```
# -*- coding: utf-8 -*-

# ============================================================ #
# ============================================================ #
# Programme étudiant la connexité d'un graphe simple non orienté #
# à partir de sa matrice d'adjacence.                          #
# JANVIER 2015                                                 #
# ============================================================ #
# ============================================================ #

# N'importons de numpy que ce dont nous avons réellement besoin !
# Rappelons que la fonction "dot" de la bibliothèque numpy permet
# d'effectuer des produits matriciels.
from numpy import zeros, dot

# Fonction pour la construction de la matrice d'adjacence.
# Les arguments sont :
# (1) L'ordre du graphe
# (2) La liste des arêtes du graphe
def fMatAdj(n,LA):
    # n est le nombre de sommets du graphe
    # LA est une liste dont chaque élément est une liste comportant deux entiers
    # i et j si l'arête [S(i)S(j)] existe.
    Adj = zeros((n,n))
    # Construction de la matrice d'adjacence Adj. La validité de la liste LA est
    # vérifiée au fur et à mesure.
    for k in range(len(LA)):
        i, j = LA[k][0], LA[k][1]
        if len(LA[k]) > 2 or i == j or Adj[i,j] != 0:
            return ('Liste non valide !')
        else:
            Adj[i,j], Adj[j,i] = 1, 1
    return Adj

# Fonction d'étude de la connexité d'un graphe simple à partir de sa matrice d'adjacence
# (seul argument fourni).
def connexe(Adj):
    # La matrice fournie est-elle carrée ?
    # Remarque : on pourrait mettre en oeuvre bien d'autres tests !
    nl, nc = Adj.shape[0], Adj.shape[1]
    if nl != nc:
        return('La matrice d\'adjacence n\'est pas carrée !')
    # La matrice d'adjacence est carrée.
    # On calcule alors la somme de ses puissances (l'exposant variant de 0 à n-1).
    MC = zeros((n,n))
    for i in range(n):
        MC[i,i] = 1
    SMC = MC
    for i in range(1,n):
        MC = dot(MC,Adj)
        SMC += MC
    # Y a-t-il un zéro dans la matrice somme obtenue ?
    # On tient compte ici du fait que la matrice étudiée, SMC, est symétrique.
    test = True
    for i in range(n):
        for j in range(i,n):
            if SMC[i,j] == 0:
                test = False
                break
    return(test)

# PROGRAMME PRINCIPAL
# ===================

# Définition du graphe par ses arêtes.
# On travaille ici avec un petit graphe d'ordre 5.
n = 5
LA = [[0,2],[2,3],[3,1],[0,4],[2,4],[1,2]]
# On peut simplement donner à n la valeur 6 pour "créer" un graphe non connexe.
# n = 6

# Construction de la matrice d'adjacence et affichage.
MA = fMatAdj(n,LA)
print('La matrice d\'adjacence :')
print(MA)
```

```
# Le graphe étudié est-il connexe ?
if connexe(MA):
    print('Le graphe considéré est connexe !')
else:
    print('Le graphe considéré n\'est pas connexe !')

# ================
# FIN DU PROGRAMME
```