Tests « boîte noire »

Corrigé

1. Exercice : tests « boîte noire »

Je vais vous proposer un exercice en Python sur les tests de type "boîte noire" avec une correction utilisant pytest. Le test "boîte noire" consiste à vérifier le bon fonctionnement d'une fonction sans avoir accès à son code interne, en se basant uniquement sur les spécifications.

Exercice :

Vous devez écrire un ensemble de tests pour la fonction définie ci-dessous.

Il faut vérifier le bon fonctionnement d'une fonction sans avoir accès à son code interne.

Contraintes :

1. Les espaces et la casse ne doivent pas être pris en compte (par exemple, "A Santa at NASA" doit être considéré comme un palindrome).
2. Les caractères spéciaux doivent être ignorés (par exemple, "No 'x' in Nixon" doit aussi être considéré comme un palindrome).

Voici les spécifications de la fonction :

```
def is_palindrome(s: str) -> bool:
    """
    Vérifie si la chaîne de caractères s est un palindrome.

    :param s: Chaîne de caractères à vérifier
    :return: True si s est un palindrome, False sinon
    """
```

Les tests à effectuer :

- **test_empty_string** : vérifie que la fonction retourne True pour une chaîne vide, car une chaîne vide est considérée comme un palindrome.
- **test_single_character** : un seul caractère est toujours un palindrome.
- **test_simple_palindrome** : vérifie un cas simple où la chaîne est un palindrome.
- **test_non_palindrome** : teste une chaîne qui n'est pas un palindrome.
- **test_palindrome_with_spaces** : vérifie que la fonction ignore les espaces et la casse.
- **test_palindrome_with_special_characters** : s'assure que les caractères spéciaux sont ignorés.
- **test_mixed_case_palindrome** : teste que la fonction ignore la différence entre majuscules et minuscules.
- **test_long_non_palindrome** : un cas pour vérifier qu'une longue chaîne non-palindrome est bien détectée.

Fichier de test : test_is_palindrome.py

```
import pytest

from palindrome import is_palindrome  # On suppose que la fonction est définie dans le fichier palindrome.py


def test_empty_string():

  ...
```

Correction de la fonction :

Voici un exemple de fonction is_palindrome qui répond aux spécifications :

```
import re
def is_palindrome(s: str) -> bool:
    clean_s = re.sub(r'[^a-zA-Z0-9]', '', s).lower()
    # On vérifie si la chaîne nettoyée est égale à elle-même inversée
    return clean_s == clean_s[::-1]
```

Correction avec pytest

```
import pytest
from palindrome import is_palindrome  # On suppose que la fonction est définie dans le fichier palindrome.py

def test_empty_string():
    assert is_palindrome("") == True, "Une chaîne vide est un palindrome"

def test_single_character():
    assert is_palindrome("a") == True, "Un seul caractère est un palindrome"

def test_simple_palindrome():
    assert is_palindrome("madam") == True, "'madam' est un palindrome"

def test_non_palindrome():
    assert is_palindrome("hello") == False, "'hello' n'est pas un palindrome"

def test_palindrome_with_spaces():
    assert is_palindrome("A Santa at NASA") == True, "'A Santa at NASA' est un palindrome, sans tenir compte des espaces et de la casse"

def test_palindrome_with_special_characters():
    assert is_palindrome("No 'x' in Nixon") == True, "'No 'x' in Nixon' est un palindrome avec des caractères spéciaux"

def test_mixed_case_palindrome():
    assert is_palindrome("RaceCar") == True, "'RaceCar' est un palindrome malgré les majuscules/minuscules"

def test_long_non_palindrome():
    assert is_palindrome("This is not a palindrome") == False, "'This is not a palindrome' n'est pas un palindrome"

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

Explication des tests :

- **test_empty_string** : vérifie que la fonction retourne True pour une chaîne vide, car une chaîne vide est considérée comme un palindrome.
- **test_single_character** : un seul caractère est toujours un palindrome.
- **test_simple_palindrome** : vérifie un cas simple où la chaîne est un palindrome.
- **test_non_palindrome** : teste une chaîne qui n'est pas un palindrome.
- **test_palindrome_with_spaces** : vérifie que la fonction ignore les espaces et la casse.
- **test_palindrome_with_special_characters** : s'assure que les caractères spéciaux sont ignorés.
- **test_mixed_case_palindrome** : teste que la fonction ignore la différence entre majuscules et minuscules.
- **test_long_non_palindrome** : un cas pour vérifier qu'une longue chaîne non-palindrome est bien détectée.