

Représentation de graphe non orienté Recherche de connexité / Détection de cycle

A effectuer en binôme.

But du programme

- Lire un graphe non orienté dans un fichier ; le stocker dans une structure de données de votre choix.
- Déterminer si le graphe est connexe ou non.
- Déterminer s'il contient au moins un cycle ou non.

A vous de définir les traces d'exécution de votre programme pour qu'il soit simple de vérifier son bon fonctionnement.

Graphes à prendre en compte

Graphes non orientés.

Graphes valués ou non, cela n'a pas d'importance pour ce TP.

Sommets : numérotés par des nombres entiers positifs, à partir de 0, sans rupture de séquence.

Fichier de données

Basez-vous sur ce que vous avez fait au TP 1.

Langage de programmation

C

C++ sans utilisation des constructeurs de type class, map, list, ...

Rendu du travail

Délai : 7 jours, samedi et dimanche inclus.

Envoi par email à la même adresse que le TP 1.

Graphes de test fournis en séance de TP.

Sujet du mail : EFREI – L3 – TG – TP2 – *nom1 nom2*

Corps du message : vide

Pièces jointes :

Un fichier au format .txt contenant les traces d'exécution de votre programme.

Autant de fichier de code source (.c, .cpp, .h) que nécessaire.

Les fichiers de données utilisés par votre programme pour lire les graphes de test.

Pas d'archive. Pas de fichier binaire. Pas de « fichier projet » géré par votre outil de développement.

Tous les fichiers doivent avoir un nom préfixé par les noms des deux binômes, par exemple « barbot-lepoivre-tp2.cpp », barbot-lepoivre-tp2-graphe1.txt », ...

Tout manquement à ces consignes sera pénalisé. Les pénalités pourront aller jusqu'à '-10' !

Éléments de notation

Seront pris en compte :

- Votre code lui-même : choix des structures de données, algorithmes choisis et mise en œuvre de ces algorithmes.
- Son adaptabilité à d'autres graphes que ceux demandés en traces d'exécution.
- Les commentaires, et plus généralement la clarté de votre code.
- Les traces fournies.

Toute récupération évidente, même partielle, de code fait par un autre groupe se verra sanctionnée par un 0/20 à tous les étudiants concernés.

Recherche de cycle

Vous avez vu en cours une technique utilisable pour les graphes orientés :

Partant du graphe complet, on élimine itérativement les points d'entrée du graphe (ceux du graphe initial et ceux créés en cours de traitement), et bien entendu les arcs correspondants.

Lorsqu'il n'y a plus de point d'entrée dans le graphe restant :

- S'il reste encore au moins un sommet, on en déduit que le graphe de départ contenait un circuit ;
- Si le graphe est vide, on en déduit qu'il n'y avait pas de circuit.

Vous pouvez adapter cette technique en considérant non plus les points d'entrée, mais les sommets ayant zéro ou un seul voisin, c'est-à-dire les sommets qui ne sont l'extrémité que d'au plus une arête.

En plus de tout cela : attention aux boucles, puisqu'une boucle est un cycle !

Recherche de connexité

Il suffit de choisir un sommet de départ quelconque du graphe, puis de déterminer tous les sommets appartenant à au moins une chaîne dont une extrémité est le sommet de départ.

Si tous les sommets du graphe sont ainsi identifiés, alors le graphe est connexe, sinon il ne l'est pas.
