

Ce cours est extrait de :

*ING1 – Génie Mathématique*  
*Théorie des graphes – Notes de cours*  
*Maria Malek, Romain Dujol*  
*2019 – 2020*

# Graphe non orienté

- **Définition (Graphe non orienté)**

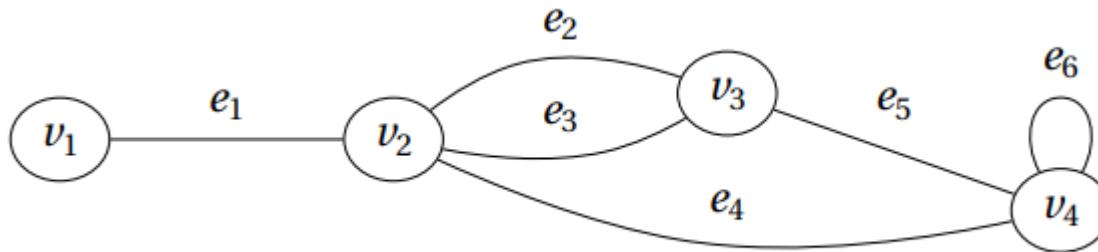
Un graphe non orienté est un triplet  $G = (V, E, \gamma)$  où :

- $V$  est un ensemble fini non vide: tout élément de  $V$  est appelé un sommet de  $G$  ;
- $E$  est un ensemble fini (éventuellement vide): tout élément de  $E$  est appelé une arête;
- $\gamma$  est une application de  $E$  dans  $V \times V$ .

- **Extrémités des arêtes**

Pour tout arête  $e \in E$ , avec  $\gamma(e) = (\nu, \nu')$ , les sommets  $\nu$  et  $\nu'$  sont appelés les extrémités de  $e$ .

# Graphe non orienté



est une représentation de  $G = (\{v_1, v_2, v_3, v_4\}, \{e_1, e_2, e_3, e_4, e_5, e_6\}, \gamma)$  avec :  $\gamma : E \rightarrow V^2$

$e_1$	$\mapsto$	$(v_1, v_2)$
$e_2$	$\mapsto$	$(v_2, v_3)$
$e_3$	$\mapsto$	$(v_2, v_3)$
$e_4$	$\mapsto$	$(v_2, v_4)$
$e_5$	$\mapsto$	$(v_3, v_4)$
$e_6$	$\mapsto$	$(v_4, v_4)$

# Graphe non orienté

Soit  $G = (V, E, \gamma)$  un graphe non orienté.

- **Sommets adjacents**

On dit que  $v \in V$  et  $v' \in V$  sont **voisins** ou **adjacents** si et seulement si il existe une arête  $e \in E$  dont les extrémités sont  $v$  et  $v'$ .

- **Sommet isolé**

Si  $v \in V$  n'est adjacent à aucun autre sommet, il est dit **isolé**.

- **Boucle**

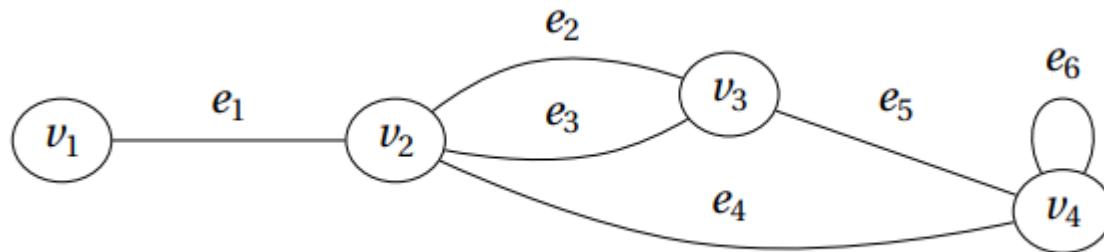
Une **boucle** est une arête dont les deux extrémités sont égales.

- **Arêtes parallèles**

Deux arêtes de  $G$  sont dites **parallèles** si et seulement si elles partagent les mêmes extrémités.

# Graphe non orienté

- Trouver les boucles, les arêtes parallèles et les sommets adjacents.



# Graphe non orienté

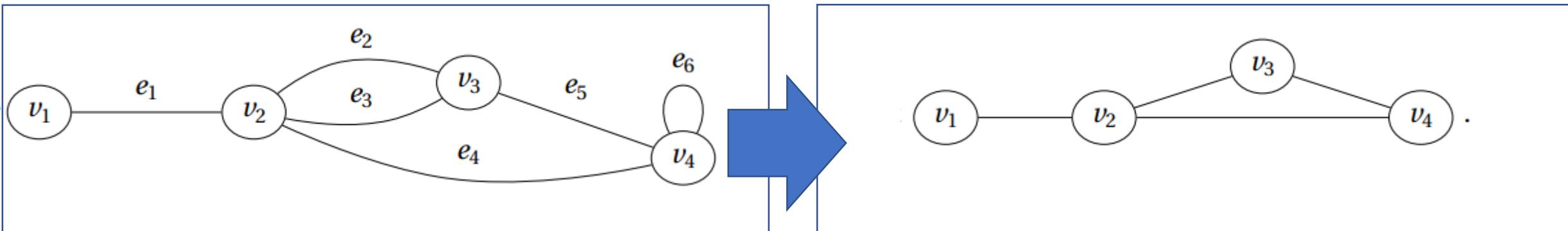
- **Graphe simple**

Un graphe non orienté  $G = (V, E, \gamma)$  est dit **simple** si et seulement si il n'a ni boucle, ni arêtes parallèles.

- **Graphe sous-jacent**

Soit  $G = (V, E, \gamma)$  un graphe non orienté. Le graphe **sous-jacent de  $G$**  est le graphe non orienté simple  $G' = (V, E')$  tel que :

- $E' \subseteq V \times V$
- $\forall (v, v') \in E' \Leftrightarrow v \neq v'$

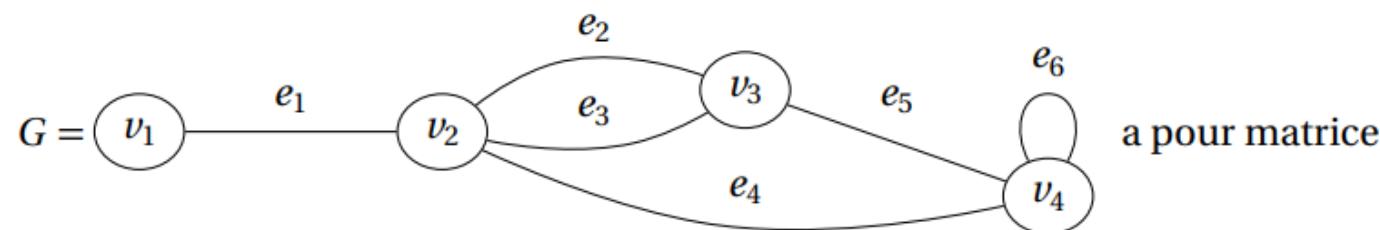


# Représentation d'un graphe non orienté

Soit  $G = (V, E, \gamma)$  un graphe non orienté. On note  $V = \{v_i\}_{1 \leq i \leq n}$  et  $n = \text{card}(V)$ .

- **Matrice d'adjacence**

La **matrice d'adjacence de  $G$**  est une matrice  $M$  carrée d'ordre  $n$  telle que  $M_{i,j}$  est le nombre d'arêtes d'extrémités  $v_i$  et  $v_j$ , avec  $1 \leq i \leq n$  et  $1 \leq j \leq n$ .



a pour matrice

$$M = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 2 & 1 \\ 0 & 2 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}$$

# Graphe non orienté

- Caractériser la matrice **d'adjacence** d'un graphe simple.
- Écrire une fonction python *show(graph)* permettant d'afficher un graphe.
- Écrire une fonction python *isSimple(graph)* permettant de vérifier si un graphe est simple.
- Écrire une fonction *graphSimple(graph)* python permettant de retourner le graphe sous-jacent au graphe passé en paramètre.

# Isomorphisme de graphes

(Isomorphisme de graphes).

Soit  $G_1 = (V_1, E_1, \gamma_1)$  et  $G_2 = (V_2, E_2, \gamma_2)$  deux graphes non orientés.

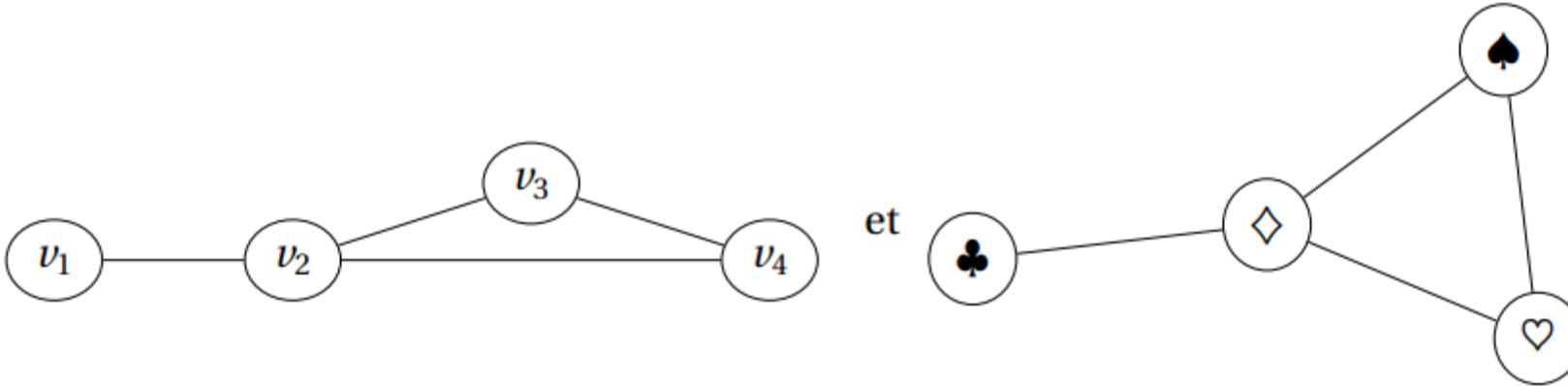
Un **isomorphisme de  $G_1$  dans  $G_2$**  est un couple  $(\phi, \psi)$  tel que :

- $\phi$  est une bijection de  $V_1$  dans  $V_2$ ;
- $\psi$  est une bijection de  $E_1$  dans  $E_2$ ;
- $\forall e \in E, \forall (v, v') \in V^2, v$  et  $v'$  extrémités de  $e \iff \phi(v)$  et  $\phi(v')$  extrémités de  $\phi(e)$

(Graphes isomorphes). Deux graphes non orientés  $G_1$  et  $G_2$  sont dits **isomorphes** si et seulement si il existe un isomorphisme de  $G_1$  sur  $G_2$ .

# Isomorphisme de graphes

Exemple.



sont isomorphes via  $\phi_1 : \{v_1, v_2, v_3, v_4\} \rightarrow \{\clubsuit, \diamondsuit, \heartsuit, \spadesuit\}$  ou  $\phi_2 : \{v_1, v_2, v_3, v_4\} \rightarrow \{\clubsuit, \diamondsuit, \heartsuit, \spadesuit\}$ .

$$\begin{array}{rcl} v_1 & \mapsto & \clubsuit \\ v_2 & \mapsto & \diamondsuit \\ v_3 & \mapsto & \heartsuit \\ v_4 & \mapsto & \spadesuit \end{array}$$

$$\begin{array}{rcl} v_1 & \mapsto & \clubsuit \\ v_2 & \mapsto & \diamondsuit \\ v_3 & \mapsto & \spadesuit \\ v_4 & \mapsto & \heartsuit \end{array}$$

# Chaines et cycles

(Chaîne). Soit  $G = (V, E, \gamma)$  un graphe non orienté.

Une **chaîne de  $G$**  est une suite finie  $C = (v_0, e_1, v_1, \dots, e_i, v_i, \dots, e_p, v_p)$  telle que :

- $\forall i \in \llbracket 0, p \rrbracket, v_i \in V$
- $\forall i \in \llbracket 1, p \rrbracket, e_i \in E$  et ses extrémités sont  $v_{i-1}$  et  $v_i$ .

L'entier  $p$  est appelé la **longueur de  $C$**  et noté  $\ell(C)$ .  $v_0$  et  $v_p$  sont appelés les **extrémités de  $C$** .

(Chaîne simple). Soit  $G = (V, E, \gamma)$  un graphe non orienté.

Une chaîne  $C = (v_0, e_1, v_1, \dots, e_p, v_p)$  de  $G$  est dite **simple** si et seulement si les arêtes constituant  $C$  sont distinctes deux à deux :

$$\forall (i, j) \in \llbracket 1, p \rrbracket^2, (i \neq j) \implies e_i \neq e_j$$

(Chaîne élémentaire). Soit  $G = (V, E, \gamma)$  un graphe non orienté.

Une chaîne  $C = (v_0, e_1, v_1, \dots, e_p, v_p)$  de  $G$  est dite **élémentaire** si et seulement si les sommets constituant  $C$  sont distincts deux à deux :

$$\forall (i, j) \in \llbracket 0, p \rrbracket^2, (i \neq j) \implies v_i \neq v_j$$

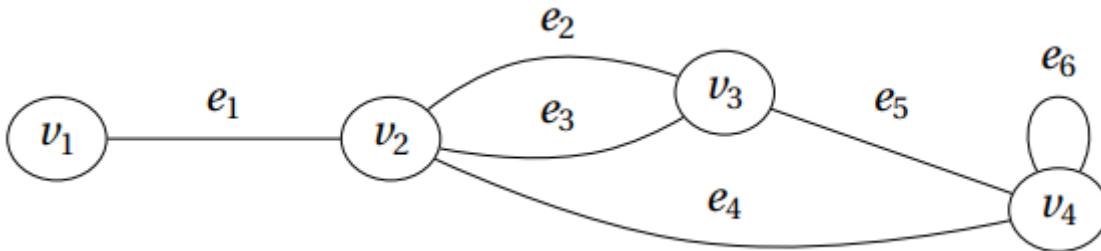
# Graphe non orienté

- Soit  $G$  un graphe et  $C$  est une chaîne de  $G$ . Laquelle des deux implications suivantes est toujours vraie (justifier votre réponse):
  - **$C$  est élémentaire  $\Rightarrow C$  est simple** Si on ne répète pas les noeuds, on ne répète pas les arrêtes donc simple
  - $C$  est simple  $\Rightarrow C$  est élémentaire
- Écrire une fonction python `isChain(graph, listNodes)` permettant de vérifier si une liste de noeuds d'un graphe forment une chaîne du graphe.
- Écrire une fonction python `isChainSimple(graph, listNodes)` permettant de vérifier si une liste de noeuds d'un graphe forment une chaîne **simple** du graphe.
- Écrire une fonction python `isChainElementary(graph, listNodes)` permettant de vérifier si une liste de noeuds d'un graphe forment une chaîne **élémentaire** du graphe.
- Écrire une fonction python `chainElementary(graph, listNodes)` permettant de retourner une chaîne élémentaire **sous-jacente** à la chaîne passée en paramètre et une liste vide si les noeuds ne forment pas une chaîne.

# Cycle dans un graphe

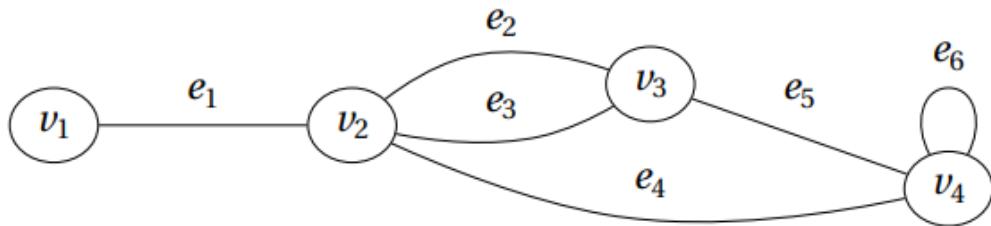
(Cycle). Soit  $G = (V, E, \gamma)$  un graphe non orienté.

Un **cycle de  $G$**  est une chaîne simple  $C = (v_0, e_1, v_1, \dots, e_p, v_p)$  telle que  $\ell(C) > 0$  et  $v_p = v_0$ .



# Degré d'un sommet

(Degré d'un sommet). Soit  $G = (V, E, \gamma)$  un graphe non orienté et  $v \in V$ .  
On appelle **degré de  $v$** , noté  $d(v)$ , le réel  $d(v) = \text{card}\{e \in E, v \text{ est une extrémité de } e\}$ .



$v$	$v_1$	$v_2$	$v_3$	$v_4$
$d(v)$	1	4	3	4

# Degré d'un sommet

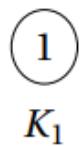
**Exercice 1.1.** *Une entreprise compte cinq employés. Est-il possible que chacun d'entre eux serre la main à trois autres personnes exactement?*

**Exercice 1.2.** *Soit  $G$  un graphe non orienté.*

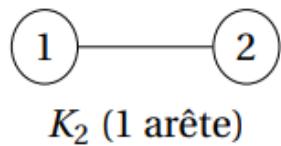
1. *Soit  $v$  un sommet de  $G$  de degré impair. Montrer qu'il existe un sommet  $v'$  de degré impair, distinct de  $v$  tel qu'il existe une chaîne entre  $v$  et  $v'$ .*
2. *En déduire que, si  $G$  a exactement deux sommets de degré impair, alors il existe une chaîne entre ces deux sommets.*

# Graphe complet

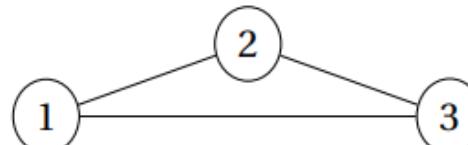
(Graphe complet). Un graphe non orienté est dit **complet** si et seulement si il est simple et que tous les sommets sont adjacents deux à deux.



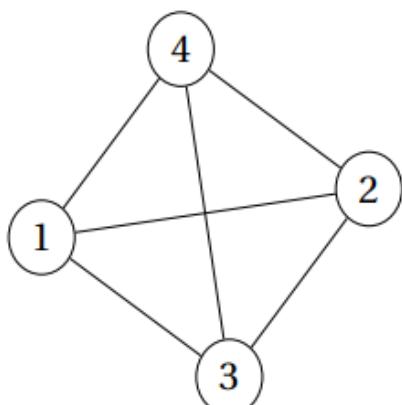
$K_1$



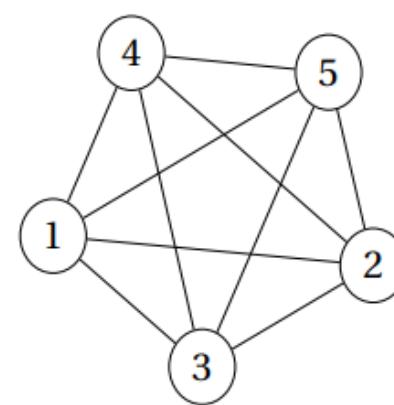
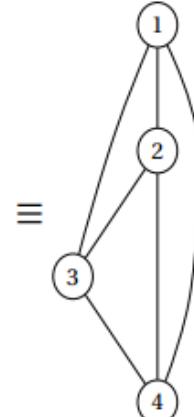
$K_2$  (1 arête)



$K_3$  (3 arêtes)



$K_4$  (6 arêtes)



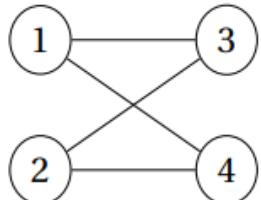
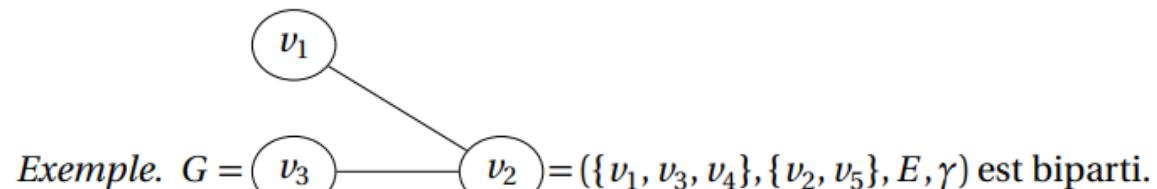
$K_5$  (10 arêtes)

# Graphe biparti

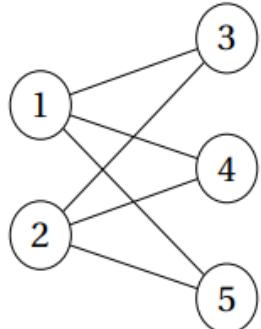
**Théorème** □ (Graphe biparti complet  $K_{n_1, n_2}$ ). Soit  $n_1$  et  $n_2$  deux entiers naturels non nuls.

Alors tout graphe biparti complet  $G = (V_1, V_2, E, \gamma)$  tel que  $\text{card } V_1 = n_1$  et  $\text{card } V_2 = n_2$  est isomorphe au graphe simple  $K_{n_1, n_2} = ([1, n_1], [n_1 + 1, n_1 + n_2], E_{n_1, n_2}, \gamma_{n_1, n_2})$  contenant toutes les arêtes possibles pour un graphe biparti :

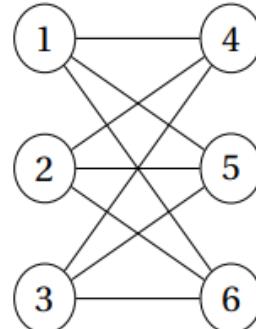
$$\forall (v_1, v_2) \in V_1 \times V_2, v_1 v_2 \in E_{n_1, n_2}$$



$K_{2,2}$



$K_{2,3}$



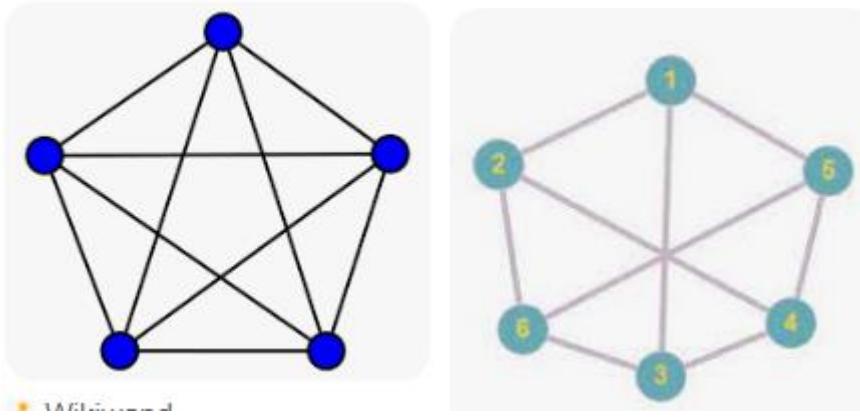
$K_{3,3}$

# Graphe régulier

**Définition 1.24** (Graphe régulier). Soit  $k$  un entier naturel.

Un graphe  $G = (V, E, \gamma)$  est dit  **$k$ -régulier** si et seulement si tout sommet de  $G$  est de degré  $k$  :  
 $\forall v \in V, d(v) = k$ .

$G$  est dit régulier si et seulement si il existe un entier naturel  $k$  tel que  $G$  est  $k$ -régulier.



# Graphes particuliers

- Écrire une fonction python  $isFull(graph)$  qui vérifie si un graphe est complet.
- Écrire une fonction python  $isBiparti(graph)$  qui vérifie si un graphe est biparti. Elle retourne trois valeurs :un booléen, la liste des noeuds de la première partition et une autre contenant la liste des noeuds de la deuxième partition. Les deux listes sont vides si le graphe n'est pas biparti.
- Définir formellement un graphe valué et reprendre la méthode  $show(graph)$ .

# Graphes connexes

(Graphe connexe). Soit  $G = (V, E, \gamma)$  un graphe non orienté.

$G$  est dit **connexe** si et seulement si pour tout couple  $(v, v') \in V^2$ , il existe une chaîne d'extrêmités  $v$  et  $v'$ .

(Composantes connexes). Soit  $G = (V, E, \gamma)$  un graphe non orienté.

Une **composante connexe de  $G$**  est un sous-graphe connexe maximal au sens de l'inclusion des sommets et des arêtes.

# Graphes connexes

- Écrire une fonction python  $isconnexe(graph)$  qui vérifie si un graphe est connexe.
- Écrire une fonction python  $compConnexe(graph)$  qui retourne l'ensemble des composantes connexes du graphe passé en paramètre.