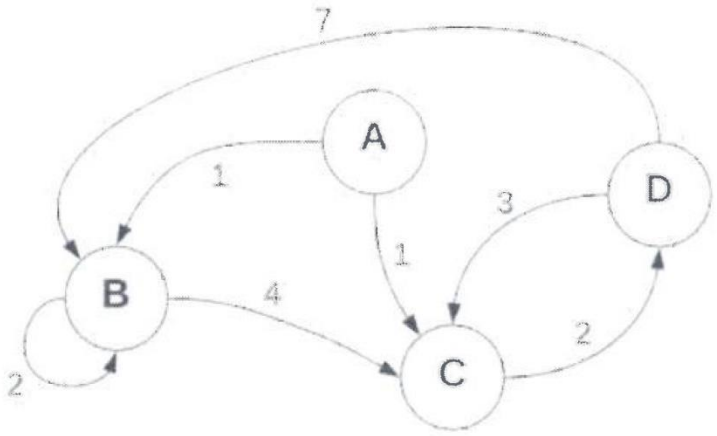
Exercice 1 : **Manipulation de dictionnaire**

Q1. Créer le dictionnaire ci-contre avec les 5 numéros de téléphone ;
Q2. Ajouter votre numéro de téléphone ;
Q3. Modifier le numéro de Jamel qui devient : 0606060606 ;
Q4. Supprimer le numéro de Jimmy ;
Q5. Afficher le numéro de Jacques ;
Q6. Afficher la liste des noms des contacts (sans le numéro) ;
Q7. Afficher à qui appartient le numéro 0623456789.

Jean	0612345678
Josephine	0623456789
Jamel	0634567890
Jacques	0645678901
Jimmy	0656789012

Exercice 2 : **Analyse de graphe**

On considère le graphe ci-dessous :



Q 1 : Vrai ou faux :
- Il s'agit d'un graphe orienté.
- Il possède 7 arcs.
- Il possède 7 arêtes.
- Il possède 5 sommets

Q 2 : Déterminer :
- $d_+(C)$
- $d_-(C)$
- $d(B)$

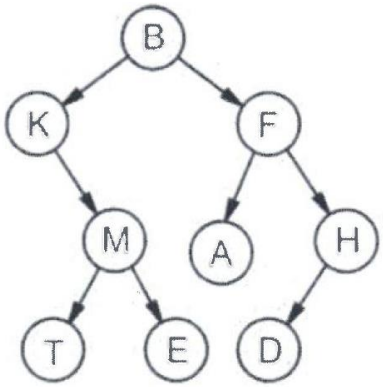
Q 3 : On propose pour ce graphe la matrice d'adjacence ci-dessous. Corriger les éventuelles erreurs.

$$\begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 2 & 4 & 7 \\ 1 & 4 & 0 & 5 \\ 0 & 7 & 5 & 0 \end{pmatrix}$$

Q 4 : On décide de négliger les poids des liens de ce graphe, qui devient non pondéré. Quelle est sa liste d'adjacence ?

Exercice 3 : **Parcours**

On considère le graphe ci-dessous (de type arbre orienté) :



On utilise comme sommet de départ le sommet « B ».

Q 1 : Préciser l'ordre dans lequel les sommets sont explorés lors d'un parcours en largeur ?

Q 2 : Préciser l'ordre dans lequel les sommets sont explorés lors d'un parcours en profondeur ?

Exercice 4 : **Existe-t-il un voison non connu ?**

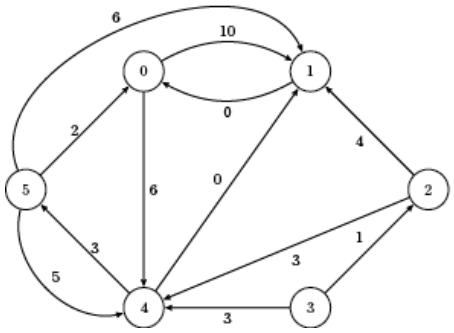
On considère un graphe défini en mémoire par une liste d'adjacence `L`.
Un programme de parcours en profondeur s'appuie sur une liste `noeuds_connus`, et sur un entier `noeud_courant` designant un nœud particulier.

Rédiger les lignes du programme python correspondant aux étapes algorithmiques :

si il existe un voisin du noeud courant non connu
alors ajouter le voisin `a la liste noeuds connus ;

Exercice 5 : **Plus court chemin**

On considère le graphe ci-dessous et sa matrice d'adjacence M.



$$\begin{bmatrix} 66 & 10 & 66 & 66 & 6 & 66 \\ 0 & 66 & 66 & 66 & 66 & 66 \\ 66 & 4 & 66 & 66 & 3 & 66 \\ 66 & 66 & 1 & 66 & 3 & 66 \\ 66 & 0 & 66 & 66 & 66 & 3 \\ 2 & 6 & 66 & 66 & 5 & 66 \end{bmatrix}$$

Q 1 : Exécuter l'algorithme de Dijkstra en partant du sommet 0.

Q 2 : Exécuter l'algorithme de Dijkstra en partant du sommet 3.

Q 3 : Existe-t-il un chemin de 4 à 2 ?