POLYTECHNIQUE MONTRÉAL
LE GÉNIE EN PREMIÈRE CLASSE

LOG 2810
STRUCTURES DISCRETES

TP1 : GRAPHES

Session	Hiver 2019
Pondération	10 % de la note finale
Taille des équipes	3 personnes
Date de remise du projet	26 février 2019 (23h55 au plus tard)
Directives particulières	Soumission du livrable par moodle uniquement (https ://moodle.polymtl.ca).
	Toute soumission du livrable en retard est pénalisée à raison de 10% par jour de retard.
	Programmation C++, Python ou Java
Les questions sont les bienvenues et peuvent être envoyées à : Justine Pepin (justine.pepin@polymtl.ca). Paulina Stevia Nouwou Mindom (paulina-stevia.nouwou-mindom@polymtl.ca)	

1 Connaissances requises

- Notions d'algorithmique et de programmation C++, Python ou Java.
- Notions de théorie des graphes.

2 Objectif

L'objectif de ce travail pratique est de vous permettre d'appliquer les notions théoriques sur les graphes que nous avons vues en cours, sur des cas concrets mais hypothétiques, tirés de votre quotidien. A cet effet, il est question dans ce travail de permettre à un agent de jouer au jeu Qui est-ce ? efficacement.

3 Mise en situation

Avec l'utilisation de plus en plus massive des réseaux sociaux, il est pertinent pour tout ingénieur informatique d'avoir une compréhension approfondie de leur fonctionnement. Or, les réseaux sociaux, dans leur fondement, sont tout simplement des graphes dont les sommets représentent des individus et dont les arcs représentent les relations entre ces individus. Savoir manipuler les graphes de connexions serait donc un atout non seulement en entreprise, mais aussi en environnement de recherche, puisque c'est un domaine aux multiples applications. Le jeu de société « Guess who ? », aussi connu sous le nom « Qui est-ce ? » en français, nous servira de moyen d'apprentissage pour la manipulation des graphes et l'application du raisonnement déductif. Il vous est donc demandé d'implémenter un agent pouvant jouer à ce jeu.

4 Description

La version du jeu Qui est-ce ? auquel votre agent devra jouer diffère légèrement de la version originale. Voici les contraintes du jeu :

- Un fichier `individus.txt` rescense tous les individus que votre agent doit prendre en compte, ainsi que leurs caractéristiques respectives. On considère que tous les individus portent un nom unique qui les distingue.
- Un fichier `relations.txt` rescense toutes les relations amicales existantes entre les individus, ainsi que la qualité de chaque relation (sur une échelle de 0 à 100, 0 dénotant l'absence de relation entre deux personnes, 100 dénotant une relation d'inimitié entre deux personnes et 1 dénotant une grande amitié entre deux personnes. Entre 1 et 100, l'amitié décroît de façon constante).
- Le format de ces fichiers est détaillé dans l'annexe.
- Deux objectifs sont à accomplir par l'agent afin que celui-ci soit fonctionnel. L'agent doit, en guise de premier objectif, identifier les deux personnes mystère choisies au préalable par son adversaire. L'adversaire est un humain qui interagit avec l'agent au moyen d'une interface console. L'agent doit ensuite, en guise de second objectif, établir la meilleure chaîne de contacts en respectant quelques contraintes détaillées plus bas.
- Pour identifier les deux individus mystère, l'agent doit poser des questions jusqu'à ce qu'il puisse mentionner avec certitude leurs noms. À chaque question, l'adversaire doit répondre par oui ou non. Comme il y a deux individus mystère, trois réponses sont possibles : oui pour les deux individus, non pour les deux individus et oui pour un seul individu (mais on ne précise pas lequel !).
- La question posée par l'agent doit être structurée autour de l'absence ou de la présence d'une caractéristique parmi les différentes caractéristiques mentionnées dans le fichier `individus.txt`. Par exemple, si la caractéristique « couleur des cheveux » se trouve dans le fichier `individus.txt`, avec les options « blonds », « marrons » ou « noirs », alors l'agent pourrait poser la question : « Les individus mystère ont-ils les cheveux noirs ? »
- À tout moment de cette phase du jeu, l'adversaire peut exiger de demander la liste des suspects restants selon l'agent. Il lui suffit d'entrer la lettre "s" au lieu de répondre à la question de l'agent pour que s'affichent les noms des personnes mystère potentielles. Si l'adversaire entre une réponse qui n'est pas attendue, alors l'agent pose à nouveau sa question jusqu'à ce qu'on lui réponde convenablement.
- Un exemple d'affichage de réponse aux questions de l'agent se trouve en annexe.
- Une fois que l'agent a identifié les deux personnes mystère, il doit demander à son adversaire : « Les individus mystère étaient-ils <Nom1> et <Nom2> ? » Encore trois réponses sont valides ici ; oui pour les deux individus, non pour les deux individus et oui pour un seul individu. Dans le cas où l'agent s'est trompé, il demande à son adversaire de rectifier les 2 noms ou le nom erroné. Si l'adversaire tente d'entrer un nom qui ne se retrouve pas dans le réseau social, l'agent demande à nouveau à son adversaire de rectifier sa réponse jusqu'à ce qu'elle soit acceptable.
- En nommant les noms des personnes mystères, lorsque l'adversaire rectifie la réponse de l'agent, si l'adversaire ment à l'agent, l'agent ne doit pas le signaler à l'adversaire et doit simplement accepter le(s) nom(s) fourni(s) comme la bonne réponse.
- Le nombre de questions posées à l'adversaire avant que l'agent propose une solution doit être conservé en mémoire car il sera affiché avec les résultats.

- L'agent passe ensuite au deuxième objectif ; il doit trouver la meilleure chaîne de contacts entre les deux individus, c'est-à-dire celle qui sera de la meilleure qualité possible, après avoir déterminé le sous-graphe des caractéristiques désirables.
- Pour déterminer le sous-graphe des caractéristiques désirables, l'agent doit demander à son adversaire de sélectionner trois caractéristiques indésirables. Ensuite, il doit annuler toute relation entre deux individus qui partagent l'un de ces traits. Autrement dit, deux individus partageant une caractéristique indésirable verront leur relation passer d'une valeur entre 0 et 100 à 0. Ces deux individus ne sont donc plus connectés socialement dans le sous-graphe des caractéristiques désirables.
- Si un individu mystère a une relation indésirable, on n'enlève pas le lien entre cet individu mystère et son voisin. Autrement dit, les individus mystère gardent tous leurs liens de relation intacts.
- Pour trouver la meilleure chaîne de contacts entre les deux individus, l'agent doit utiliser le sous-graphe des caractéristiques désirables, et à partir du premier individu mystère, doit trouver un chemin jusqu'au second individu mystère en se servant des arcs dont la valeur est non nulle.
- L'agent doit afficher le résultat de son travail selon le format indiqué en annexe lorsque l'utilisateur du programme l'exige. Le résultat comprend, en plus du sous-graphe des caractéristiques désirables : la meilleure chaîne entre les deux individus mystère, le nombre de questions posées par l'agent avant de formuler une tentative pour identifier les deux individus mystère, le nom des individus mystère trouvés, le nom des individus mystère qui n'ont pas été devinés et les trois caractéristiques indésirables.

5 Composants à implémenter

C1. Écrire une fonction « creerReseauSocial() » qui lit les fichiers texte contenant les informations sur les individus et les relations qu'ils entretiennent, et génère le réseau social correspondant. Cette fonction prend en paramètre deux noms de fichiers à lire.

C2. Écrire une fonction « afficherReseauSocial() » qui affiche le réseau social selon le format présenté en annexe.

C3. Écrire une fonction « identifierIndividus() » où l'agent trouve les noms des deux individus mystère.

C4. Écrire une fonction « enleverArcsIndesirables() » où l'agent génère le sous-graphe des caractéristiques désirables. Cette fonction prend en paramètre trois caractéristiques indésirables.

C5. Écrire une fonction « trouverChaineContacts() » où l'agent trouve la meilleure chaîne de contacts entre deux individus à partir du sous-graphe des caractéristiques désirables. Cette fonction prend en paramètre deux noms d'individus.

C6. Écrire une fonction « afficherResultat() » où l'agent présente le résultat de ses accomplissements selon le format en annexe.

C7. Faire une interface console qui affiche le menu suivant :

   (a) Créer le réseau social.
   (b) Afficher le réseau social.
   (c) Jouer à Qui est-ce ?
   (d) Afficher le résultat.
   (e) Quitter.

5.1 Notes

- Le programme doit toujours réafficher le menu, tant que l'option (e), ou « Quitter », n'a pas été choisie.
- L'utilisateur doit entrer un index valide, sinon le programme le signale et affiche le menu de nouveau.
- L'option (a) permet de lire de nouveaux fichiers d'individus et de relations. Les noms des fichiers doivent être demandés à l'utilisateur. Le programme prend les noms des fichiers à lire à partir de la console, car l'utilisateur du programme peut donner les fichiers de son choix, en autant qu'ils respectent les formats en annexe.
- L'option (b) permet d'afficher dans la console le résultat de la lecture des fichiers et de la création du réseau social. Cette option ne peut pas être choisie avant l'option (a).
- L'option (c) permet à l'utilisateur de jouer à Qui est-ce ? en devenant l'adversaire de l'agent. L'agent doit tour à tour identifier les individus mystère et trouver la meilleure chaîne de contacts entre ces individus. Cette option ne peut pas être choisie avant l'option (a).
- L'option (d) permet à l'utilisateur de visualiser les résultats de la dernière partie de Qui est-ce ? jouée contre l'agent. Cette option ne peut pas être choisie avant l'option (c).

Prenez note que selon votre convenance, le mot "fonction" peut être remplacé par "méthode" et vice-versa, et que plusieurs autres fonctions/méthodes peuvent être ajoutées pour vous faciliter la tâche.

6 Livrable

Le livrable attendu est constitué du code source et du rapport de laboratoire. Le livrable est une archive (ZIP ou RAR ou tar.gz) dont le nom est formé des numéros de matricule des membres de l'équipe, séparés par un trait de soulignement (_). L'archive contiendra les fichiers suivants :

- les fichiers .cpp ou .py ou .java ;
- les fichiers .h le cas échéant ;
- le rapport au format PDF ;
- les fichiers .txt passés en argument.

L'archive ne doit pas contenir de programme exécutable, de fichier de projet ou solution de Visual Studio, de répertoire Debug ou Release, etc. Les fichiers .cpp et .h, ou .py, ou .java suffiront pour l'évaluation du travail.

6.1 Rapport

Un rapport de laboratoire rédigé avec soin est requis à la soumission (format .pdf, maximum 8 pages). Sinon, votre travail ne sera pas corrigé. Le rapport doit obligatoirement inclure les éléments ou sections suivantes :

1. Page de présentation : elle doit contenir le libellé du cours, le numéro et l'identification du TP, la date de remise, les matricules et noms des membres de l'équipe.
2. Introduction avec vos propres mots pour mettre en évidence le contexte et les objectifs du TP.
3. Présentation de vos travaux : une explication de votre solution. Ajoutez le diagramme de classes complet, contenant tous les attributs et toutes les méthodes ajoutées.

4. Difficultés rencontrées lors de l'élaboration du TP et les éventuelles solutions apportées.

5. Conclusion : expliquez en quoi ce laboratoire vous a été utile, ce que vous avez appris, etc. Vous pouvez également indiquer le temps passé sur ce TP à des fins d'ajustement pour la prochaine session.

Notez que vous ne devez pas mettre de code source dans le rapport.

6.2 Soumission du livrable

La soumission doit se faire uniquement par Moodle. N'oubliez pas d'inscrire vos noms dans le Wiki de la page du cours pour déclarer les membres de votre équipe afin qu'une boîte de remise commune soit créée et que vous puissiez remettre votre travail.

7 évaluation

éléments évalués	Points
Qualité du rapport : respect des exigences du rapport, qualité de la présentation des solutions	2
Qualité du programme: compilation, structures de données, gestion adéquate des variables et de toute ressource (création, utilisation, libération), passage d'arguments, gestion des erreurs, documentation du code, etc.	3
Composants implémentés : respect des requis, logique de développement, etc.	
C1	4
C2	2
C3	3
C4	4
C5	3
C6	2
C7	2
Total de points	25

8 Documentation

- `https://www.draw.io/` (Diagrammes UML)
- `https://www.lucidchart.com/pages/examples/uml_diagram_tool` (UML)
- `learnxinyminutes.com` (Apprendre rapidement les bases d'un nouveau langage)

Annexe

1 Format des fichiers

1. Le fichier `Individus.txt` comprend les individus et leurs caractéristiques selon le format suivant :
   $individu_1$ $caractéristique_1$ $caractéristique_2$ $caractéristique_3$
   $individu_2$ $caractéristique_1$ $caractéristique_2$ $caractéristique_3$
   ...
   Après le nom de l'individu suit la couleur de ses cheveux, qui est la première caractéristique (noir-N, roux-R, blond-B, marron-M). Ensuite vient la couleur de ses yeux (bleu-B,vert-V, noir-N, gris-G, marron-M) et enfin le département auquel il appartient (génie informatique-GI, génie électrique-GE, génie physique-GP, génie chimique-GC, génie aérospatial-GA, génie mécanique-GM, génie biomédical-GB, génie industriel-GInd, génie énergétique-ER).

2. Le fichier `Relations.txt` comprend les individus et le pourcentage de relation existant entre eux. Lorsque deux individus ne se connaissent pas, le pourcentage est de 0 et ce n'est pas écrit dans le fichier. Pour chaque ligne i du fichier on a :
   $individu_i$ $pourcentage_i$ $voisin_i$
   ...

2 Affichage du graphe

L'affichage du graphe se fait de la façon suivante :
($individu_i$ $voisin_i$ ($pourcentage\ de\ relation\ entre\ eux$))
($individu_i$ $voisin_i$ ($pourcentage\ de\ relation\ entre\ eux$))
...
Le pourcentage de relation doit être non nul.

3 Exemples d'affichage suite à la réponse aux questions

Agent : Est ce que les individus ont les cheveux noirs ?
Adversaire : s
Agent : Les suspects encore sur la liste sont :
Fabien
Rose
Ingrid
...

Agent : Est ce que les individus ont les cheveux noirs ?
Adversaire : Oui pour les deux
Agent : Est-ce que les individus sont en génie chimique ?
Adversaire : Non aucun des deux
Agent : Est-ce que les individus sont en génie mécanique ?

Adversaire : Oui pour un seul
... On continue l'enchaînement question-réponse.
Vous pouvez inventer un code pour répondre aux questions plus rapidement, comme par exemple : **o** oui pour les deux, **n** non pour les deux, **u** oui pour un seul.

4 Affichage du résultat

1. L'affichage du sous-graphe des caractéristiques désirables selon le format de l'affichage du graphe présenté plus haut ;
2. La meilleure chaîne :
   $(individu - mystere_1 \implies individu - quelconque_1 \implies individu - quelconque_2 \implies ... \implies individu - mystere_2)$;
3. Nombre de questions posées par l'agent ;
4. Noms des individus mystère trouvés ;
5. Noms des individus mystère qui n'ont pas été devinés, s'il y a lieu.
6. Les trois caractéristiques indésirables.