EFREI 2019/2020
Théorie des Graphes
Mini projet de manipulation de graphe : <u>Chemins les plus courts</u>

ATTENTION
1. L'énoncé du projet est divisé en deux parties distinctes.
2. 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.

Organisation du travail

Travail par 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 par voie d'annonce sur Moodle dans les jours qui suivent.

Constitution des équipes : à remettre à votre enseignant au plus tard le 27 mars.

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

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

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.

Soutenances

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

Au cas où le confinement reste en place à la date de la soutenance, elle se fera sur Teams ou autre outil de téléconférence selon des modalités qu'il nous reste à établir.

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 (au cas où une soutenance physique a lieu) :

- 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…

Graphes de test

Des graphes de test vous seront fournis avant la soutenance sous forme graphique, dans un délai suffisant pour vous permettre de les coder par des fichiers .txt au format que vous aurez choisi. Ces fichiers devront impérativement être préparés à l'avance, et ils feront partie du rendu. Ils 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.

Rendu du travail :

Toutes les équipes devront remettre leur travail à leur enseignant à la même date. Des consignes vous seront communiquées ultérieurement.

Le contenu du rendu :

- Code source : Tout fichier code que vous avez tapé vous-même à l'exclusion de tout autre fichier (**donc aucun fichier produit par le logiciel durant la compilation ou exécution)**, bien commenté.
- 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), dans le même répertoire que le code.
- Un ppt ou pdf de la présentation.
- Les traces d'exécution sous forme d'un fichier .txt. Vous devrez exécuter votre programme sur l'intégralité des graphes de test 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.

Tout fichier que vous utilisez (et transmettrez à votre enseignant) doit être préfixé par votre numéro d'équipe : par exemple, si vous avez un fichier « main » et si vous utiliser le langage C++, et que vous êtes dans l'équipe B2, ce fichier doit avoir le nom B2-main.cpp (ou B2_main.cpp).

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, et cela entrainera une pénalité.

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.

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.

<u>Annexe 1</u> – 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.

<u>Communication</u>

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 TG-PRJ-<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 « TG-PRJ-A5 Rendu final ».

Ces consignes sont très importantes et leur violation même partielle risque d'entrainer des pénalités. 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