Exercice 1

Un graphe non orienté est *biparti* si l'on peut partager l'ensemble de ses sommets en deux sous-ensembles tels qu'aucune arête du graphe relie deux sommets appartenant au même sous-ensemble.

Proposer un algorithme efficace permettant de déterminer si un graphe non orienté est biparti. Quelle est sa complexité ?

Exercice 2

```
PP(G)
  pour chaque sommet u de X faire
      couleur[u] <- BLANC
      pere[u] <- nil
  temps <- 0
  pour chaque sommet u de X faire
      si couleur[u] = BLANC
          alors Visiter_PP(u)
```

```
Visiter_PP(u)
  couleur[u] <- GRIS
  d[u] <- temps <- temps + 1
  pour chaque v de Adj[u] faire
      si couleur[v] = BLANC
          alors pere[v] <- u
                Visiter_PP(v)
  couleur[u] <- NOIR
  f[u] <- temps <- temps + 1
```

Appliquer l'algorithme de parcours en profondeur PP au graphe G_1 de la Fig. 1 (on conviendra que dans les listes d'adjacence les sommets sont rangés dans l'ordre croissant de leur numéro) :

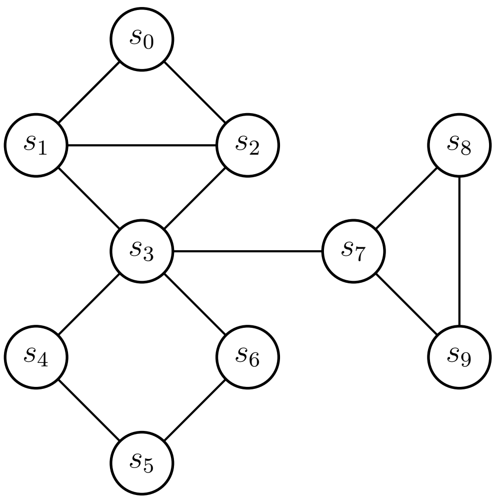

Figure 1 – G_1

Exercice 3

1. Modifier l'algorithme PP pour compter le nombre de sommets de la composante connexe d'un sommet donné d'un graphe non orienté. Peut-on obtenir le même résultat en utilisant le parcours en largeur ?
2. Modifier l'algorithme PP pour tester si un graphe non orienté possède un cycle.

Exercice 4

Pour les deux graphes non orientés H_1 et H_2 ci-dessous dire si T_1 et T_2 peuvent respectivement être des arbres couvrants obtenus par un parcours en profondeur – Justifier les réponses en spécifiant l'éventuel sommet de départ et l'ordre des sommets dans les listes d'adjacence.

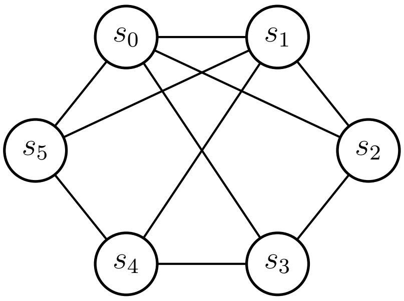

FIGURE 2 – H_1

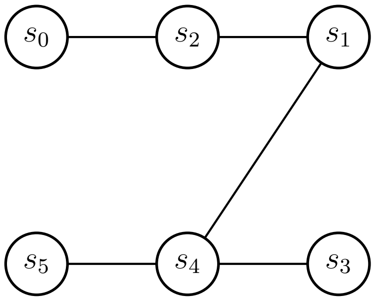

FIGURE 3 – T_1

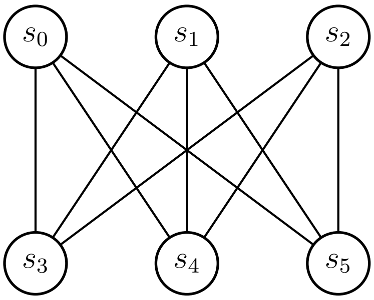

FIGURE 4 – H_2

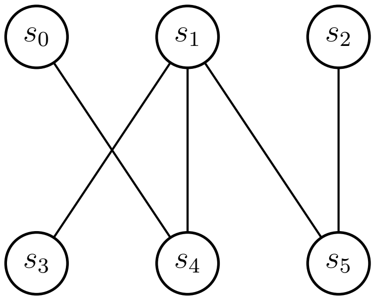

FIGURE 5 – T_2