrome avec espaces :
Résultat attendu pour 'a man a plan a canal panama' : True
Résultat obtenu : True
Résultat attendu pour 'was it a car or a cat I saw' : True
Résultat obtenu : True
.Test palindrome simple :
Résultat attendu pour 'radar' : True
Résultat obtenu : True
Résultat attendu pour 'level' : True
Résultat obtenu : True
.
----------------------------------------------------------------------
Ran 3 tests in 0.005s

OK
```

Exercice 7 :
Voici la function calculer_moyenne :

```python
def calculer_moyenne(liste):
    if not liste:
        raise ValueError("La liste ne peut pas être vide")
    return sum(liste) / len(liste)
```

Pour les tests j'ai tester une liste vide et non vide :

Résultat :

```
n.debugpy-2024.0.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '54722' '--' 'C:\Examen\f7\test_calculer_moyenne.py'
Résultat attendu pour une liste non vide : 3.0
Résultat obtenu pour une liste non vide : 3.0
.Résultat attendu pour une liste vide : ValueError
Résultat obtenu pour une liste vide : ValueError
.
----------------------------------------------------------------------
Ran 2 tests in 0.004s

OK
```