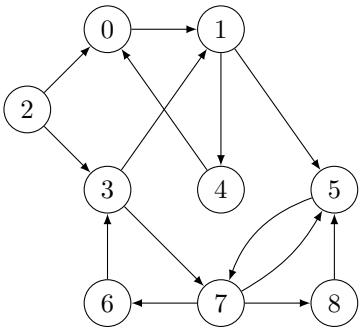
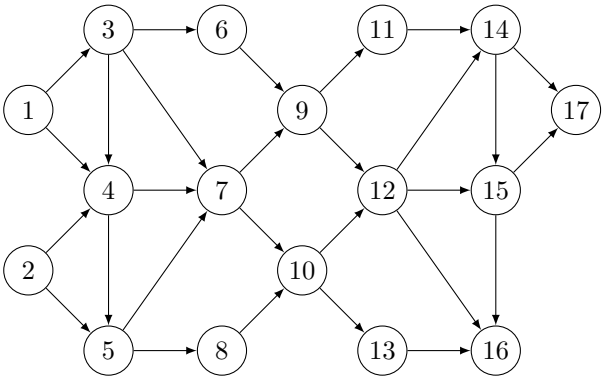
Fiche TD15 : Graphes - vocabulaire et parcours

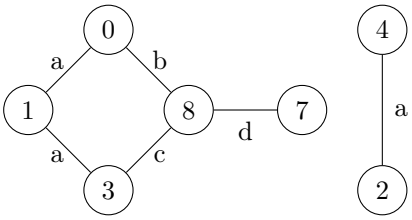
Exercice 1 *Quelques gammes*

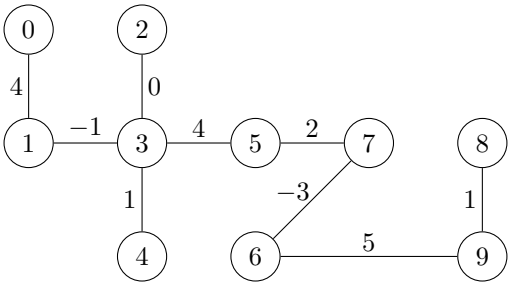
Pour chacun des graphes ci-dessous, qu'on appelera respectivement G_1, G_2, G_3 et G_4 dans l'ordre de lecture, indiquer :

1. S'ils sont acycliques, orientés, pondérés.
2. Quel est leur ordre et quel est le degré (sortant / entrant si cela est pertinent) de chacun de ses sommets.
3. Quelles sont leurs composantes connexes / fortement connexes (lorsque cette notion a un sens).









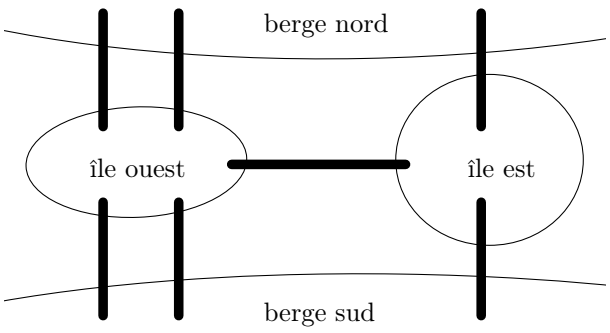
4. Quelle est la longueur d'un plus court chemin du sommet 0 vers le sommet 7 dans G_1 ? Même question entre les sommets 7 et 0 dans G_1 puis les sommets 10 et 3 de G_2.
5. Justifier que pour tout couple de sommets (u, v) dans G_4, il existe un unique chemin simple entre u et v. Quel est le poids du seul chemin entre les sommets 3 et 8 ?

Exercice 2 *Les ponts de Königsberg*

Dans tout l'énoncé, on ne considère que des graphes connexes non orientés. Un cycle eulérien dans un tel graphe G est un cycle de G empruntant une et une seule fois chaque arête de G. Si l'existence d'un tel cycle est acquise dans G, on dit que G est un graphe eulérien.

1. Montrer l'implication suivante : si un graphe est eulérien alors le degré de chacun de ses sommets est pair.
2. Explique comment étendre le résultat précédent à un multigraphe (deux sommets dans une telle structure peuvent être reliées par plusieurs arêtes différentes).

La ville de Königsberg (aujourd'hui Kalingrad) est traversée par une rivière au milieu de laquelle se trouvent deux îles. Pour traverser la rivière, on pouvait, en 1735, utiliser le système de ponts suivant :



3. Est-il possible de se promener dans les rues de Kalingrad à partir d'un point de départ quelconque dans la ville, de passer une et une seule fois sur chacun des ponts et de revenir à son point de départ ?

La première trace du résultat de la question 1 remonte à une présentation de travaux par Leonhard Euler en 1735. Ce dernier pressentait que la réciproque était vraie, mais il faut attendre les travaux de Carl Hierholzer plus de cent ans plus tard pour en avoir la preuve.

4. Prouver la réciproque de la question 1. *Indication : On peut par exemple procéder par récurrence.*

Remarque culturelle : Les aléas du temps font qu'en 2022, il n'y a plus que 5 *ponts à cet emplacement à Kalingrad. Leur configuration est telle qu'il n'est toujours pas possible de construire un cycle eulérien permettant de les traverser tous ; en revanche, il est possible de trouver un chemin eulérien (cad, on peut les traverser une et une seule fois si on accepte de ne pas commencer et terminer la promenade au même endroit) ce qui n'était pas possible du temps d'Euler.*

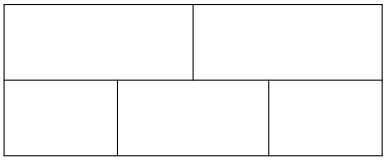
Exercice 3 *Modélisations par un graphe*

Dans cet exercice, on présente quatre problèmes indépendants les uns des autres. L'objectif est de les modéliser à l'aide d'un graphe adapté et d'étudier ce graphe de sorte à pouvoir les résoudre.

1. Est-il possible de créer un réseau d'ordinateurs comportant 15 machines et tel que chacune d'entre elles soit reliée à exactement trois autres ?

2. On considère un pays qui compte 101 villes. Des liaisons aériennes (aller-retour) existent entre certaines d'entre elles. La capitale du pays est reliée aux 100 villes et chaque autre ville possède 10 liaisons aériennes différentes. Est-il possible de supprimer la moitié des liaisons aériennes de la capitale tout en conservant la possibilité de voyager depuis n'importe quelle ville vers n'importe quelle autre ? *Indication : On pourra raisonner sur le graphe des liaisons aériennes dans lequel on a supprimé la capitale.*

3. Est-il possible de tracer une courbe fermée sans lever le crayon et qui coupe chacun des 16 segments de la figure ci-dessous exactement une fois ?



4. On considère des dominos dont les faces sont numérotées de 1 à $n \geq 1$. Pour quelles valeurs de n est-il possible d'arranger ces $n(n+1)/2$ dominos de sorte à former une boucle fermée ? *Indication : Faire le lien avec le graphe complet à n sommets.*

Exercice 4 *Implémentations de graphes*

1. Donner la matrice d'adjacence du graphe G_1 de l'exercice 1. Donner une représentation de ce graphe par listes d'adjacence. Reprendre ces deux questions avec le graphe G_4.

Les questions suivantes visent à décrire quelques opérations sur un graphe et à comparer leurs complexités selon l'implémentation choisie pour ce graphe : liste d'arêtes, listes d'adjacence ou matrice d'adjacence. Dans chacun de ces trois cas, on suppose que les sommets sont numérotés et que le nombre total de sommets est connu.

2. Pour chacune des trois implémentations, expliquer comment ajouter un sommet au graphe représenté. Reprendre la même question avec la suppression d'un sommet, l'ajout d'une arête et la suppression d'une arête (dans ces deux derniers cas, on pourra traiter séparément le cas d'un graphe non orienté).

3. Quelle est la complexité de chacun des 12 algorithmes proposés ?

La transposée d'un graphe orienté $G = (S, A)$ est le graphe $G^t = (S, \{(v, u) \,|\, (u, v) \in A\})$. Autrement dit, on inverse tous les arcs de G.

4. Décrire un algorithme permettant de calculer la transposée d'un graphe orienté lorsqu'il est représenté par matrice d'adjacence puis lorsqu'il est représenté par listes d'adjacence.

5. Estimer la complexité des deux algorithmes précédents. Dans quelles situations préférer l'une des implémentations par rapport à l'autre ?

Exercice 5 *Lecture de propriétés du graphe sur sa matrice d'adjacence*

On note M la matrice d'adjacence d'un graphe orienté G qui comporte n sommets.

1. Montrer que pour tout $p \in \mathbb{N}^\star$, le coefficient d'indice (i, j) dans M^p est égal au nombre de chemins de longueur p qui mènent de i à j.
2. Montrer que le graphe G contient un cycle si et seulement si $M^n \neq 0$.
3. Quel serait le coût d'un algorithme (à décrire) qui utiliserait cette propriété pour détecter un cycle dans G ?

Remarque : Ce n'est pas la première fois que l'on remarque que des propriétés d'un graphe se traduisent en des propriétés sur sa matrice d'adjacence. En revanche, si l'objectif est de détecter la présence d'un cycle, il est bien plus efficace d'utiliser l'exercice 8 que de calculer les itérées d'une matrice d'adjacence !

Exercice 6 *Parcours ou pas parcours ?*

1. Effectuer un parcours en profondeur de graphe G_1 à partir du sommet 2. On déterminera une arborescence pour ce parcours en plus d'indiquer l'ordre de visite des sommets.
2. Reprendre la question 1 avec un parcours en largeur.
3. Pour chacune des affirmations suivantes, indiquer si elle est vraie ou fausse en justifiant :
   a) Un parcours en profondeur de G_1 à partir de 6 est $(6, 3, 7, 5, 8, 1, 4, 0, 2)$.
   b) Un parcours en profondeur de G_1 à partir de 1 est $(1, 4, 0, 5, 7, 8, 6, 3)$.
   c) Un parcours en largeur de G_1 à partir de 2 est $(2, 0, 3, 1, 7, 5, 6, 8, 4)$.
   d) Un parcours en largeur de G_1 à partir de 2 est $(2, 3, 7, 6, 5, 8, 0, 1, 4)$.

Exercice 7 *Diamètre d'un graphe*

Le diamètre d'un graphe non orienté la longueur du plus long chemin acyclique qu'il est possible d'y trouver.

En exploitant l'algorithme de parcours en largeur, proposer un algorithme naïf permettant de calculer le diamètre d'un graphe et estimer sa complexité lorsque le graphe est représenté par listes d'adjacence.

Remarque : Vous avez déjà implémenté un algorithme de calcul de diamètre durant le TP12 dans le contexte particulier où les graphes considérés étaient des arbres binaires enracinés.

Exercice 8 *Détection de cycles*

Le bon fonctionnement de certains algorithmes (tri topologique, algorithme de Dijkstra...) exige que leur entrée soit un graphe ayant de "bonnes" propriétés vis-à-vis de leurs cycles. C'est une des raisons pour lesquelles on peut souhaiter avoir un algorithme permettant de les détecter.

1. Comment détecter simplement la présence d'un cycle dans un graphe non orienté connexe ?

On se place ci-dessous dans le cas d'un graphe orienté G. L'algorithme de parcours en profondeur vu en cours propose de considérer un sommet comme définitivement traité (donc colorié en noir) AVANT le traitement de ses voisins. On modifie légèrement l'algorithme de parcours en profondeur de sorte à ce qu'un sommet soit colorié en noir APRES le traitement de ses voisins. On obtient alors l'algorithme de détection de cycles suivant :

```
cycle(G) =
  Attribuer la couleur "blanc" à tous les sommets de G
  Pour chaque sommet s de G
  Si s est blanc
    visiter(G,s)
  Renvoyer "Pas de cycle"

visiter(G,s) =
  Colorier s en gris
  Pour chaque sommet u adjacent à s
    Si u est blanc
      visiter(G,u)
    Si u est gris
      Renvoyer "Présence de cycle"
  Colorier s en noir
```

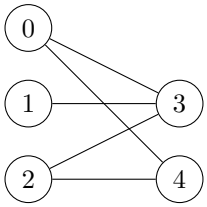
2. Expliquer pourquoi cet algorithme est correct.

3. Déterminer la complexité de la détection d'un cycle dans un graphe orienté.

4. Comment adapter cet algorithme de sorte à calculer un cycle s'il en existe un ?

Exercice 9 *Graphes bipartis*

Un graphe $G = (S, A)$ non orienté est dit biparti si on peut partitionner S en deux ensembles disjoints S_1 et S_2 tels que pour toute arête $(u, v) \in A$, $u \in S_1$ et $v \in S_2$. Par exemple, le graphe ci dessous est biparti (la partition $S_1 = \{0, 1, 2\}$ et $S_2 = \{3, 4\}$ convient):



1. Montrer qu'un arbre est un graphe biparti.

2. Montrer qu'un graphe est biparti si et seulement si toutes ses composantes connexes le sont.

Ce dernier résultat montre que pour détecter si un graphe est biparti, il suffit de travailler composante connexe par composante connexe. Sans perte de généralité, on peut donc supposer que G est connexe.

3. Montrer qu'un graphe connexe est biparti si et seulement si il ne contient aucun cycle de longueur impaire. *Indication : Pour la réciproque, on pourra raisonner par l'absurde et considérer un arbre couvrant du graphe.*

4. Concevoir un algorithme permettant de détecter si un graphe est biparti. *Indication : Utiliser un parcours !*

Remarque : Les graphes bipartis interviennent naturellement dès qu'on souhaite modéliser des relations entre deux classes d'objets différents. De manière générale, résoudre un problème de couplage est plus simple dans un graphe biparti que dans un graphe qui ne l'est pas et certains algorithmes permettant de résoudre ce problème ne fonctionnent que dans un graphe biparti.

Exercice 10 *Tri topologique*

1. Effectuer un tri topologique sur le graphe G_2.

Si G est un graphe orienté, on définit comme suit le graphe G' des composantes fortement connexes de G. Le graphe G' a autant de sommets que le nombre de composantes fortement connexes dans G. Les arêtes de G' sont les couples (U, V) de sommets de G' tels qu'il existe un sommet u dans la composante fortement connexe (associée à) U, un sommet v dans la composante fortement connexe V et une arête (u, v) dans G.

2. Montrer que le graphe des composantes connexes d'un graphe orienté est acyclique.

3. Effectuer un tri topologique sur le graphe des composantes connexes associé au graphe ci-dessous.