R. Malgouyres, R. Zrour et F. Feschet
Initiation à lalgorithmique et à la programmation en C,
Cours avec 129 exercices corrigés, 3e édition,
DUNOD, Collection Sciences Sup, 2014,
Nouvelle présentation 2015.

Algorithmique et programmation en *C*

TP n° 15
Graphes

Objectifs :
Le but du TP est de construire des chemins dans un réseau informatique. On représentera le réseau sous forme d'un graphe avec matrices d'adjacence. On dessinera e graphe sous en utilisant l'utilitaire linux dotty graphviz.

Le réseau intranet d'une grande entreprise est formé de noeuds, reliés entre eux par des cables. On suppose qu'entre deux noeuds donnés ne passe qu'un seul cable, qui est orienté.

Exercice 1 Proposer un format de fichier pour décrire un réseau et écrire une fonction de chargement du réseau en mémoire centrale sous forme de listes d'adjacence. On suppose que chaque noeud du réseau est identifié par un nom et une adresse IP.

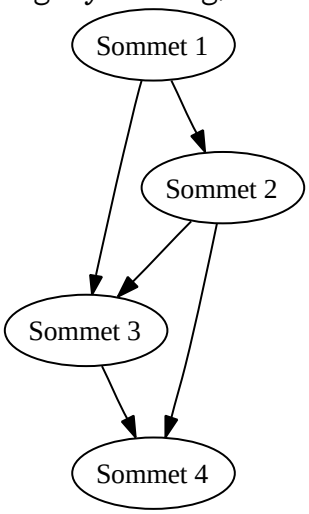
Exercice 2 Le format texte `dot` permet de représenter un graphe, puis de le visualiser grace à l'utilitaire `dotty`. Pour cela, on exportera son graphe au format dot suivant :

- La premire ligne du fichier contient les deux mots `digraph family` ;
- La deuxième ligne contient une accolade ouvrante ;
- Vient ensuite la liste des sommets, avec un sommet sur chaque ligne. Chaque sommet est déclaré par un `ID`, et entre crochet un `label` qui sera affiché avec le sommet.
- Vient ensuite la liste des arcs, chaque arc étant donné par `ID1 -> ID2`.
- La dernière ligne contient une accolade fermante.

```
digraph family
{
1 [label="Sommet 1"]
2 [label="Sommet 2"]
3 [label="Sommet 3"]
4 [label="Sommet 4"]

1->2
2->3
1->3
2->4
3->4

}
```



On fera un appel système pour exécuter la commande bash `dotty fichier.txt` qui dessinera le graphe.

Exercice 3 Écrire une fonction qui détermine si deux noeuds peuvent communiquer dans le réseau, ou bien si la configuration des cables rend cette communication impossible. On pourra utiliser un parcours en profondeur récursif.

Exercice 4 Écrire une fonction qui, étant donnés deux noeuds du réseau, affiche le plus court chemin (c'est à dire celui qui a le moins de connexions) entre les deux noeuds (si un tel chemin existe !). On pourra utiliser un parcours en largeur.