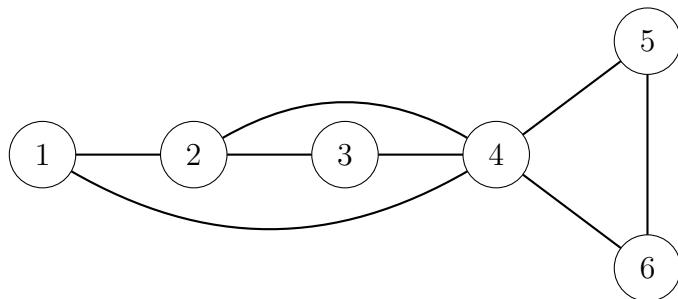


Exercice 1

Dans cet exercice nous souhaitons utiliser l'algorithme PP du parcours en profondeur pour déterminer les points d'articulation d'un graphe non-orienté. Un *point d'articulation* est un sommet dont la suppression augmente le nombre de composantes connexes du graphe.

1. Déterminer les points d'articulation du graphe G suivant :



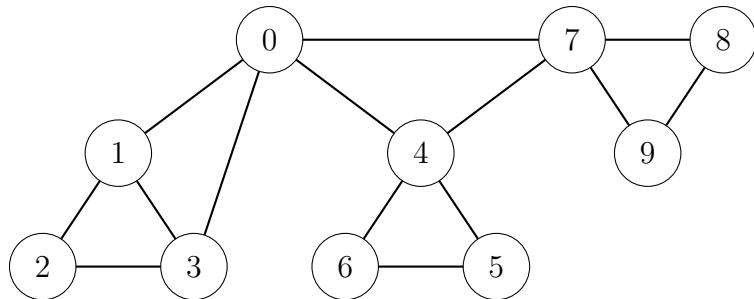
2. Modifier $PP(G)$ en $NB_CC(G)$ afin de compter le nombre de composantes connexes du graphe G . En utilisant $NB_CC(G)$ écrire un algorithme calculant les points d'articulation du graphe G . Calculer sa complexité.

Dans la suite, nous allons mettre en place un algorithme plus efficace de calcul des points d'articulation. Pour cela, on fixe un sommet r et on considère l'arborescence de liaison $T(r)$ définie par le parcours en profondeur de G à partir de r . Les arcs de *liaisons* sont les arcs (u, v) de G où $u = \text{pere}(v)$ c'est-à-dire les arcs de $T(r)$, tandis que les arcs de *retour* sont les arcs (u, v) de G qui ne sont pas de liaison et où v est un ancêtre de u dans $T(r)$. On peut montrer que dans un PP sur un graphe non-orienté, les arcs sont soit de liaison soit de retour.

Pour tout sommet v , on définit $1[v]$ comme la plus petite valeur de $d[u]$ où u est soit égal à v , soit l'extrémité d'un arc retour (w, u) avec w descendant de v ($w=v$ possible).

3. Calculer l'arbre $T(r)$ pour G en prenant $r=1$. Distinguer les arcs de retour, et calculer $1[u]$ pour tous les sommets u de G .
4. Modifier l'algorithme PP pour calculer $1[v]$ pour tout sommet v d'un graphe G .
5. Établir que v est un point d'articulation si et seulement si :
 - soit $v = r$ et v a au moins deux fils dans $T(r)$,
 - soit $v \neq r$ et v a au moins un fils w dans $T(r)$ tel que $1[w] \geq d[v]$.
6. Modifier l'algorithme PP pour calculer, pour tout sommet v d'un graphe, $pa[v]$ qui vaut *vrai* si et seulement si v est un point d'articulation. Comparer la complexité de cet algorithme avec celle de l'algorithme de la question 2.

7. Tester l'algorithme modifié sur le graphe G ci-dessus et sur le graphe obtenu en ajoutant à G l'arête $\{6, 3\}$.
8. Tester l'algorithme sur le graphe H ci-dessous, en partant du sommet $r=0$ et en suivant l'ordre croissant des sommets.



RAPPEL :

```

0  PP(G)                                0  Visiter_PP(u)
1  pour chaque sommet u de X faire       1  couleur[u] <- GRIS
2    couleur[u] <- BLANC                2  d[u] <- temps <- temps + 1
3    pere[u] <- nil                     3  pour chaque v de Adj[u] faire
4    temps <- 0                           4    si couleur[v] = BLANC
5    pour chaque sommet u de X faire       5    alors pere[v] <- u
6      si couleur[u] = BLANC             6    Visiter_PP(v)
7        alors Visiter_PP(u)             7  couleur[u] <- NOIR
                                         8  f[u] <- temps <- temps + 1

```