# Réponses à la question n°1 :

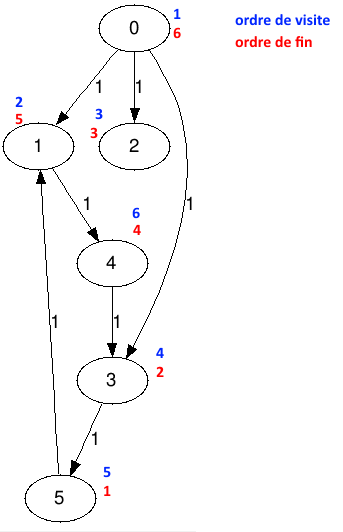
Premier cas et deuxième cas :

Les deux premiers algorithmes notifient tous les voisins d’un sommet en cours de visite comme visité puis parcourent en profondeur le dernier qui a été ajouté à la pile, au lieu de notifier un seul voisin comme visité puis de le parcourir en profondeur.

Conséquences :

* Dans certains cas, on ne visitera pas un voisin en pensant qu’il a déjà été visité, alors qu’on ne l’a pas parcouru en profondeur, ou qu’on n’est pas en train de le parcourir en profondeur.
* Il fournit un ordre suffixe faux, et donc un tri topologique faux.

Sur cette image, on peut voir l’ordre de visite des deux premiers algorithmes. On peut voir que le tri topologique obtenu par l’inverse de l’ordre de fin est faux.



Troisième cas :

La pile peut être plus grande que O(n) puisqu’à chaque itération, on visite un sommet et on ajoute les voisins non visités dans la pile, autrement dit, on ajoute tous les autres embranchements possibles ce qui peut être plus grand que n (le nombre de sommets dans le graphe).

Par exemple, si on a un graphe fortement connexe de 100 sommets, le premier sommet visité ajoutera 99 sommets non visités dans la pile. Le deuxième ajoutera 98 sommets non visités dans la pile. Le troisième 97 etc..

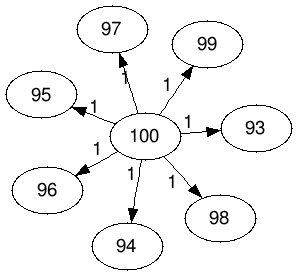
On atteint très vite les 500 éléments dans la pile, qui va donc dépasser 1000.

D’après le test effectué dans la classe DFS, on peut voir que pour un graphe fortement connexe de 100 sommets, la pile peut aller jusqu’à une taille de 5120.

Quatrième cas :

Le temps d’exécution est supérieur à O(n+m). En effet, lorsqu’on dépile, on revient sur un sommet déjà parcouru, on va alors reparcourir toutes ses arêtes, et certaines seront parcourues plusieurs fois.

Si on prend un graphe de 100 sommets tel qu’un sommet soit relié aux 99 sommets, le parcours en profondeur parcourra un grand nombre de fois certaines arêtes.

Un graphe de la forme suivante.