TP 1 – TESTS

Info1.Algo1 - 2022-2023 Semestre Impair

Exercice 1

Dans le fichier **ex01__palindrome.py** est proposée une fonction et la fonction de test correspondante :

- la fonction `est_palindrome` accepte en paramètre une chaîne de caractère et retourne `True` si celle-ci est un palindrome, `False` sinon.
- la fonction `test_palindrome` est censée tester la fonction `est_palindrome`.

Cependant **la fonction `est_palindrome` est mal conçue** et la fonction de test ne permet pas de révéler cette erreur.

1) Quelle situation a été oubliée dans `test_palindrome` ? L'ajouter à la fonction de test. Que constatez-vous.

2) Corriger la fonction `est_palindrome` pour qu'elle fonctionne sur tous les cas de tests.

Exercice 2

Dans le fichier **ex02__maximum.py** sont proposées deux fonctions :

- La fonction `maximum` accepte en paramètre une liste non vide contenant des éléments de type `int` et est censée retourner le maximum de la liste.
- La fonction `test_maximum` est censée tester la fonction `maximum`.

Cependant **la fonction `test_maximum` est mal conçue** et il manque au moins un test qui aurait permis de révéler une **erreur présente dans la fonction `maximum`**.

1) Quelle situation a été oubliée dans `test_maximum`? L'ajouter à la fonction de test.

2) Corriger la fonction `maximum` de façon à satisfaire le dernier test ajouté.

Contrainte : effectuer la correction de façon minimale en ne modifiant qu'une seule ligne.

Exercice 3

Dans le fichier **ex03_doublons.py** sont proposées deux fonctions :

- La fonction `eliminer_doublons` accepte en paramètre une liste contenant des entiers et est censée retourner cette même liste dans laquelle toute séquence de deux (ou plus) entiers identiques consécutifs est remplacée par un seul.
- La fonction `test_eliminer_doublons` est censée tester la fonction `eliminer_doublons`.

Cependant **la fonction `test_eliminer_doublons` est mal conçue** et il manque au moins un test qui aurait permis de révéler une **erreur présente dans la fonction `eliminer_doublons`**.

1) Quelle situation a été oubliée dans `test_eliminer_doublons`? L'ajouter à la fonction de test.

2) Corriger la fonction `eliminer_doublons` de façon à satisfaire le dernier test ajouté.

Contrainte : effectuer la correction de façon minimale en ne modifiant que la structure de contrôle, sans ajouter de ligne de calcul.

Exercice 4

Dans le fichier **ex04_facteurs_premiers.py** sont proposées deux fonctions :

- La fonction `facteurs_premiers` accepte en paramètre un entier `n` strictement positif et est censée retourner une liste contenant les facteurs premiers de `n`.

En particulier la fonction s'engage à retourner une liste qui ne contient que des nombres premiers, classés par ordre croissant, et dont le produit est égal à la valeur de son paramètre d'appel. On considèrera le produit des éléments d'une liste vide comme étant égal à 1.

- La fonction `test_facteurs_premiers` est censée tester la fonction `facteurs_premiers`.

Cependant **la fonction `test_facteurs_premiers` est mal conçue** et il manque au moins un test qui aurait permis de révéler une **erreur présente dans la fonction `facteurs_premiers`**.

1) a) En appelant la fonction `facteurs_premiers` sur différents entiers, déterminer quelle situation a été oubliée.

b) Ajouter cette situation à la fonction `test_facteurs_premiers`

2) Corriger la fonction `facteurs_premiers` de façon à satisfaire le dernier test ajouté.

Contrainte : effectuer la correction de façon minimale en ne modifiant que la structure de contrôle, sans ajouter de ligne de calcul.

Exercice 5

Dans le fichier **ex05_occurrences.py** est fournie la fonction `nb_occurrences` qui accepte en paramètres :

- une liste d'entiers.
- un valeur entière.

et retourne le nombre de fois où la valeur apparaît dans la liste.

Compléter sa fonction de test associée `test_nb_occurrences` avec chacun des 9 cas suivants :

- liste vide.
- liste à un seul élément égal à la valeur recherchée.
- liste à un seul élément distinct de la valeur recherchée.
- liste de longueur $n \geq 5$ ne contenant pas la valeur recherchée.
- liste de longueur $n \geq 5$ contenant une seule fois la valeur recherchée en début.
- liste de longueur $n \geq 5$ contenant une seule fois la valeur recherchée en fin.
- liste de longueur $n \geq 5$ contenant une seule fois la valeur recherchée ni en début ni en fin.
- liste de longueur $n \geq 5$ contenant plusieurs fois la valeur recherchée (mais aussi d'autres valeurs).
- liste de longueur $n \geq 5$ ne contenant que la valeur recherchée.

La fonction `nb_occurrences`, en plus d'effectuer le calcul souhaité, analyse le cas qui lui est présentée et réalise un affichage différent pour chacun des 9 cas que l'on doit tester. Lorsque vous exécutez le fichier python, vous devez alors obtenir un affichage comparable au suivant :

```
Test de la fonction nb_occurrences
|X| | | | | | | |
| |X| | | | | | |
| | |X| | | | | |
| | | |X| | | | |
| | | | |X| | | |
| | | | | |X| | |
| | | | | | |X| |
| | | | | | | |X|
| | | | | | | | |X|
  OK
```

- Si le `OK` final n'est pas affiché, c'est que l'un des tests effectué est incorrect

- Il faut obtenir une croix dans chacune des 9 colonnes, peu importe dans quel ordre.

  Attention ! Il n'est pas demandé de réécrire la fonction `nb_occurrences`.

Exercice 6

On souhaite tester la fonction `est_bissextile` qui doit déterminer si une année **(entière et positive)** donnée en paramètre est bissextile et retourner le booléen correspondant.

On rappelle qu'une année est bissextile si elle est divisible par 4 mais pas par 100, ou bien si elle est divisible par 400.

1) Quels sont les 4 cas que l'on doit tester ? On utilisera en particulier les critères de divisibilité par 4 et 100.

2) Dans le fichier **ex06_annee_bissextile.py**, compléter le corps de la fonction de test `test_est_bissextile` avec ces 4 cas.

Dans ce fichier, on vous donne une fonction `est_bissextile` qui a été volontairement rendue illisible *(on parle alors d'obfuscation du code)* mais qui est correcte et indique par un affichage que les cas à tester…ont bien été testés. Lorsque vous exécutez le fichier python, vous devez alors obtenir un affichage comparable au suivant :

```
Test de la fonction est_bissextile
|X| | | |
| |X| | |
| | |X| |
| | | |X|
  OK
```

- Si le `OK` final n'est pas affiché, c'est que l'un des tests effectué est incorrect
- Il faut obtenir une croix dans chacune des colonnes, peu importe dans quel ordre.

  Attention ! Il n'est pas demandé de réécrire la fonction `est_bissextile`.

Exercice 7

Dans le fichier **ex07_mystere.py**, compléter la fonction `mystere` qui accepte en paramètres :

- une liste d'entiers.
- un entier quelconque.

et retourne une liste d'entiers.

Contraintes :

- La fonction doit évidemment valider la fonction de test associée.
- Toute réponse non générique est interdite et le nombre de `if` doit être minimisé. Un seul `if` pour traiter un cas très dégradé reste acceptable mais il est possible d'écrire une solution qui n'en utilise…aucun !