# TD1 : structures de données pour les graphes

## 1 Matrice d'adjacence et listes d'adjacence

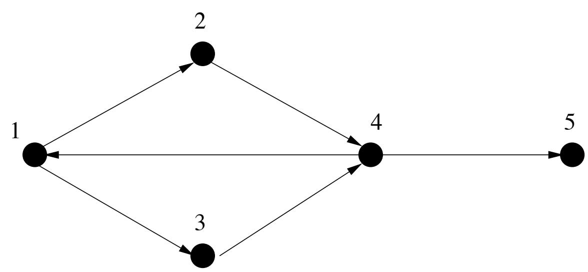
1. Considérer le graphe de la figure ci-dessous. Existe-t-il un chemin du sommet 1 au sommet 5 ? Et du sommet 5 au sommet 1 ?[1]

2. Donner les représentations de ce graphe par **matrice d'adjacence** et par **listes d'adjacence** (aussi appelées, surtout dans le cas orienté, **listes de successeurs**).

3. Mêmes questions pour le graphe non orienté obtenu en omettant l'orientation des arcs.

4. Etant donné un graphe $G$ avec $n$ sommets et $m$ arcs, quelle est la taille en mémoire de chacune de ces structures de données ? Mêmes questions dans le cas non orienté.

5. Ecrire une fonction qui teste l'adjacence entre deux sommets $x$ et $y$ pour chacune de ces structures de données. Discuter le temps d'exécution dans chaque cas.




## 2 Combinatoire, arbres, connexité

On notera désormais par $n$ le nombre de sommets d'un graphe et par $m$ le nombre de ses arcs où arêtes, selon qu'il est orienté ou pas.

1. Montrer que dans tout graphe orienté on a $m \leq n(n-1)$ et que dans tout graphe non orienté, $m \leq n(n-1)/2$. Donner des exemples où ces inégalités sont atteintes.

2. Rappelons qu'un **arbre** (non orienté) est un graphe connexe sans cycle.

   (a) Montrer que tout arbre possède une *feuille*, c-à-d un sommet ayant au plus un voisin.

   (b) Montrer que dans tout arbre, $m = n - 1$.

   (c) En déduire un algorithme qui reconnaît si un graphe donné est un arbre.

3. Montrer que dans tout graphe connexe il existe un sommet que l'on peut supprimer sans détruire la propriété de connexité.

[1] Les algorithmes de recherche de chemins dans les graphes feront l'objet des prochaines séances de TD.