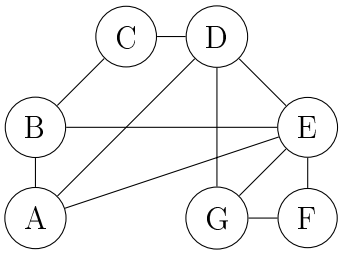
Corrigé Code de partage avec Capytale : ded2-1732010

Exercice 1

On considère le graphe G suivant :

1. Déterminer la matrice d'adjacence M de ce graphe et la rentrer avec Python.




```
M=np.array([
[0,1,0,1,1,0,0],
[1,0,1,0,1,0,0],
[0,1,0,1,0,0,0],
[1,0,1,0,1,0,1],
[1,1,0,1,0,1,1],
[0,0,0,0,1,0,1],
[0,0,0,1,1,1,0]])
```

2. A l'aide de Python, calculer M^4, quel est le nombre de chaînes de longueur 4 reliant B et C ?

   On calcule M^4 par exemple avec la commande `al.matrix_power(M,k)` (après avoir importé `numpy.linalg` au préalable). Le coefficient $2,3$ ou $3,2$ de M^4 vaut 5, donc il y a 5 chaines de longeur 4 qui lient B et C

3. Le graphe G est-il complet ?

   Non, car il n'y a pas d'arête reliant A à C par exemple.

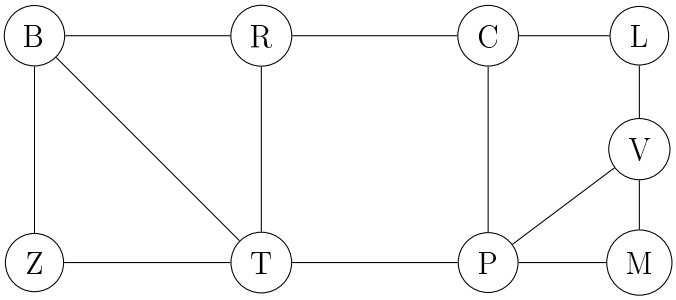
4. Le graphe G est-il connexe ?

   Oui cela se voit à l'œil nu. On peut tester le critère de connexité avec Python (cf. programme ci-dessous), à savoir calculer $I_7 + M + M^2 + \cdots + M^6$ et vérifier que ses coefficients sont tous strictement positifs, ce qui est largement le cas.

```
# matrice de connexité
M_C=np.zeros(7) # ne contient que des zéros au début
for k in range (0,7):
    M_C=M_C+al.matrix_power(M,k) # on complète en ajoutant les puissances (
   entre 0 et 6) de M
```

Exercice 2 - autoroutes du soleil

Le graphe ci-contre représente les autoroutes entre les prinipales villes du sud de la France : Bordeaux (B), Clermont-Ferrand (C), Lyon (L), Marseille (M), Montpellier (P), Brive (R), Toulouse (T), Valence (V) et Biarritz (Z).



1. Définir la matrice d'adjacence M de ce graphe et la rentrer dans Python avec la commande ci-dessous.

```
M=np.array([
[0,1,0,0,0,0,0,1,1],[1,0,1,0,0,0,0,1,0],[0,1,0,1,0,0,1,0,0],
[0,0,1,0,1,0,0,0,0],[0,0,0,1,0,1,1,0,0],[0,0,0,0,1,0,1,0,0],
[0,0,1,0,1,1,0,1,0],[1,1,0,0,0,0,1,0,1],[1,0,0,0,0,0,0,1,0]])
```

2. Quel est l'ordre de ce graphe ?
   Ecrire une fonction Python qui prend en argument une matrice d'adjacence A et renvoie l'ordre du graphe, le vérifier avec M

   Ce graphe est d'ordre 9, avec une matrice d'adjacence, on peut le trouver en calculant la taille de la matrice (par exemple avec `len`).

```
#fonction qui donne l'ordre d'une matrice
def ordre(M):
    return len(M)
```

   On teste ensuite avec `ordre(M)` (si une matrice M a été définie préalablement).

3. Tester la commande `np.eye(7)`, que renvoie-t-elle ?

   Elle renvoie la matrice I_7

4. Ce graphe est-il connexe ? est-il complet ?

   Oui il est connexe cela se voit à l'œil nu. On peut utiliser le critère de connexité comme à l'exercice 1 :

```
# matrice de connexité
M_C=np.zeros(9) # ne contient que des zéros au début
for k in range (0,9):
    M_C=M_C+al.matrix_power(M,k) # on complète en ajoutant les puissances (
   entre 0 et 8) de M
```

   Le graphe n'est pas complet, par exemple il n'existe pas d'arête entre Z et R

5. Déterminer le degré de chaque sommet. Ecrire une fonction Python qui prend en argument une matrice d'adjacence A et un sommet i et renvoie le degré du sommet i, le vérifier avec M

   A partir de la matrice d'adjacence, on trouve le degré d'un sommet en faisant la somme des coefficients de la ligne correspondante (attention Python commence à la ligne 0) :

```
#fonction qui donne le degré d'un sommet (le sommet 1 est à la ligne 0)
def degre(M,i):
    return np.sum(M[i-1,:])
```

6. Le graphe G admet-il une chaîne eulérienne, un cycle eulérien ? Ecrire une fonction Python qui prend en argument une matrice d'adjacence A d'un graphe connexe et renvoie `True` si le graphe est eulérien et `False` sinon.

   Le graphe n'admet pas de chaine eulérienne et a fortiori pas de cycle eulérien car il est connexe et plus de 2 sommets sont de degré impair (par exemple (B, R et C). Pour tester si une matrice quelconque est eulérienne, on peut vérifier que chaque sommet est de degré pair. Pour tester la parité d'un nombre on peut utiliser le critère vu dans un T.P. précédent : un nombre entier n est pair si et seulement si $2\left\lfloor \frac{n}{2} \right\rfloor = n$, on peut le tester avec la fonction suivante :

```
# fonction pour tester le caractère eulérien d'une matrice
def euler(M) :
    euler = True # valeur par défaut
    for i in range(0,ordre(M)):
        d=degre(M,i)
        if 2*np.floor(d/2)!=d: # teste la parité
            euler = False
    return euler
```

7. Un touriste atterrit à Lyon et loue une voiture. Déterminer, en justifiant par une phrase, s'il pourra visiter toutes les villes en empruntant une et une seule fois chaque autoroute.

Non car le graphe n'admet pas de chaine eulérienne.