EFREI 2019/2020
Théorie des Graphes
Mini projet de manipulation de graphe

ATTENTION
Votre planning des cours, TD et soutenances en Théorie des Graphes est relativement court. Les soutenances de projet sont prévues vers fin novembre.
En conséquence :
1. L'énoncé du projet est divisé en deux parties distinctes.
2. Il vous est vivement conseillé de démarrer la réalisation de la première partie dès que vous avez acquis suffisamment de connaissances en cours et travaux dirigés. Ce qui vous est demandé en seconde partie (travail sur des graphes d'ordonnancement) se base sur les développements que vous aurez faits en première partie.
3. Le travail sur la partie I uniquement ne vous permettra pas d'obtenir la note maximale de 20/20. La notation sera probablement décomposée de la façon suivante : partie I = 12, partie II = 8.

Travail à faire en équipes

Nombre d'équipes par groupe TD/TP : 12

Le nombre d'étudiants par équipe sera calculé en fonction du nombre d'étudiants dans chaque groupe TD/TP. Votre enseignant vous l'indiquera lors des séances de TD.

Constitution des équipes : à remettre à votre enseignant au plus tard le 15 octobre 2018.

Aucun changement ne sera possible après cette date. A défaut d'une constitution des équipes fournie par les délégués de chaque groupe TD, les enseignants pourront décider eux-mêmes de la constitution des équipes, sans possibilité de modification.

Langage de programmation

Vous pouvez choisir le langage de programmation qui vous convient. Le langage choisi doit cependant être suffisamment maîtrisé par tous les membres d'une même équipe afin que tous puissent participer. Durant la soutenance, votre enseignant pourra poser n'importe quelle question à n'importe quel membre de l'équipe.

Votre programme doit pouvoir être compilé / exécuté par votre enseignant, sur son PC. Il vous indiquera ce dont il dispose. Vous devrez vous y adapter.

Il est plus que probable qu'au sein de chaque équipe, au moins un étudiant pourra disposer d'un environnement similaire à celui de l'enseignant.

Evaluation

Rendu du travail par email : code source + traces d'exécutions (les conditions d'envoi vous seront communiquées plus tard par votre enseignant).

Des graphes de test vous seront fournis plus tard. Vous devrez exécuter votre programme sur l'intégralité de ces graphes et fournir les traces d'exécution correspondantes.

Un graphe non testé, ou pour lequel les traces d'exécution ne seront pas fournies, sera considéré comme un graphe sur lequel votre programme ne fonctionne pas correctement.

Soutenance : présentation + démonstration de votre programme (les conditions précises vous seront indiquées ultérieurement) + questions/réponses.

Partie I : Lecture et affichage d'un graphe. Détection de circuit. Calcul de rang.

Graphes à prendre en compte

Graphes orientés et valués (une valeur numérique pour chaque arc).
Au maximum 1 arc d'un sommet donné vers un autre sommet donné.
Les sommets sont identifiés par leur numéros. Les numéros commencent à 0 et il n'y a pas de rupture de séquence dans la numérotation. Ainsi, un graphe qui contient 15 sommets aura ses sommets numérotés de 0 à 14.

Votre programme doit être capable d'importer un graphe quelconque répondant aux critères ci-dessus. En particulier, votre programme ne doit pas définir en dur le nombre de sommets et le nombre d'arcs. Il doit être capable de s'adapter à ce qu'il lit sur le fichier.

Fonctions à mettre en œuvre

Déroulement du programme

Mettre en place un programme qui exécute les actions suivantes :

1. Lecture d'un graphe donné dans un fichier texte (.txt) et stockage en mémoire
   Voir annexe 1.
2. Affichage du graphe sous forme matricielle (matrice d'adjacence + matrice des valeurs).
   Attention : cet affichage doit se faire à partir du contenu mémoire, et non pas directement en lisant le fichier.
3. Détection de la présence ou non de circuit dans le graphe (utilisation de la méthode de votre choix).
4. Si le graphe ne contient pas de circuit, calcul du rang de chaque sommet (utilisation de la méthode de votre choix).

Lors de son exécution, afin de faciliter le déroulement de votre soutenance, votre programme doit être capable de « boucler » sur une série de graphes que vous aurez préparés. Stopper votre programme après chaque graphe puis le relancer n'est pas une bonne solution.

La structure globale de votre programme est illustrée par le pseudo-code suivant :

```
Début
    Tant que l'utilisateur décide de tester un graphe faire
        Choisir le graphe à traiter
        Lire le graphe sur fichier et le stocker en mémoire
        Afficher les matrices correspondant au graphe
        Détecter s'il y a un circuit ou non
        si il n'y a pas de circuit dans le graphe alors
            calculer les rangs des sommets et les afficher
        finsi
    fait
Fin
```

Les 4 étapes doivent être mises en œuvre individuellement.

Il est bien évident qu'il est possible de mettre en œuvre la détection de circuit dans l'algorithme de calcul de rang. Le but du TP/Projet est de vous faire apprendre le plus d'algorithmes « clés » de la théorie des graphes, et donc nous vous l'interdisons.

Attention ! Dans l'annexe 5, vous trouverez les consignes concernant les noms de fichiers. Ne faite pas l'utilisateur, pour un choix de graphe, tapez le nom en entier ! S'il vous faut que le programme lise le fichier, disons, L3NEW-TG-A5-g3.txt (c.à.d. le graphe n°3), il doit suffire que l'utilisateur tape « 3 ».

Traces d'exécution

Chacune des 4 étapes doit afficher des traces du déroulement de l'algorithme. En voici un exemple :

<table>
<tr><th>Etape</th><th>Exemple de trace (ce ne sont que des exemples : vous pouvez faire ce que vous voulez, pourvu que l'on comprenne vite et sans problème comment votre algorithme fonctionne)</th></tr>
<tr><td>1</td><td>Affichage à chaque ligne du fichier en entrée de ce qui a été lu, par exemple (voir annexe 1 pour la structure du fichier) :
<pre>
* Lecture du graphe sur fichier
4 sommets
5 arcs
3 -> 1 = 25
1 -> 0 = 12
2 -> 0 = -5
0 -> 1 = 0
2 -> 1 = 7
</pre></td></tr>
<tr><td>2</td><td>Toujours en prenant l'exemple de l'annexe 1 :
<pre>
* Représentation du graphe sous forme matricielle
Matrice d'adjacence
     0   1   2   3   0   1   2   3
 0   F   V   F   F   0   0   1   0   0
 1   V   F   F   F   ou   1   1   0   0   0
 2   V   V   F   F   2   1   1   0   0
 3   F   V   F   F   3   0   1   0   0

Matrice des valeurs
    0   1   2   3
0   *   0   *   *
1   12   *   *   *
2   -5   7   *   *
3   *   25   *   *
</pre></td></tr>
<tr><td>3</td><td>Par exemple, avec la méthode de suppression des points d'entrée :
<pre>
* Détection de circuit
* Méthode d'élimination des points d'entrée
Points d'entrée :
2 3
Suppression des points d'entrée
Sommets restant :
0 1
Points d'entrée :
Aucun
Le graphe contient au moins un circuit.
</pre></td></tr>
</table>

4	*Par exemple, en reprenant le graphe de l'annexe 1 mais en enlevant un arc (pour enlever le circuit)*
	4
	4
	3 1 25
	1 0 12
	2 0 -5
	2 1 7
	`* Calcul des rangs`
	`* Méthode d'élimination des points d'entrée`
	`Rang courant = 0`
	`Points d'entrée :`
	`2 3`
	`Rang courant = 1`
	`Points d'entrée :`
	`1`
	`Rang courant = 2`
	`Points d'entrée :`
	`0`
	`Graphe vide`
	`Rangs calculés :`
	`Sommet  0  1  2  3`
	`Rang    2  1  0  0`

Partie II : Ordonnancement (calcul des calendriers)

La partie I traite de graphes quelconques.
La partie II traite de graphes d'ordonnancement (voir propriétés dans point 5).

La réalisation de la partie II s'ajoute à ce qui a été fait en partie I. Le code de la partie I est utilisé par les développements à réaliser en partie II.

Fonctions supplémentaires à mettre en œuvre

5. Vérifier si le graphe est un graphe d'ordonnancement « correct », c'est-à-dire tel qu'il respecte les propriétés vues en cours et TD :
   - un seul point d'entrée,
   - un seul point de sortie,
   - pas de circuit,
   - valeurs identiques pour tous les arcs incidents vers l'extérieur à un sommet,
   - arcs incidents vers l'extérieur au point d'entrée de valeur nulle,
   - pas d'arcs à valeur négative.

6. Si la réponse au point 5 est « oui », calculer le calendrier au plus tôt, le calendrier au plus tard et les marges.

   Pour le calcul du calendrier au plus tard, vous devez considérer que la date au plus tard de fin de projet est égale à sa date au plus tôt.

   La structure globale de votre programme devient donc modifiée par rapport à la partie I et est illustrée par le pseudo-code suivant :

```
Début
      Tant que l'utilisateur décide de tester un graphe faire
            Choisir le graphe à traiter
            Lire le graphe sur fichier et le stocker en mémoire
            Afficher les matrices correspondant au graphe
            Détecter s'il y a un circuit ou non
            si il n'y a pas de circuit dans le graphe alors
                  calculer les rangs des sommets et les afficher
                  si le graphe est un graphe d'ordonnancement alors
                        calculer et afficher les calendriers
                  finsi
            finsi
      fait
Fin
```

   On notera que dans le test « `si le graphe est un graphe d'ordonnancement` », il n'y a pas besoin de tester à nouveau la présence ou non de circuit. Seules les conditions supplémentaires doivent être testées.

Soutenance

Date

Une inscription aux créneaux de soutenance se fera ultérieurement. Chaque équipe doit être présente au moment de sa propre soutenance uniquement.

Préparation

Des graphes de test vous seront fournis ultérieurement, avant la soutenance, sous forme de schémas.
Vous devrez créer les fichiers de données au format que vous aurez choisi.
Tous les graphes doivent être disponibles sur votre ordinateur au moment de la soutenance.

N'attendez pas ces graphes pour tester votre programme ! Il serait alors trop tard.
N'hésitez pas pendant votre développement à utiliser tous les graphes vus en cours et travaux dirigés, et bien d'autres encore. Plus vous ferez de test, plus vous pourrez être certains que votre programme fonctionne correctement.

Déroulement

- Présentation du travail (préparez une petite présentation « powerpoint » dont vous fournirez une copie papier à votre enseignant en début de soutenance)
- Compilation / exécution de votre programme

Questions / réponses tout au long de la soutenance, au bon vouloir de votre enseignant.
Votre enseignant vous dira quels graphes utiliser pendant votre soutenance.

Attention :
Les soutenances ont une durée limitée. Votre enseignant à un planning très serré. Il ne pourra pas continuer les soutenances après le créneau horaire prévu.
Il vous est donc vivement conseillé d'être vraiment prêt à l'heure du début de votre soutenance, ce qui implique :

- attendre à la porte de la salle et entrer dès que c'est à votre tour ;
- avoir préparé votre ordinateur, y avoir inclus tous vos programmes et fichiers contenant les graphes ;
- avoir vérifié le chargement de la batterie de l'ordinateur ;
- l'avoir démarré et mis en mode veille ;
- avoir un ordinateur de secours sur lequel vous avez aussi vérifié le bon fonctionnement de votre programme, au cas où…
-

Tous les ans, il y a plein d'étudiants qui pensent ne jamais avoir de problème. Tous les ans, il y en a qui en ont. Résultat : ils sont stressés et ont moins de temps pour leur soutenance. Dommage…

Annexe 1 – Structure du fichier « graphe »

Important : Cette annexe ne contient qu'un exemple de structure de fichier. Vous avez le droit de décider de votre propre structure aux conditions suivantes :
- la structure du fichier doit être simple, facilement compréhensible ;
- le graphe représenté dans le fichier doit pouvoir être modifié directement dans le fichier, de façon extrêmement simple (par exemple pour ajouter ou supprimer un arc, ou pour changer la valeur d'un arc).

Votre fichier peut, *par exemple*, avoir la structure suivante :

Ligne 1	Nombre de sommets
Ligne 2	Nombre d'arcs
Lignes 3 à 3 + « nombre d'arcs »	Extrémité initiale, suivie de l'extrémité terminale, suivie de la valeur de l'arc

Avec ce modèle de fichier, s'il contient le texte suivant :

```
4
5
3 1 25
1 0 12
2 0 -5
0 1 0
2 1 7
```

cela correspond à un graphe contenant 4 sommets numérotés de 0 à 3 (numérotation contiguë), et 5 arcs (3,1), (1,0), (2,0), (0,1), (2,1) de valeurs respectives 25, 12, -5, 0 et 7.

Annexe 2 : Rendu du travail

Toutes les équipes devront remettre leur travail à leur enseignant à la même date. La date vous sera communiquées ultérieurement.
Le contenu du rendu :

- Tout fichier code que vous avez tapé vous-même (**donc aucun fichier produit par le logiciel durant la compilation ou exécution)**, bien commenté. Tout fichier que vous utilisez doit être nommé de façon à contenir le même préfixe L3NEW-TG-<numéro d'équipe> : par exemple, si vous êtes l'équipe A5 (équipe n°5 du groupe A), si vous avez un fichier « main » et si vous utiliser le langage C++, ce fichier doit avoir le nom L3NEW-TG-A5-main.cpp.
- Tous les fichiers .txt des graphes de test (oui, malgré le fait que ce sont vos enseignants qui vous ont fourni les graphes de test), doivent se trouver dans le même répertoire que le code, et être nommés de la même façon : ainsi, l'équipe A5 nommera le fichier correspondant au graphe 3, L3NEW-TG-A5-g3.txt.
- Un ppt ou pdf de la présentation, aussi nommé selon le même modèle,
- Les traces d'exécution, nommés de la même façon (ainsi, le fichier traces d'exécution de l'équipe A5 concernant le graphe 3 sera nommé L3NEW-TG-A5-traces3.txt).

Communication

Votre enseignant vous donnera l'adresse email que vous pourrez utiliser pour l'envoi des fichiers ainsi que pour toute question.

Tout email doit avoir comme préfixe du sujet la chaîne L3NEW-TG-Projet-<numéro d'équipe>, éventuellement suivie de l'objet de votre email. Par exemple, lors du rendu, l'équipe A5 pourra envoyer un email dont le sujet est « L3NEW-TG-Projet-A5 Rendu final ».

Ces consignes sont très importantes et leur violation même partielle risque d'entrainer des pénalites. Gardez en mémoire que vos enseignants peuvent mettre en place des traitements automatiques lors de la réception de vos emails, et donc que toute défaillance de votre part peut entrainer un mauvais traitement de votre travail (cela peut aller jusqu'à la mise en spam de vos emails…), avec toutes les conséquences que cela peut entraîner.

Langage de programmation

Au choix : C, C++, Python, Java