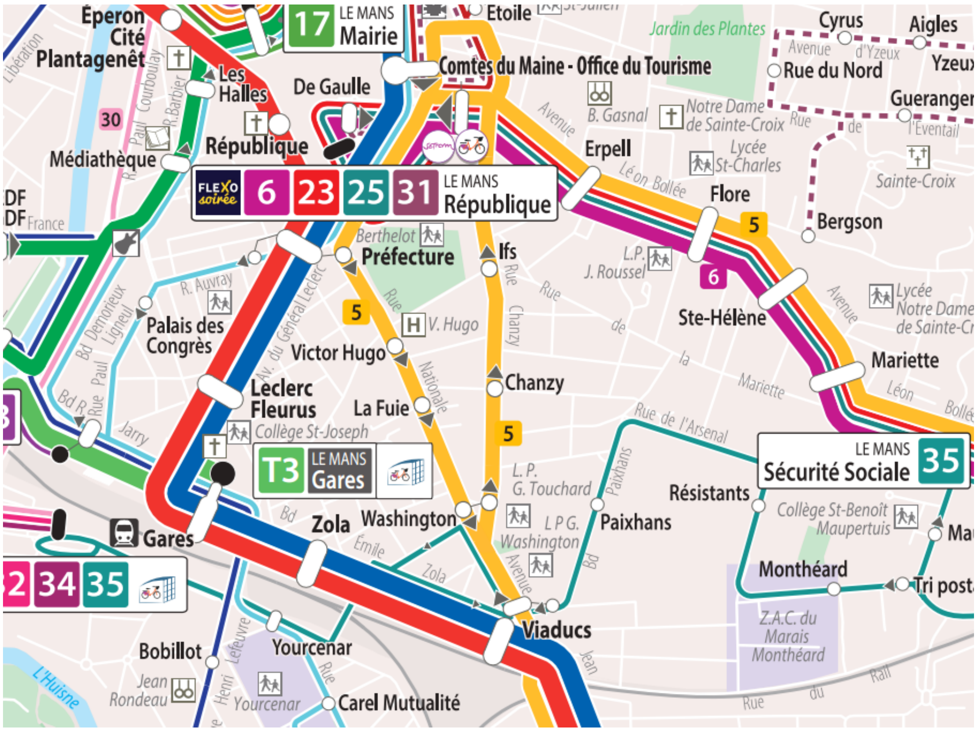
# Les Graphes

## Définitions

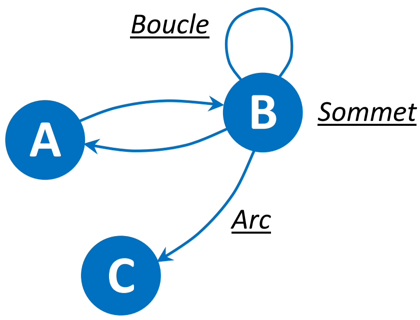
Un **graphe** est une structure permettant de **modéliser les relations entre un ensemble d'éléments**.

Les algorithmes élaborés pour résoudre des problèmes concernant les graphes ont de nombreuses applications dans tous les domaines liés à la notion de réseau.

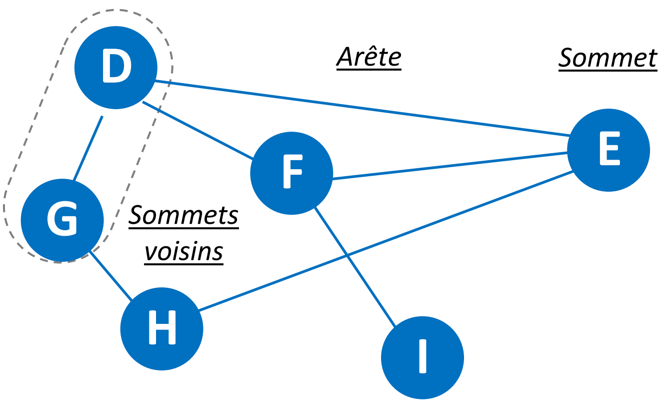
***Exemples*** : *Réseau routier ou aérien (liaisons entre villes), stations de métro (liaisons entre stations), réseau social (relations entre individus), réseau de distribution électrique (connexions entre postes sources), site Web (liaisons hyperliens entre pages), organisation logistique (relations entre événements ne pouvant avoir lieu en même temps), etc…*




## Représentation graphique

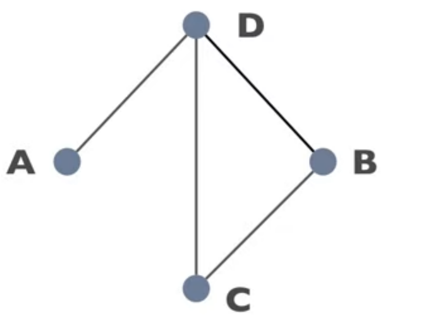
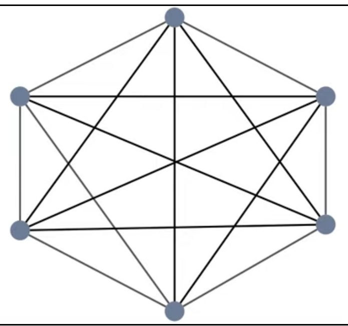
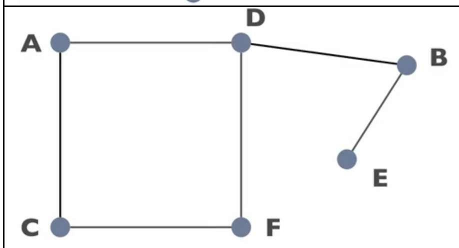
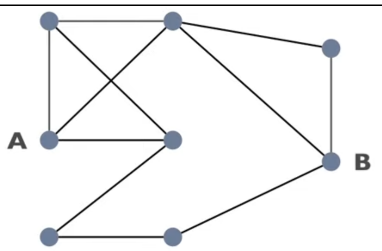
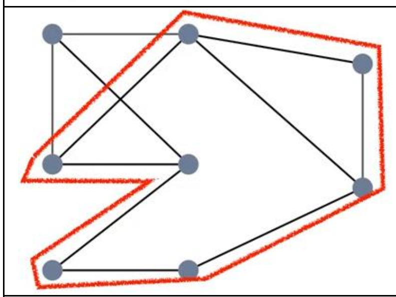
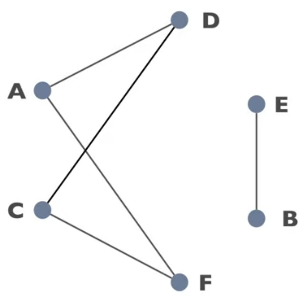



***Graphe orienté (ordre 3)***

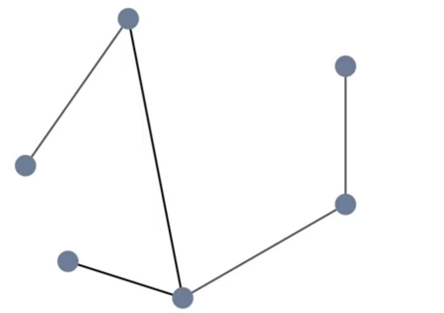
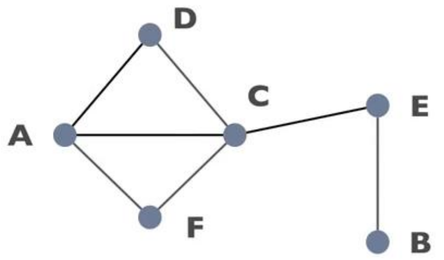



***Graphe non-orienté (ordre 6, simple)***

A partir de la vidéo de présentation des graphes, compléter le tableau suivant :

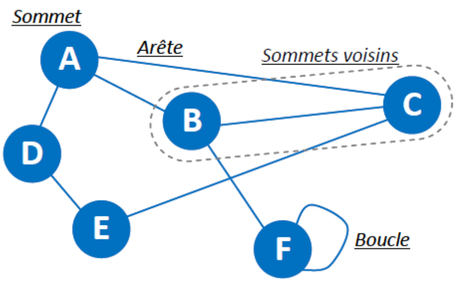
| | |
|---|---|
|  | Nombre de sommets (ordre) : **4**<br><br>Lesquelles ? **A, B, C, D**<br><br>Nombre d’arêtes : **4**<br><br>Lesquelles ? **AD, BD, BC, CD** |
|  | Le graphe est-il complet ?<br><br>**Oui, car tous les sommets sont reliés entre eux.**<br><br>Le graphe est-il connexe ?<br><br>**Oui car il n’y a aucun sommet isolé. Il y a toujours une arrête entre 2 sommets.** |
|  | Quel est le degré du sommet D ?<br><br>**Le degré est de 3, car il y a 3 voisins (A, B et F)** |
|  | Quel est le chemin le plus court entre A et B<br><br>**Longueur = 2** |
|  | Quel est la valeur de ce cycle ?<br><br>**7** |
|  | Le graphe est-il connexe ?<br><br>**Non car les sommets (E, B) n’ont pas de relation avec les sommets (A, C, D, F).** |

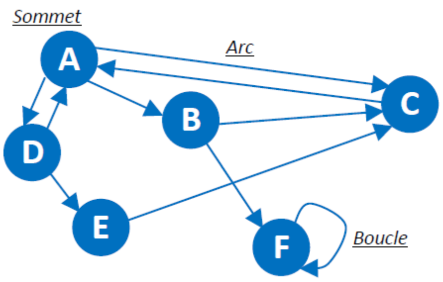
| | |
|---|---|
|  | Quel est la particularité de ce graphe ?<br><br>**C'est un arbre car celui-ci est un graphe connexe sans cycle.** |
|   | Quel est la somme des degrés du graphe ci-contre ?<br><br>**2 x 7 arêtes = 14** |

## Définitions et terminologies

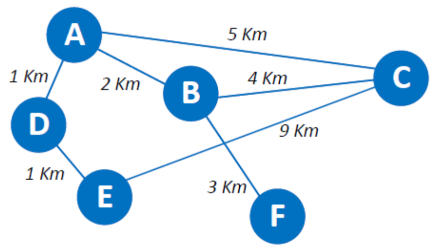
Il existe différentes possibilités pour représenter les graphes. Par exemple pour un circuit routier :



Graphe non orienté
*Liaisons entre intersections*



Graphe orienté
*avec des sens de circulation*



Graphe pondéré ou valué
*avec des distances*

On appelle graphe la donnée d'un ensemble de points appelés sommets et d'un ensemble de lignes appelées arêtes qui relient certains sommets entre eux.

Les graphes sont donc constitués de :

- **sommets** (ou **nœuds**) : ce sont les **éléments** du graphe (A, B, C, …) ;
- **arêtes** (non orientées) ou **arcs** (orientés) : ce sont les **relations entre les éléments**. Chaque liaison connecte exactement un sommet (cas d'une boucle) ou deux sommets. Il est possible d'associer à chaque arête un nombre réel positif nommé **poids** de l'arête (on parle alors de graphe **pondéré** ou **valué**).

*Remarque :* Dans un graphe non orienté, une seule arête relie deux sommets. Dans un graphe orienté, les liaisons ont un sens. Par exemple (x, y) indique qu'il y a un arc d'origine x et d'extrémité finale y.

On dit que deux sommets sont ***voisins*** (ou ***adjacents***) si au moins une arête (ou arc) les relie. On dit que l'arête est incidente aux deux sommets. *Exemple de sommets adjacents sur le graphe ci-dessus : A et B, D et E, B et F, F et F (boucle).*

Une ***boucle*** est une arête entre un sommet x et lui-même.

L'***ordre*** d'un graphe est le nombre de ses sommets.

Le ***degré d'un sommet*** x (noté d(x) ou deg(x)) désigne le nombre d'arêtes dont il est l'extrémité. *<u>Exemple</u> : le degré de C est 3 ⇒ deg(C) = 3*.
Attention, les boucles comptent double ! *<u>Exemple</u> : le degré de F est 3 ⇒* deg(F) = 3

Une ***chaîne*** est une suite quelconque d'arêtes reliant des sommets voisins (une arête ou un sommet pouvant apparaître plusieurs fois). Sa ***longueur*** est le nombre d'arêtes.
*<u>Exemple</u> : A-D-E-C-A-B, longueur 5*.

Un ***cycle*** désigne une chaîne **fermée** composée d'arêtes distinctes dont les deux extrémités sont confondues (un parcours fermé du graphe). *<u>Exemples</u> : A-B-C-A, E-C-B-A-D-E.*

La ***distance*** entre deux sommets est la longueur de la plus courte chaîne qui relie les deux sommets.

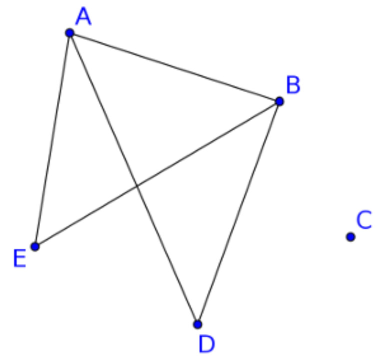
Le ***diamètre*** d'un graphe est la plus grande distance du graphe (c'est-à-dire la distance maximale du graphe pour un chemin reliant deux sommets quelconques).

Un ***sommet isolé*** est un sommet qui n'est adjacent à aucun autre sommet du graphe.

## <u>Exemple de graphes non-orienté</u>

Soit un graphe G défini par un ensemble de sommets X :{ A, B, C, D, E } et un ensemble d'arêtes A : { (A,B), (A,E), (A,D), (B,E), (B,D) } est noté G = ( X , A ).

Complétez les descriptions correspondantes au graphe G




Graphe G d'ordre : **5 car il possède 5 sommets**

Les sommets A et B sont **adjacents**

Les sommets D et E ne sont pas **adjacents**

Le sommet C est **isolé**

Le sommet A est de degré **3**, deg(A)= **3**

(A,B),(B,E),(E,A) forment **un cycle**

(A,E),(A,B),(B,D) forment **une chaîne**

| ***<u>Propriété de la somme des degrés :</u>***<br>Le nombre d'arêtes est égal à la moitié de la somme des degrés des sommets |
|---|

Vérifiez la propriété de la somme des degrés sur le graphe G

| | | |
|---|---|---|
| deg(A) = **3** | deg(B) = **3** | deg(C) = **0** |
| deg(D) = **2** | deg(E) = **2** | |

$$\frac{\mathbf{deg(A) + deg(B) + deg(C) + deg(D) + deg(E)}}{\mathbf{2}} = \frac{\mathbf{10}}{\mathbf{2}} = \mathbf{5 \text{ soit le nombre d'arêtes}}$$

**Exercice 1 :** Lors d'une soirée 4 personnes décident de trinquer le verre de l'amitié. Quel est le nombre de tintements de verre que vous allez entendre ?

<table>
<tr>
<td>

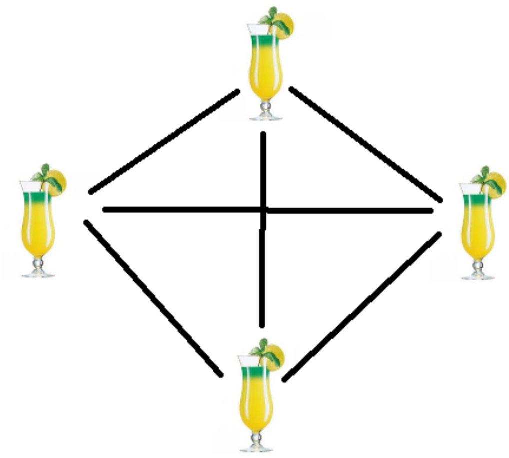
***Solution*** :

*On peut représenter le problème par un graphe dont les personnes sont les sommets (4) et les tintements de verre les arêtes. Le degré de chaque sommet vaut **3 (car une personne tinte le verre des 3 autres).***

***La somme des degrés vaut donc : 4x3 = 12.***

***Le nombre d'arêtes est donc la moitié, soit 6 tintements de verre.***

</td>
<td>



</td>
</tr>
</table>

## Structures particulières de graphes

Un graphe est :

- ***simple***, s'il est non orienté et sans boucle.
- ***complet***, si deux sommets quelconques distincts sont toujours voisins.
- ***connexe***, s'il existe toujours une chaîne reliant deux sommets (il n'y a donc pas de sommet isolé sans arête).

**Pour un graphe orienté G = (X, A) :** X est l'ensemble des sommets du graphe, A est l'ensemble des arcs du graphe.

- Pour un sommet quelconque, ses ***successeurs*** (respectivement ***prédécesseurs***) sont les sommets voisins qui sont reliés par un arc **partant du sommet** (respectivement **arrivant au sommet**).

  On désigne par **Γ+(x)**, l'ensemble des sommets successeurs du sommet x ;

  On désigne par **Γ-(x)**, l'ensemble des sommets prédécesseurs du sommet x ;

- ***Degré sortant*** : On notera d+(x) ou deg+(x) le nombre d'arcs **sortants** du sommet x.
- ***Degré entrant*** : On notera d-(x) ou deg-(x) le nombre d'arcs entrants du sommet x.

  Le degré d'un sommet sera (d+) + (d-)

- Dans un graphe orienté, un ***circuit*** est une suite d'arcs consécutifs (chemin) dont les deux sommets extrémités sont identiques.
- Le graphe (ou une partie du graphe nommée ***composante***) est ***fortement connexe***, s'**il existe toujours un chemin entre deux sommets quelconques** (et dans les **deux sens**).

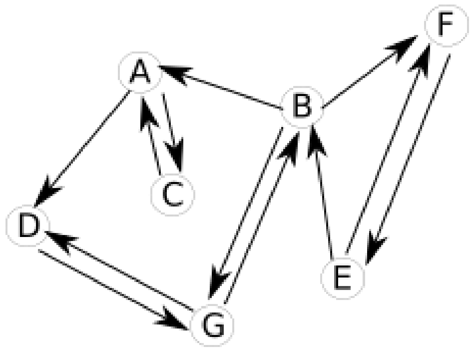
Si le voisinage est immédiat, on dit que la relation est de **niveau 1**. De proche en proche, les voisins indirects plus lointains (reliés à un degré supérieur) seront de **niveau 2**, **3**, .etc…

**Exemple de graphes orienté**

Soit un graphe G défini par un ensemble de sommets X :{ A, B, C, D, E, F,G } et un ensemble d'arcs A : { (A,D), (A,C), (C,A), (B,A), (D,G), (G,D), (G,B), (B,G), (B,F), (E,B), (E,F), (F,E) } est noté G = ( X , A ).

Complétez les descriptions correspondantes au graphe G




Graphe d'ordre **7**

Les successeurs du sommet B sont **A, F et G**

$\Gamma$+(B) = { **A, F, G**}

Les prédécesseurs du sommet B sont **G et E**

$\Gamma$-(B) = { **E, G**}

deg+(B) = **3** deg-(B) = **2**

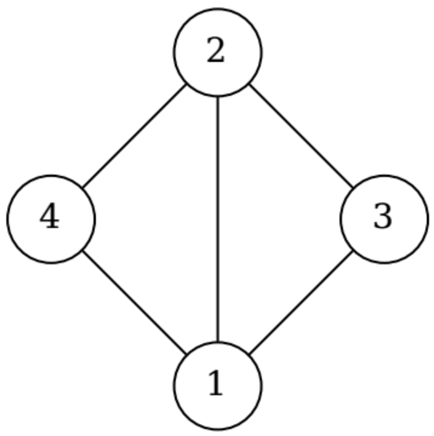
Le sommet B est de degré **5**

(A,D),(D,G),(G,B),(B,F) forment **un chemin**

(B,F),(F,E),(E,B) forment **un circuit**

## Exercice 2 :

On considère le graphe G = (X, A) suivant :

| | |
|---|---|
| <br> | Est-ce un graphe simple ? Orienté ?<br>**Graphe simple et non-orienté**<br><br>Quels sont les voisins de 1 ?<br>**2, 4, 3**<br><br>Combien peut-on ajouter d'arêtes à ce graphe ?<br>**Une arrête entre 3 et 4**<br><br>Quel est l'ordre du graphe G ?<br>**Le graphe G est d'ordre 4** |

Quel est le degré des sommets du graphe G ?

deg(1) = **3** **deg(2) = 3** **deg(3) = 2** **deg(4) = 2**

Quel est le nombre d'arêtes en appliquant la propriété de la somme des degrés ?

$$\frac{\deg(1) + \deg(2) + \deg(3) + \deg(4)}{2} = \frac{10}{2} = 5$$ **soit le nombre d'arêtes**

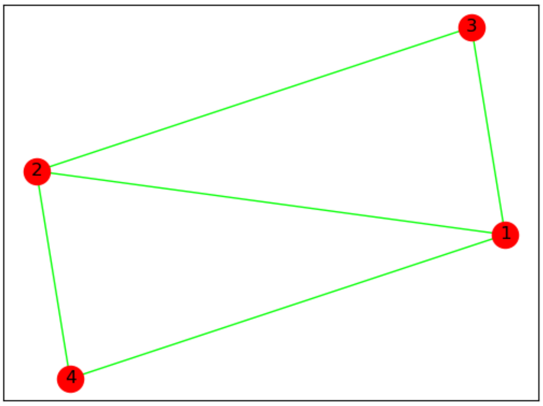
  

Nous allons utiliser le langage Python pour visualiser le graphe G précédent. Pour cela, le tracé du graphe s'effectuera avec la bibliothèque networkx.

Code python

```
import networkx as nx
import matplotlib.pyplot as plt

g=nx.Graph() #non orienté
g.add_node(1)
g.add_node(2)
g.add_node(3)
g.add_node(4)
g.add_edge(1,2)
g.add_edge(1,4)
g.add_edge(1,3)
g.add_edge(2,4)
g.add_edge(2,3)

matrice=nx.adjacency_matrix(g)
print(matrice.todense())

print(nx.info(g))
nx.draw_networkx(g,node_color='#FF0000',edge_color='#00FF00')

plt.show()
```

Résultat affiché




Matrice d'adjacence

| Sommet | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | **0** | **1** | **1** | **1** |
| 2 | **1** | **0** | **1** | **1** |
| 3 | **1** | **1** | **0** | **0** |
| 4 | **1** | **1** | **0** | **0** |

## Exercice 3 :

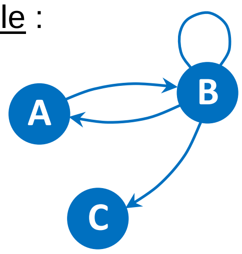
On souhaite organiser un tournoi de football avec 4 équipes (numérotées de 1 à 4). Chaque équipe rencontre une seule fois toutes les autres.

1. Représenter la situation sous la forme d'un graphe.
   **Graphe identique au verre de l'amitié**
2. Combien d'arêtes possède-t-il ? En déduire le nombre de matchs au total pour ce tournoi.
   **6 arêtes, donc 6 matchs.**
3. Ce graphe est-il connexe ? **Oui car il existe toujours une chaîne reliant 2 sommets (pas de sommet isolé).**
4. Ce graphe est-il complet ? **Oui car tous les sommets sont voisins.**

# Matrice d'adjacence

La **matrice d'adjacence** est une représentation mathématique d'un graphe. C'est une **matrice carrée** de côté le **nombre de sommets** du graphe (**ordre**), qui contient le nombre d'arêtes reliant les couples de sommets.

Exemple :




$$ADJ = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

***Remarque*** *:*

*Le "1" (rouge) de la diagonale de la matrice illustre la présence de la boucle du graphe.*

**Définition : On appelle matrice d'adjacence la matrice A dont le terme $a_{ij}$ vaut 1 si les sommets sont reliés par une arrête (ou arcs) et 0 sinon. I et j variant de 1 à n.**

Particularités structurelles :

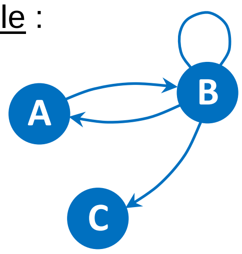
La matrice d'adjacence d'un graphe :

- non orienté est symétrique par rapport à sa diagonale ; **(coefficients $a_{ij}$ et $a_{ji}$ égaux pour tout i et j compris entre 1 et n)**
- sans boucle n'a que des 0 sur sa diagonale ;
- sans arêtes multiples n'a que des 1 ou des 0 (cas le plus fréquent) ;
- complet n'a que des 1, sauf la diagonale qui n'a que des 0 ;
- n'est pas unique, car elle dépend de l'ordre choisi des sommets.

# Liste d'adjacence

Une autre façon classique de représenter un graphe est d'associer à chaque sommet la **liste de ses voisins** (graphe non orienté) ou de **ses successeurs** (graphe orienté).

Exemple :



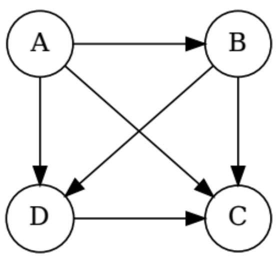

| | |
|---|---|
| A | B |
| B | A, B, C |
| C | |

*Liste des successeurs du graphe orienté ABC*

## Exercice 4 :

Un club de tennis doit sélectionner deux joueurs parmi quatre pour représenter le club à un tournoi national. Les quatre joueurs sont notés A, B, C et D. Pour réaliser la sélection, le club organise des matchs : chaque joueur rencontre les trois autres.

On donne le résultat sous la forme du graphe orienté suivant :




Règle :
- Tout match gagné donne un point ;
- Tout match perdu enlève un point ;

Les joueurs sélectionnés sont les joueurs ayant obtenu le plus grand nombre de points.

Le sens de l'arc A -> B indique que le joueur A a battu le joueur B.

Compléter le tableau des matchs ci-dessous :

| A | | | B | | | C | | | D | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Gagnés | Perdus | Total | Gagnés | Perdus | Total | Gagnés | Perdus | Total | Gagnés | Perdus | Total |
| **3** | **0** | **3** | **2** | **1** | **1** | **0** | **3** | **-3** | **1** | **2** | **-1** |

Quels sont les joueurs qualifiés ? **Les joueurs A et B sont qualifiés**

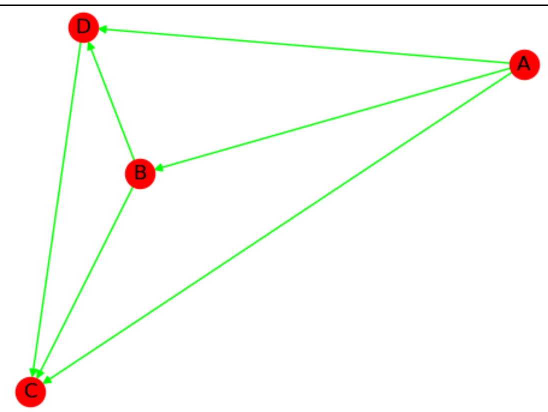
Complétez le code python afin de tracer le graphe avec la bibliothèque [networkx](networkx).

Code python

```
import networkx as nx
import matplotlib.pyplot as plt
g=nx.DiGraph() #orienté

g.add_node('A')
g.add_node('B')
g.add_node('C')
g.add_node('D')

g.add_edge('A','B')
g.add_edge('A','C')
g.add_edge('A','D')
g.add_edge('B','D')
g.add_edge('B','C')
g.add_edge('D','C')

matrice=nx.adjacency_matrix(g)
print(matrice.todense())

print(nx.info(g))
nx.draw_networkx(g,node_color='#FF0000',edge_color='#00FF00')

plt.show()
```

Complétez la matrice d'adjacence.

Résultat affiché




Matrice d'adjacence

| Sommet | A | B | C | D |
|---|---|---|---|---|
| A | **0** | **1** | **1** | **1** |
| B | **0** | **0** | **1** | **1** |
| C | **0** | **0** | **0** | **0** |
| D | **0** | **0** | **1** | **0** |

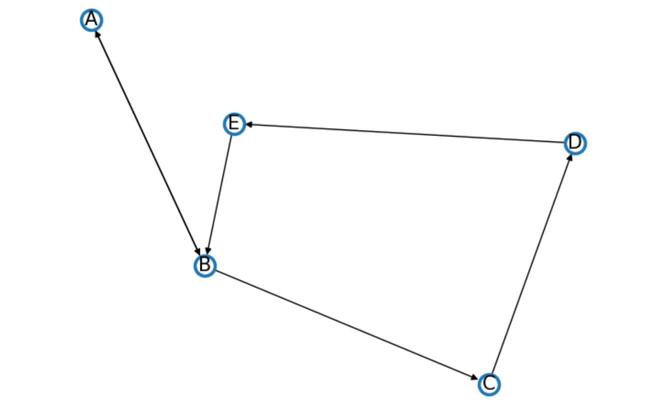
## Exercice 5 :

Représentez le graphe à partir de la matrice d'adjacence ci-dessous. Vérifiez le graphe obtenu avec le code python fourni.

Matrice d'adjacence

| Sommet | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | 1 | 0 | 0 | 0 |
| B | 1 | 0 | 1 | 0 | 0 |
| C | 0 | 0 | 0 | 1 | 0 |
| D | 0 | 0 | 0 | 0 | 1 |
| E | 0 | 1 | 0 | 0 | 0 |

Résultat affiché




Code python

```python
import networkx as nx
import numpy as np
import matplotlib.pyplot as plt

A = [[0, 1, 0, 0 , 0],
     [1, 0, 1, 0, 0],
     [0, 0, 0, 1, 0],
     [0, 0, 0, 0, 1],
     [0, 1, 0, 0, 0]]

g = nx.from_numpy_matrix(np.array(A),create_using=nx.DiGraph)

raw_labels = ["A", "B", "C", "D", "E"]
lab_node = dict(zip(g.nodes, raw_labels))

for edge in g.edges(data=True):
    print(edge)
print(nx.info(g))
pos = nx.spring_layout(g)
nx.draw(g, pos,edge_color='#000000')
nx.draw_networkx_nodes(g, pos, node_size=150,node_color='#FFFFFF')
nx.draw_networkx_labels(g, pos, labels=lab_node, font_size=15, font_family='sans-serif')

plt.show()
```

Pourquoi peut-on affirmer au vue de la matrice que le graphe est orienté ?

**La matrice n'est pas symétrique, le graphe est forcément orienté.**

**(coefficients $a_{ij} \neq a_{ji}$ pour tout i et j compris entre 1 et n)**

# Exploration des graphes (parcours)

Beaucoup de problèmes sur les graphes nécessitent que l'on explore l'ensemble des sommets et des arcs/arêtes du graphe. Deux des principales stratégies d'exploration sont le **parcours en largeur** (par niveau) et le **parcours en profondeur** (chemins les plus longs).

Le problème consiste à déterminer un **ordre** sur les visites des sommets voisins d'un sommet du graphe. Le sommet de départ est nommé ***racine***. Dans un parcours, chaque **sommet** apparaît **une fois et une seule**. Lors de l'exploration du graphe, les sommets visités sont marqués.

Si, après l'exploration, certains sommets du graphe restent non visités, il suffit d'en sélectionner un comme nouvelle racine et de recommencer.

## Parcours en largeur ou BFS (Breadth First Search)

À partir d'une racine choisie, on explore le graphe en considérant tous les voisins de chaque sommet, en les mémorisant dans une **file** dans l'ordre du parcours. Chaque sommet visité est marqué lors de sa première découverte. Puis on progresse d'un niveau de voisinage et ainsi de suite.

https://youtu.be/NrQGxfFMYzs

## Parcours en profondeur ou DFS (Depth First Search)

À partir d'une racine choisie, on suit un chemin dans le graphe, en mémorisant les sommets visités dans une **pile**, jusqu'à ce qu'on ne puisse plus progresser. Dans ce cas, on fait marche arrière pour trouver un autre chemin à partir d'un sommet qui n'a pas déjà été visité.

https://youtu.be/kcedjJOjDpg