2026 sujet 5

Question 1

```
def total_simple(empreinte):
    """Fonction qui renvoie l'empreinte carbone totale d'un dictionnaire associant
    une empreinte carbone à des noms de catégories"""
    total = 0
    for cle, valeur in empreinte.items():
        total += valeur
    return total

def test_total_simple():
    # On récupère le dictionnaire à partir du JSON
    empreinte = chargement_json("empreinte_ada_agr.json")
    assert total_simple(empreinte) == 7252
```

Question 2

Si `empreinte` n'est pas un dictionnaire c'est une valeur numérique et alors on la renvoie. Sinon, il faut rappeler récursivement la fonction sur les valeurs du dictionnaire :

```
def total_rec(empreinte):
    """Fonction récursive qui renvoie l'empreinte carbone totale représentée
    par un dictionnaire dont les valeurs peuvent aussi être des dictionnaires"""
    total = 0
    if est_dictionnaire(empreinte):
        for cle, valeur in empreinte.items():
            total = total + total_rec(valeur)
        return total
    else:
        return empreinte
```

Pour les tests, on peut simplement ajouter le test avec les valeurs réelles :

```
def test_total_rec():
    test_dico1 = {"a": 1, "d": 2}
    assert total_rec(test_dico1) == 3
    test_dico2 = {"a": {"b": 1, "c": 2}, "d": {"e": 3}}
    assert total_rec(test_dico2) == 6
    # On récupère le dictionnaire à partir du JSON
    empreinte = chargement_json("empreinte_ada.json")
    assert total_rec(empreinte) == 7252
```

Question 3

La fonction fonctionne correctement sur l'exemple!

```
>>> empreinte = chargement_json("empreinte_ada.json")
>>> alerte_valeur_aberrante(empreinte, 1000).
True
```

Par contre, s'il y a des dictionnaires imbriqués, la fonction va renvoyer `False` dès le premier dictionnaire sans valeur supérieure à la limite à cause du `return False` à la fin :

```
>>> alerte_valeur_aberrante({"a": {"b": 1, "c": 2}, "d": {"e": 3}}, 2)
False
```

Une solution est de compter récursivement les valeurs aberrantes pour éviter les problèlmes de return et de renvoyer `True` ou `False` en fonction de leur nombre :

```
def alerte_valeur_aberrante_aux(empreinte, limite):
    """
    Fonction qui compte récursivement les
    valeurs aberrantes
    """
    nb = 0
    for categorie, valeur in empreinte.items():
        if est_dictionnaire(valeur):
            nb += alerte_valeur_aberrante(valeur, limite)
        else:
            if valeur > limite:
                nb += 1
    return nb

def alerte_valeur_aberrante(empreinte, limite):
    """
    Fonction qui détermine si au moins une valeur du dictionnaire
    dépasse strictement la limite donnée.
    """
    if alerte_valeur_aberrante_aux(empreinte, limite) > 0:
        return True
    else:
        return False
```

Question 4

Voici les tests :

```
# Tests
# Dictionnaire simple cas positif
test_dico1 = {"a": 100, "d": 1000}
assert alerte_valeur_aberrante(test_dico1, 500) == True
# Dictionnaire simple cas negatif à la limte
assert alerte_valeur_aberrante(test_dico1, 1000) == False
# Dictionnaire vide
assert alerte_valeur_aberrante({}, 1000) == False
# Dictionnaires imbriqués
test_dico2 = {"a": {"b": 1, "c": 2}, "d": {"e": 3}}
assert alerte_valeur_aberrante(test_dico2, 2) == True
```