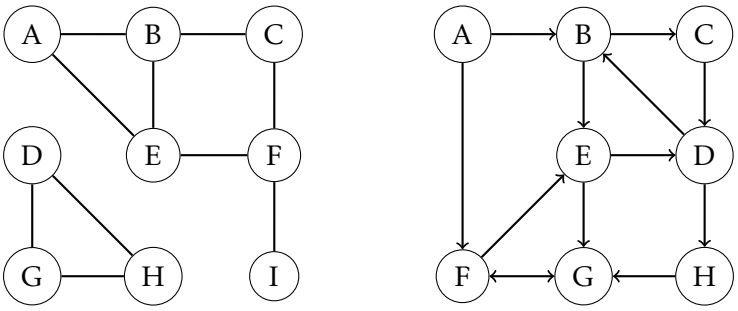
TD 4 & 5 Graphes : parcours

Exercice 1 (Parcours)**.**

1. Appliquez l'algorithme du parcours en largeur issu du sommet (A) sur les deux exemples suivants. Marquez chaque sommet avec son nombre `dist`.



2. Appliquez maintenant l'algorithme du parcours en profondeur. Quand plusieurs choix s'offrent à vous, vous choisirez les sommets dans l'ordre alphabétique. Vous marquerez chaque sommet avec ses valeurs `pre` et `post`.

Exercice 2 (Plus courts chemins entre ensembles)**.** Écrivez un algorithme qui prend en entrée un graphe orienté $G = (V, E)$ et deux ensembles de sommets $V_1, V_2 \subseteq V$, et renvoie un chemin de longueur minimale entre V_1 et V_2. La complexité de l'algorithme sera en $O(|V| + |E|)$.

Exercice 3 (Cycles)**.** Dans un graphe $G = (V, E)$, et pour un sommet $v \in V$, on appelle *cycle issu de* v un chemin $v = v_0 \to v_1 \to \cdots \to v_n = v$ dans G tel que les sommets $v_1, \ldots, v_{n-1}$ sont distincts deux à deux.

1. Écrivez un algorithme qui prend en entrée un graphe non orienté $G = (V, E)$, et renvoie `True` s'il existe un cycle dans G, et `False` sinon.
2. Même question pour un graphe non orienté.
3. Écrivez un algorithme qui prend en entrée un graphe orienté $G = (V, E)$ et un sommet $v \in V$, et renvoie, s'il existe, un cycle de longueur minimale passant par v ; ou `None` sinon. La complexité de l'algorithme devra être $O(|V| + |E|)$.
4. (Bonus) Même question mais avec G non orienté.

Exercice 4 (Connexité)**.** Dans un graphe non orienté, une *composante connexe* est un sous-ensemble maximal de sommets qui sont tous reliés par des chemins les uns aux autres.

1. Écrivez un algorithme prenant en entrée un graphe non orienté $G = (V, E)$ et renvoie la liste de ses composantes connexes. La complexité devra être en $O(|V| + |E|)$.
2. Écrivez un algorithme prenant en entrée un graphe non orienté $G = (V, E)$ ainsi qu'un sommet $v \in V$, et renvoie le nombre de sommets de la composante connexe contenant v.
3. Écrivez un algorithme prenant prend en entrée un graphe non orienté $G = (V, E)$ et renvoie la liste des sommets de sa plus grande composante connexe.

Exercice 5 (Distances)**.** Soit $G = (V, E)$ un graphe connexe non orienté. On note $d(u, v)$ la distance du plus court chemin entre deux sommets u et v.

1. Montrer que pour tout $u, v, w \in V$, $d(u, v) \leq d(u, w) + d(w, v)$.
2. Montrer que pour tout $u, v_1, v_2 \in V$, si $(v_1, v_2) \in E$, alors $|d(u, v_1) - d(u, v_2)| \leq 1$.
3. Montrer que G contient un cycle de longueur impaire si et seulement si, quelque soit $u \in V$, il existe $v_1, v_2 \in V$ tels que $(v_1, v_2) \in E$ et $d(u, v_1) = d(u, v_2)$.

Exercice 6 (Graphes bipartis)**.** Un graphe non orienté $G = (V, E)$ est *biparti* s'il existe une partition (V_1, V_2) de V telle que pour toute arrête $(u, v) \in E$, u et v ne sont pas dans la même partition (i.e. on a soit $u \in V_1$ et $v \in V_2$, soit $v \in V_2$ et $u \in V_1$). On dit aussi que G est 2-*coloriable*.

1. Montrer que si G est *biparti*, alors il ne contient pas de cycle de longueur impaire.
2. Montrer que si G ne contient que des cycles de longueur paire, alors G est biparti.
3. En utilisant les questions et l'exercice précédents, écrivez un algorithme qui prend en entrée un graphe non orienté $G = (V, E)$ et renvoie `True` si G est biparti, `False` sinon. La complexité de l'algorithme sera $O(|V| + |E|)$.
4. Écrivez un algorithme qui prend en entrée un graphe non orienté $G = (V, E)$ et renvoie, s'il existe, un 2-coloriage de G : autrement dit, renvoie une fonction $c : V \mapsto \{1, 2\}$ telle que $(u, v) \in E \implies c(u) \neq c(v)$.

Exercice 7 (Forte connexité)**.** Un graphe orienté $G = (V, E)$ est dit *fortement connexe* si pour toute paire de sommets $u, v \in V$, v est accessible par un chemin depuis u.

1. Étant donné un graphe orienté $G = (V, E)$, on définit le graphe transposé $G^T = (V, E^T)$ par $E^T = \{(v, u) \in V^2 \mid (u, v) \in E\}$ (autrement dit, on renverse les arêtes de G).

   Pour $G = (V, E)$ un graphe orienté, et $u \in V$ un sommet arbitraire, montrez que G est fortement connexe si et seulement si, pour tout sommet $v \in V$, il existe un chemin de u à v à la fois dans G et dans G^T.
2. À partir de la question précédente, écrivez un algorithme qui prend en entrée un graphe orienté $G = (V, E)$, et renvoie `True` si G est fortement connexe, `False` sinon. La complexité de l'algorithme sera $O(|V| + |E|)$.

Remarque : Cette dernière question est très similaire à l'algorithme de Kosaraju, qui permet de déterminer les composantes « fortement connexes » d'un graphe orienté en temps $O(|V| + |E|)$. Vous étudierez ces questions plus précisément au second semestre.