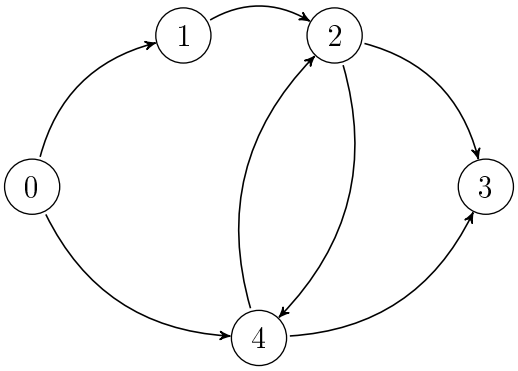
fiche 9

Graphe

1 Représentation par matrice d'adjacence

La matrice d'adjacence est une matrice booléenne (carrée et d'ordre n) telle que $m.(i).(j)$ vaut true si et seulement si on a un arc du sommet i au sommet j.



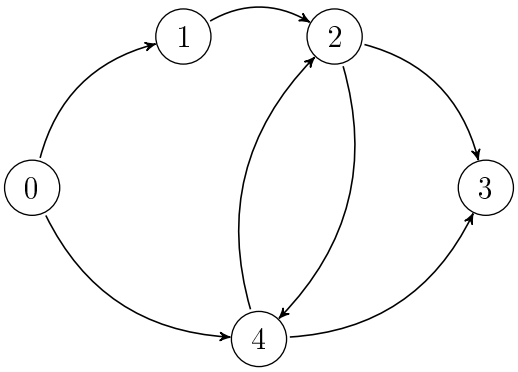
1. Définir l'exemple avec sa matrice d'adjacence.
2. Écrire deux fonctions `taille` et `nb_arcs` comptant respectivement le nombre de sommets et le nombre d'arcs d'un graphe.
   Exemple :

```
>>> taille(exemple_1)
5
>>> nb_arc(exemple_1)
7
```

3. Écrire les fonctions `ajout_arc` et `supprime_arc` :
   - La fonction `ajout_arc` prend en argument un graphe et les sommets i et j, et ajoute l'arc (i, j). (La fonction ne renvoie rien.)
   - La fonction `supprime_arc` prend en argument un graphe et les sommets i et j, et supprime l'arc (i, j).(La fonction ne renvoie rien.)
4. Écrire la fonction `supprime_sommet` qui rend inaccessible le sommet i du graphe.(la fonction ne renvoie rien).
5. Écrire la fonction `desoriente` qui transforme un graphe orienté en graphe non orienté. La fonction prend en argument un graphe orienté, et renvoie un graphe non orienté.
6. Écrire la fonction `non_oriente` qui renvoie vrai si le graphe en argument est un graphe non orienté.

2 Représentation par listes d'adjacence

Les listes d'adjacence sont représentées par un tableau v de listes d'entiers tel que $v[s]$ contient la liste des voisins du sommet s.



1. Définir l'exemple avec ce type.
2. Écrire deux fonctions `taille` et `nb_arcs` comptant respectivement le nombre de sommets et le nombre d'arcs d'un graphe.
3. Écrire les fonctions `ajout_arc` et `supprime_arc` :
   - La fonction `ajout_arc` prend en argument un graphe et les sommets i et j, et ajoute l'arc (i, j).
   - La fonction `supprime_arc` prend en argument un graphe et les sommets i et j, et supprime l'arc (i, j).
4. Écrire la fonction `supprime_sommet` qui rend inaccessible le sommet i du graphe.
5. Écrire la fonction `desoriente` qui transforme un graphe orienté en graphe non orienté.
6. Écrire les fonctions `mat_of_liste` et `liste_of_mat` :
   - La fonction `mat_of_liste` renvoie la matrice d'adjacence du graphe dont on connait les listes d'adjacence.
   - La fonction `liste_of_mat` renvoie les listes d'adjacence du graphe dont on connait la matrice d'adjacence.

3 Création d'un graphe à l'aide de classe

On propose d'implémenter un graphe à l'aide des classes suivantes :

```
class Sommet :
    def __init__ (self,etiquette = None):
        self.etiquette = etiquette

class Arc :
    def __init__ ( self , depart , arrivee ) :
        self.origine = depart # de la classe Sommet
        self.arrivee = arrivee # de la classe Sommet

class Graphe :
    """ stock un graphe oriente """
    def __init__ ( self, nom = None):
        self.nom = nom
        self.list_sommet = []
        self.list_arc = []
```

1. **Pour la classe `Arc` :**
   (a) Écrire la méthode spéciale `__str__` pour afficher le début puis la fin d'un arc.
   (b) Écrire la méthode setter pour changer la finalité d'un arc.
2. **Pour la classe `Graphe` :**
   (a) Écrire la méthode `ajoute_sommet` qui ajoute un sommet au graphe.
   (b) Écrire la méthode `appartient` qui vérifie l'appartenance d'un sommet à un graphe.
   (c) Écrire la méthode `ajoute_arc` qui ajoute un arc au graphe pour deux sommets donnés (on utilisera des assertions pour vérifie la possibilité de construction de l'arc).
3. **Construction d'un graphe.**
   (a) Écrire la fonction `mat_of_tab` qui renvoie le graphe pour une matrice d'adjacence donnée.
   (b) Écrire la fonction `mat_of_liste` qui renvoie le graphe pour une liste d'adjacence donnée.