BÀI TẬP KIỂM THỬ MÃ NGUỒN VỚI PYTEST

I. YÊU CẦU CHUNG

Với mỗi bài tập dưới đây, hãy thực hiện các yêu cầu sau:

- **Bước 1.** Tạo file chứa đoạn mã nguồn cần kiểm thử.
- **Bước 1.** Lập bảng test case (tối thiểu 5 test case/1 bài tập) theo mẫu sau (ở bước này để trống cột Kết quả kiểm thử):

Mã số	Đầu vào	Đầu ra dự kiến	Kết quả kiểm thử
1			
2			
3			
…			

- **Bước 3.** Sử dụng thư viện PyTest và viết các câu lệnh để kiểm thử đoạn mã nguồn đã cho (01 câu lệnh/1 test case) trong cùng file.
- **Bước 4.** Chạy thử các câu lệnh kiểm thử với file mã nguồn ở bước 1 và ghi lại kết quả vào cột Kết quả kiểm thử trong bảng test case đã lập ở bước 2.

II. CÁC BÀI TẬP

Bài tập 1

Cho kiểm tra số nguyên tố như sau:

```
# Define a function 'is_prime' to check if a number is prime.
def is_prime(number):
    if number < 2:
        return False
    for i in range(2, int(number**0.5) + 1):
        if number % i == 0:
            return False
    return True
```

Bài tập 2

Cho hàm kiểm tra mảng đã được sắp xếp theo thứ tự tăng dần như sau:

```
# Define a function 'is_sorted_ascending' to check if a list is sorted in ascending order.
def is_sorted_ascending(lst):
    # Check if all elements at index i are less than or equal to elements at index i+1.
    return all(lst[i] <= lst[i+1] for i in range(len(lst)-1))
```

Bài tập 3

Cho hàm kiểm tra hai danh sách bằng nhau như sau:

```
# Define a function 'lists_are_equal' to check if two lists are equal.
def lists_are_equal(nums1, nums2):
    return nums1 == nums2
```

Bài tập 4

Cho hàm kiểm tra xem một chuỗi có là chuỗi Palindrome hay không như sau:

```
# Define a function 'is_palindrome' to check if a string is a palindrome.
def is_palindrome(string):
    return string == string[::-1]
```

Bài tập 5

Cho hàm kiểm tra tập tin có tồn tại hay không như sau:

```
import os
# Define a function 'file_exists' to check if a file exists in a specified directory.
def file_exists(directory, filename):
     # Create the full file path by joining the directory and filename.
    file_path = os.path.join(directory, filename)
    # Check if the file exists at the specified path.
    return os.path.exists(file_path)
```

Bài tập 6

Cho hàm tính diện tích của hình tròn dựa vào bán kính như sau:

```python
import math
# Calculate the area of a circle based on the given radius.
def calc_area(radius):
    s = math.pi * radius ** 2
    return round(s, 2)
```

Bài tập 7

Cho hàm tính thể tích của khối lập phương dựa vào cạnh như sau:

```python
# Calculate the volume of a cubic based on the given edge.
def cubic_volume(a):
    return round(a**3, 3)
```

Bài tập 8

Số may mắn là các số nguyên dương chỉ chứa các chữ số là 4 hoặc 7.

Bình thích sự may mắn, tuy nhiên số lượng số may mắn không nhiều. Chính vì thế, Bình định nghĩa ra một loại số mới, gọi là số gần may mắn với điều kiện số các chữ số may mắn (4 và 7) trong số đó là một số may mắn.

Cho hàm kiểm tra số gần may mắn như sau:

```python
# Check if a number is a nearly-lucky number.
def is_nearly_lucky_number(n):
    temp = str(n.count('4') + n.count('7'))
    return (temp.count('4') + temp.count('7') == len(temp))
```

Bài tập 9

Sau khi quan sát nhân tình thế thái, Bình rút ra được 3 chân lý sau:

- "Mấy đứa trẻ trung đều đẹp cả".
- "Mấy đứa đẹp đều có người yêu".
- "Mấy đứa không đẹp mà cũng có người yêu chắc chắn là vì nó giàu".

Cho 4 biến logic: tre, dep, gau, giau lần lượt tương ứng với giá trị chân lý của các mệnh đề sau:

- "Bạn X trẻ trung".
- "Bạn X xinh đẹp".
- "Bạn X có người yêu".
- "Bạn X giàu có".

Cho hàm kiểm tra xem X có thỏa các chân lý mà Bình quan sát hay không như sau:

```
# Check with input (tre, dep, gau, giau), whether it is right or wrong.
def check_logic_of_life(tre, dep, gau, giau):
    out = 1
    if (tre == 1 and dep == 0) or (dep == 1 and gau == 0) or (dep == 0 and gau == 1 and giau == 0):
        out = 0
    return out
```

Bài tập 10

Cho hàm tính số hộp cần sử dụng để đựng một số lượng trứng cho trước như sau:

```
# Calculate how many egg boxes will be required for a given quantity of eggs.
def boxes_required(egg_qty, box_capacity, box_qty):
    # param qty: Number of eggs.
    # param box_capacity: How many eggs will fit in a box.
    # return: The number of boxes that are required.
    return (int(egg_qty / box_capacity) + 1 == box_qty)
```

Bài tập 11

Cho hàm kiểm tra chức năng của hàm upper() như sau:

```
# Test upper method for string
def string_upper(s, S):
    return (s.upper() == S)
```

Bài tập 12

Nhập vào hai ma trận vuông A và B, biết 2 ma trận có cùng kích thước n. Hãy kiểm tra xem liệu B có phải là A sau khi thực hiện phép quay 90, 180, 270, 360 độ.

```
import numpy as np
# Check whether matrix B is the rotated form of matrx A.
def rotate_matrix(A, B):
    A = np.array(A, int)
    B = np.array(B, int)
    flag = False
    for k in range(1, 4+1):
```

```
        if np.array_equal(B, np.rot90(A, k)):
            flag = True
            break
    return flag
```

Bài tập 13

Số dư còn lại khi lấy lũy thừa m^n chia cho giá trị 1.000.000.007.

Cho hàm tính giá trị lũy thừa nhanh bằng cách đệ quy như sau:

```
# Check the result of calculating a power with faster way.
def fast_power(m, n):
    if n == 0:
        return 1
    if n%2 == 0:
        return fast_power(m, n//2)**2 % 1000000007
    else:
        return fast_power(m, n//2)**2 * m % 1000000007
```

Bài tập 14

Cho hàm tính giá trị trung bình của các phần tử trong một danh sách như sau:

```
# Calculate mean of a list.
def find_mean(A):
    sum = 0
    for i in range(len(A)):
        sum += A[i]
    return sum / len(A)
```

Bài tập 15

Xác định xem bảng Sudoku 9 x 9 có hợp lệ hay không. Chỉ những ô đã điền mới cần được xác thực theo các quy tắc sau:

- Mỗi hàng phải chứa các chữ số từ 1-9 không lặp lại.
- Mỗi cột phải chứa các chữ số từ 1-9 không lặp lại.
- Mỗi bảng con trong số chín bảng con 3 x 3 của bảng 9 x 9 phải chứa các chữ số 1-9 không lặp lại.

Ghi chú:

- Bảng Sudoku (được lấp đầy một phần) có thể hợp lệ nhưng không nhất

thiết phải giải được.

- Chỉ những ô đã điền mới cần được xác thực theo các quy tắc đã đề cập.

Ví dụ:

- **Ví dụ 1:**
  - Đầu vào: board =

    [["5","3",".",".","7",".",".",".",".","."],

    ["6",".",".","1","9","5",".",".",".","."],

    [".","9","8",".",".",".",".",".","6","."],

    ["8",".",".",".","6",".",".",".",".","3"],

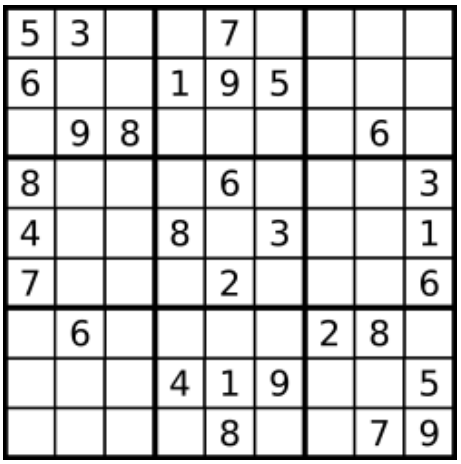
    ["4",".",".","8",".","3",".",".",".","1"],

    ["7",".",".",".",".","2",".",".",".",".","6"],

    [".","6",".",".",".",".",".","2","8","."],

    [".",".",".","4","1","9",".",".",".","5"],

    [".",".",".",".",".","8",".",".","7","9"]]



  - Đầu ra: true
- **Ví dụ 2:**
  - Đầu vào: board =

    [["8","3",".",".","7",".",".",".",".","."],

    ["6",".",".","1","9","5",".",".",".","."],

    [".","9","8",".",".",".",".",".","6","."],

    ["8",".",".",".","6",".",".",".",".","3"],

    ["4",".",".","8",".","3",".",".",".","1"],

    ["7",".",".",".",".","2",".",".",".",".","6"],

    [".","6",".",".",".",".",".","2","8","."],

    [".",".",".","4","1","9",".",".",".","5"],

    [".",".",".",".",".","8",".",".","7","9"]]

  - Đầu ra: false
  - Giải thích: Tương tự như Ví dụ 1, ngoại trừ số 5 ở góc trên cùng bên trái được sửa đổi thành 8. Vì có hai số 8 ở hộp phụ 3x3 trên cùng bên trái nên nó không hợp lệ.

Các ràng buộc:

- board.length == 9
- board[i].length == 9
- board[i][j] is a digit 1-9 or '.'

Cho hàm kiểm tra tính hợp lệ của bảng Sudoku như sau:

```
import collections
# Check whether Sudoku board is valid or not.
def is_valid_sudoku(board):
        rows,   cols,   triples   =   collections.defaultdict(set),
collections.defaultdict(set), collections.defaultdict(set)
    for i, row in enumerate(board):
        for j, c in enumerate(row):
            if c != "." and c in rows[i] or c in cols[j] or c in
triples[(i //3, j // 3)]:
                return False
            elif c != ".":
                rows[i].add(c); cols[j].add(c); triples[(i // 3,
j // 3)].add(c)
    return True
```