TD graphes et algorithme d'itinéraire

Graphes

Dessinez ci-dessous un graphe à 5 sommets avec le maximum d'arêtes:

Combien y a-t-il d'arêtes dans ce graphe ? ...
Quelle est la distance maximale et minimale entre deux sommets ? ... et ...
Dessinez maintenant un graphe connecté à 5 sommets avec le minimum d'arêtes:

Combien y a-t-il d'arêtes dans ce graphe ? ...
Quelle est la distance maximale et minimale entre deux sommets ? ... et ...

Arbres

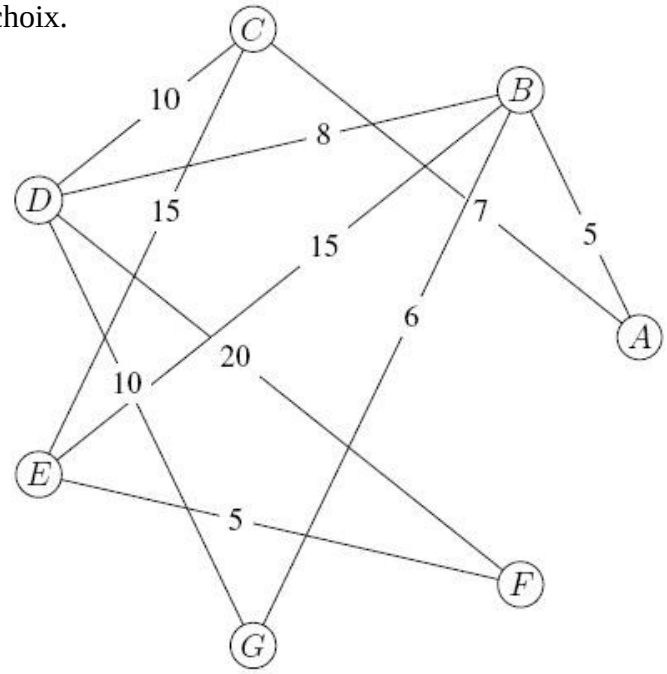
Un arbre est un graphe connecté qui n'a pas de cycle (un cycle est un chemin qui part d'un sommet et termine sur le même sommet sans repasser deux fois par une arête).
Surlignez un cycle dans le premier graphe que vous avez dessiné en haut de la feuille.
Dessinez ci-dessous un arbre quelconque:

Combien a-t-il de sommets ? ... d'arêtes ? ...
Peut-on dessiner un arbre avec le même nombre de sommets mais un nombre différent d'arêtes ?

En déduire une formule entre le nombre d'arêtes m et le nombre de sommets n d'un arbre.

Algorithme de Dijkstra (plus court chemin)

Appliquez l'algorithme de Dijkstra tel qu'on vient de le voir sur le graphe pondéré suivant, à partir du sommet de votre choix.



Bonus : Essayez d'estimer le nombre d'étapes de l'algorithme en fonction du nombre de sommets n et du nombre d'arêtes m du graphe. A votre avis, est-ce que l'algorithme est efficace ?

Avez-vous une idée d'un autre algorithme pour trouver un chemin dans un graphe ?

Plus court chemin en python avec la bibliothèque pyroutelib3

Dans edupython (ou sur un ordinateur du lycée, dans Pyzo qui se situe sur le disque public dans le dossier Winpython), recopiez le code suivant :

```
from pyroutelib3 import Router

router = Router("foot")
depart = router.findNode(43.627, 3.8189)
arrivee = router.findNode(43.6257, 3.8425)
status, route = router.doRoute(depart, arrivee)
if status == 'success':
    etapes = list(map(router.nodeLatLon, route))
print(etapes)
```

Quelques explications :

- Nous commençons par importer la bibliothèque "pyroutelib3" avec la première ligne "from pyroutelib3 import Router"
- La deuxième ligne permet de définir le véhicule qui sera utilisé pour effectuer le trajet. Dans notre cas, nous allons à pieds ("foot"), mais il est possible de choisir d'autres moyens de transport : cycle, car, horse, tram, train
- Les 2 lignes suivantes permettent de définir le point de départ et le point d'arrivée. Nous avons "router.findNode(latitude, longitude)", il suffit de renseigner la latitude et la longitude du lieu.
- La ligne "status, route = router.doRoute(depart, arrivee)" permet d'effectuer le calcul de l'itinéraire.
- La dernière ligne est exécutée uniquement si le calcul est mené à son terme ("if" de la ligne précédente). La variable "etapes" contient la liste des coordonnées des points de cheminement (points qui constituent le chemin entre le point de départ et le point d'arrivée)

Exécutez le code (cela peut prendre quelques minutes). Si vous avez une erreur à la ligne "import", téléchargez le fichier http://nodfs.xyz/pyroutelib3.py et mettez le dans le même dossier que votre fichier python.

Pour voir le trajet sur une carte, on va utiliser à nouveau la bibliothèque folium. Rajoutez à la suite du code :

```
import folium
c = folium.Map(location=[43.6244, 3.8322], zoom_start=14)
folium.PolyLine(etapes).add_to(c)
c.save("carte.html")
```

Exécutez le et ouvrez le fichier carte.html avec firefox pour voir le trajet calculé.

Bonus : Changez le mode de transport et trouvez l'itinéraire entre deux endroits de votre choix (vous pouvez aller sur openstreetmap.org, et clic-droit sur un endroit pour avoir les coordonnées)