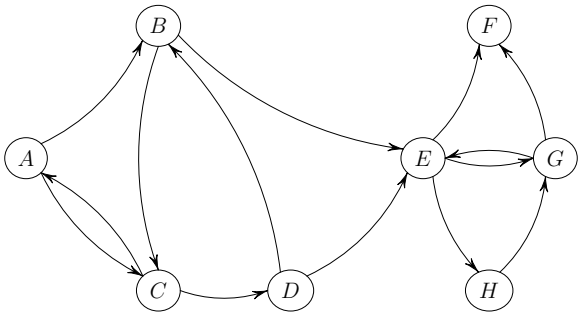
Feuille de TD n°4

Exercice 1. Appliquez l'algorithme de recherche des composantes fortement connexes au graphe ci-dessous, en détaillant les parcours en profondeur effectués.



Exercice 2. Le but de cet exercice est de démontrer la validité de l'algorithme de calcul des composantes fortement connexes étudié en cours.

1. Montrer que si deux sommets se trouvent dans la même composante fortement connexe, tous les chemins de l'un à l'autre restent dans cette composante fortement connexe.

On définit *l'aïeul* $\phi(u)$ d'un sommet u comme étant le sommet qui, parmi ceux accessibles à partir de u, termine le premier parcours en profondeur en dernier.

2. Montrer que $\phi(u)$ est un ancêtre de u selon le premier parcours en profondeur (c'est-à-dire qu'il existe un chemin rouge [1] de $\phi(u)$ vers u).

3. Montrer que u et v sont dans la même composante fortement connexe si et seulement si $\phi(u) = \phi(v)$.

Soit u le sommet par lequel on a fini le premier parcours en profondeur. C'est aussi celui par lequel on commence le second parcours.

4. Montrer que $\phi(u) = u$.

Soit v un sommet dont u est un ancêtre dans le deuxième parcours en profondeur.

5. Montrer que $\phi(v) = u$. En déduire que v et u sont dans la même composante fortement connexe.

6. Montrer que l'algorithme calcule les composantes fortement connexes.

Exercice 3. Trouver un algorithme qui détermine si un graphe orienté contient un cycle.

Exercice 4. Trouver un exemple de graphe ayant des arêtes de poids négatif (mais sans cycle de poids négatif) pour lequel l'algorithme de Dijkstra donne un résultat faux.

Pour traiter le cas des graphes ayant des arêtes de poids négatif, on considère l'algorithme de Bellman-Ford, qui calcule les plus courts chemin du sommet s à chacun des sommets accessibles depuis s, dans un graphe ne contenant pas de cycle de poids négatif :

1: initialiser s à 0 et les autres sommets à $+\infty$.
2: **pour** $i = 1$ **à** $|S(G)| - 1$ **faire**

1. C'est-à-dire formé d'arêtes que l'on a suivies au cours du parcours. Ce n'est pas une terminologie standard.

```
    pour chaque arc (u, v) faire
        si valeur(v) > valeur(u) + poids(u, v) alors
            valeur(v) = valeur(u) + poids(u, v)
        fin si
    fin pour
fin pour
pour chaque arc (u, v) faire
    si valeur(v) > valeur(u) + poids(u, v) alors
        retourner faux
    fin si
fin pour
retourner vrai
```

Exercice 5. Exécutez cet algorithme sur le contre-exemple que vous avez trouvé à l'exercice 4. Que constatez-vous ?

Exercice 6. Dans quel cas cet algorithme renvoie-t-il « faux » ?

Exercice 7. Soit v un sommet. Montrer que s'il existe un plus court chemin de s à v en au plus k étapes, alors après la k-ième étape dans la première boucle, la valeur de v est égale à la distance minimale entre s et v. En déduire la validité de l'algorithme de Bellman-Ford.

Exercice 8. Calculer la complexité de cet algorithme.