Ce devoir couvre la notion *d'isomorphisme entre graphes* (référez-vous au Chapitre 1 des notes de cours). Par souci de simplicité, nous considérerons ici des paires de graphes avec des nombres de noeuds égaux.

Q1 Implémentez un algorithme de recherche exhaustive pour décider si deux graphes sont isomorphes.

- **Données**: Les graphes vous sont donnés sous forme de matrices d'adjacence. Elles vous sont fournies dans un fichier **.csv** dont la structure est décrite en annexe.
- **Sorties**: Pour une paire de matrices d'adjacence, vous devez:
  - Décider si oui ou non les graphes associés sont isomorphes.
  - Le cas échéant, présenter un isomorphisme sous forme d'un vecteur v tel que $v[i] == j$ si le noeud i du premier graphe correspond au noeud j du second.

Pour un graphe aux noeuds V et arêtes E, considérez une fonction

$$\mathit{color} : V \times E \to \mathbb{N}^V$$

avec les propriétés suivantes: pour tout graphe (V', E') isomorphique à (V, E), si $V' = p(V)$ pour une permutation p, alors $\mathit{color}(V', E') = p(\mathit{color}(V, E))$.
Il existe une infinité de telles fonctions, par exemple:

- La fonction associant la valeur 1 à chaque noeud, i.e.,

$$\forall v \in V : \mathit{color}(V, E)[v] = 1.$$

- La fonction associant à chaque noeud son degré, i.e.,

$$\forall v \in V : \mathit{color}(V, E)[v] = \mathit{deg}(v).$$

Intuitivement, l'accès à une telle fonction peut aider lors de la recherche d'un isomorphisme. En effet, afin qu'une permutation p corresponde à un ismorphisme entre deux graphes (V, E) et (V', E'), il faut:

$$(u, v) \in E \Leftrightarrow (p(u), p(v)) \in E' \quad (1)$$

$$\forall v \in V : \mathit{color}(V, E)[v] = \mathit{color}(V', E')[p(v)] \quad (2)$$

Ainsi, (2) peut nous permettre de réduire drastiquement l'espace des fonctions p à investiguer lors de la recherche d'isomorphismes.

Nous souhaitons exploiter ces outils à l'aide d'une heuristique. Mais d'abord, quelques notations.
On dit qu'une fonction $p : V \to V' \cup \{\}$ est *partiellement définie* si il existe $v \in V$ tel que $p(v) = \{\}$. Elle est définie si $p(V) = V'$. On utilise la notation $p' = p \cup (v \mapsto v')$ pour décrire la fonction $p'(u) = p(u)$ pour $u \neq v$, $p'[v] = v'$.

L'heuristique construit un isomorphisme p à partir d'une fonction p_0 tel que $p_0(v) = \{\}\, \forall v$. Elle se résume en l'évaluation de la fonction $\mathrm{Isom}_{\mathrm{color}}((V, E), (V', E'), h)$, définie à l'algorithme 1, au point $h = p_0$.

Algorithm 1 Calcul de $\text{Isom}_{\text{color}}((V,E),(V',E'),h)$

```
if h ne satisfait pas (1) ou (2) then            ▷ Evaluez (1), (2) aux noeuds définis.
    return False
else if h est définie then         ▷ h est un isomorphisme, il a passé tous les tests.
    return True
end if
for Paires v ∈ V, v' ∈ V' de la même couleur. do
    if h(v) = {} et v' ∉ Im(h) then
        Définir h' = h ∪ (v ↦ v').
        if Isom_color((V,E),(V',E'),h') == True then
            return True
        end if
    end if
end for
return False                   ▷ On ne peut trouver un isomorphisme à partir de h.
```

Q2 Implémentez un algorithme permettant de calculer $\text{Isom}_{\text{color}}((V,E),(V',E'),h)$ pour toute paire de graphes, toute fonction color, et fonction h. (Vérifier que cela fonctionne bien en utilisant une fonction color assignant 1 à tous les noeuds, et comparant avec votre recherche exhaustive.)

Q3 Comparer les temps d'exécutions pour les fonctions suivants:

- $\text{color}(V,E)[v] = 1$,
- $\text{color}(V,E)[v] = deg(v)$.

Q4* **(pour EPL, facultatif pour les Maths):** On peut considérer des fonctions de coloration dont le codomaine est autre que $\mathbb{N}^V$. Considérez la fonction suivante:

$$\text{color}(V,E)[v] = \Big\{(k, deg(v'))|v' \in Neigh(v,k), k \in \{0\dots n\}\Big),$$

où $Neigh(v,k)$ est le voisinage de v à distance k. Implémentez l'heuristique sur base de cette fonction.

Input CSV

La forme de l'input est simple: pour une paire de graphes avec n noeuds chacuns, les matrices sont encodées dans un fichier comprenant $2n$ lignes. Chaque ligne contient n chiffres (0 ou 1), séparés par des virgules, et est terminée par un retour à la ligne ("$\backslash n$").

Voici le contenu d'un fichier représentant 2 graphes à 4 noeuds. Ces deux graphes sont isomorphiques.

```
0, 1, 1, 1
1, 0, 1, 0
1, 1, 0, 1
1, 0, 1, 0
0, 1, 0, 1
1, 0, 1, 1
0, 1, 0, 1
1, 1, 1, 0
```