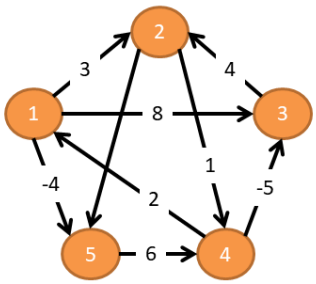
TD 6 : Plus courts chemin : Floyd-Warshall, Dijkstra, Bellman-Ford

1 Floyd-Warshall

1 **Fonction** FT_WARSHALL(G) ▷ `Algorithme de Warshall pour la fermeture transitive`
2 $A_0 \leftarrow G + Id$
3 **pour** chaque $k \in S[G]$ **faire**
4 **pour** chaque $i \in S[G]$ **faire**
5 **pour** chaque $j \in S[G]$ **faire**
6 $A_k[i,j] \leftarrow A_{k-1}[i,j]$ or $(A_{k-1}[i,k]$ and $A_{k-1}[k,j])$

L'algorithme de Warshall (voir si dessus) qui calcule la fermeture transitive et celui de Floyd-Warshall qui calcule les plus courts chemin pour tout couple de noeuds dans le graphe sont très similaire. Ces deux algorithmes se basent sur les chemins intermédiaires entre deux sommets (i,j) pour :
- Ajouter une arrête entre (i, j) si il a un chemin intermédiaire passant par k. (dans le cas de la fermeture transitive).
- Mettre à jour le plus court chemin entre (i, j), si le chemin passant par k est plus court. (dans le cas du calcule des plus courts chemins).

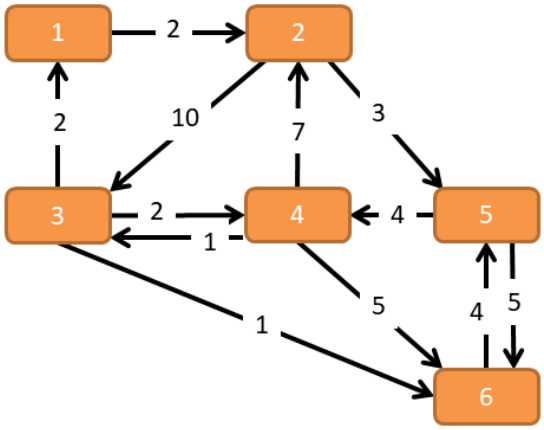
QUESTION 1 – Écrivez l'algorithme de Floyd-Warshall, puis donnez sa complexité.



QUESTION 2 – Appliquer l'algorithme de Floyd-Warshall sur le graphe ci dessus.

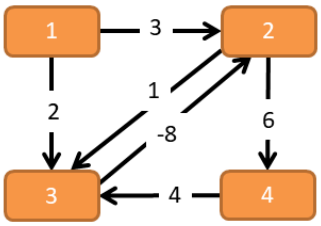
2 Dijkstra

1 **Fonction** DIJKSTRA($G, P, racine$)
2 **pour** chaque $u \in S[G]$ **faire**
3 $Q[u].cle \leftarrow \infty$ ▷ `Initialisation des clés en leurs attribuant une valeur infinie.`
4 $Q[u].parent \leftarrow NULL$ ▷ `Initialisation des plus courts chemins.`
5 $Q[racine].cle \leftarrow 0$ ▷ `On met à la racine du graphe la clé 0 car la distance pour aller vers elle même est nulle.`
6 $F \leftarrow S[G]$ ▷ `Initialisation de la file de priorité min. Elle est implémentée avec un tas binomial basé sur la clé des sommets.`
7 **tant que** $F \neq \emptyset$ **faire**
8 $u \leftarrow EXTRAIRE\_MIN(F)$
9 **pour** chaque $v \in Adj[u]$ **faire**
10 $poids\_chemin \leftarrow P(u, v) + Q[u].cle$
11 **si** $v \in F$ et $poids\_chemin < Q[v].cle$ **alors**
12 $Q[v].parent \leftarrow u$
13 $Q[v].cle \leftarrow poids\_chemin$
14 $DIMINUER\_CLE(v, poids\_chemin)$



QUESTION 3 – Appliquez l'algorithme de Dijkstra sur le graphe ci-dessus.

QUESTION 4 – Pouvons-nous remplacer la ligne 7 de l'algorithme de Dijkstra par **tant que** $F > 1$ dans le but d'économiser une itération ?
Justifiez.



QUESTION 5 – Que constatez vous en appliquant l'algorithme de Dijkstra sur le graphe ci-dessus ? Pourquoi ?

QUESTION 6 – Soit $G = (S, A)$ un graphe orienté pondéré par la fonction $fiabilite : (u, v) \rightarrow [0; 1]$, avec $(u, v) \in A$. Cette fonction représente la fiabilité du canal de communication entre le sommet u et v. Autrement dit, $fiabilite(u, v)$ est la probabilité pour que la communication entre u et v ne soit pas interrompue. On suppose que ces probabilités sont indépendantes. Donnez un algorithme permettant de trouver le chemin le plus fiable entre 2 sommet donnés. Écrivez le pseudo code de cet algorithme.

3 Bellman-Ford

L'algorithme de Bellman-Ford permet de calculer les plus courts chemins depuis une source unique même si le graphe comprend des cycles négatifs.

QUESTION 7 – Appliquez l'algorithme de Bellman-Ford sur le graphe de la question 5. Quelle partie de l'algorithme détecte le problème soulevé dans la question 5 ?

QUESTION 8 – Quand privilégier Dijsktra à Bellman-Ford et pourquoi ?