	**Langages et Programmation**	**NSI T^{ale}**
	Représentation des Graphes la liste d'adjacence	Cours/TD

1. Introduction

L'objectif de ce TP est d'écrire quelques premières fonctions sur les graphes, et en particulier les fonctions de conversions.
Une classe `Tableau` (pour le tableau des degrés des sommets d'un graphe) et une classe Matrice (pour les matrices d'adjacences) sont fournies. Observer le choix qui a été fait pour gérer les indices, puisque les sommets d'un graphe sont numérotés de {1,...,n} alors que les indices en Python commencent à 0.
Enfin, la classe **`Liste_adjacence`** est à compléter.

2. Graphe non orienté

Dans toute la suite, nous considérons que `G` est un graphe non orienté. Les plus rapides pourront essayer d'adapter ces fonctions au cas des graphes orientés.

→ Compléter la classe Liste_adjacence

La classe **`Liste_adjacence`** vous est proposée pour représenter les graphes par leurs listes d'adjacence via l'attribut `listA`. Remarquez que celui-ci est construit de telle sorte que `G.listA[i]` est la liste d'adjacence du sommet `i` du graphe `G`. Il n'y a par convention, pas de sommet `0`, donc pour tout graphe `G`, `G.listA[0]` est une liste vide.

Compléter la définition de cette classe en y ajoutant :

- `nbSommets(self)` : retourne le nombre de sommets du graphe `G`
- `nbAretes(self)` : retourne le nombre d'arêtes du graphe `G`
- `ajoutArete(self,i,j)` : ajoute l'arête `(i, j)` au graphe `G`
- `enleveArete(self,i,j)` : enlève une arête `(i, j)` du graphe `G` (si elle existe !)
- `getAretesSommet(self, i)` : retourne la liste d'adjacence du sommet `i` du graphe `G`
- `degSommet(self,i)` : retourne le degré du sommet `i`
- `degre(self)` : retourne un `Tableau D` tel que `D.tab[i]` est le degré du sommet `i`

→ Conversions entre représentations

Dans le programme principal, écrire les fonctions de conversions suivantes :

- `listeToMatrice(G)` : retourne la matrice d'adjacence représentant `G` donné par ses listes d'adjacences.Nous utiliserons la classe Matrice.
- `areteToListe(n,L)` : retourne les listes d'adjacence (utiliser la classe `Liste_adjacence`) représentant un graphe donné par sa liste `L` d'arêtes.
- `matToListe(M)` : retourne les listes d'adjacences représentant un graphe donné par sa matrice d'adjacence `M`.