Projet Structures de données : avancé

1 Objectif

Le fichier `movies.xml` contient des données à propos d'un certain nombre de films (les données viennent du site `imdb.com`, *Internet Movie Database*). Ce fichier contient des informations sur les années de sorties et les acteurs de ces films.

Ces informations forment un graphe non dirigé. Deux acteurs sont liés par un arc s'ils ont joué ensemble dans le même film.

1.1 Chemin le plus court

Un des objectifs du projet sera de développer un programme capable de calculer le chemin le plus court entre deux acteurs de ce graphe.

Prenons un exemple concret. Le chemin le plus court entre l'acteur américain *Macaulay Culkin* et l'acteur français *Guillaume Canet* est un chemin de longueur 3 . En effet, *Macaulay Culkin* a joué avec *Donald Trump* dans *Home Alone 2*. *Donald Trump* a lui-même joué avec *Leonardo DiCaprio* dans *Celebrity*. Finalement, *Leonardo DiCaprio* a joué avec *Guillaume Canet* dans *The Beach*. Votre programme devra être capable de calculer ces plus courts chemins et de générer un document xml comme ci-dessous. (Pour le moment, ne tenez pas compte de l'attribut cost)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<path cost="309" nbMovies="3">
       <actor>Macaulay Culkin</actor>
       <movie name="Home Alone 2: Lost in New York" year="1992" />
       <actor>Donald Trump</actor>
       <movie name="Celebrity" year="1998" />
       <actor>Leonardo DiCaprio</actor>
       <movie name="Beach, The" year="2000" />
       <actor>Guillaume Canet</actor>
</path>
```

Sachez qu'il peut y avoir plusieurs chemins de longueur minimale ; vous ne devez en montrer qu'un seul.

1.2 Chemin de cout minimum

Dans le chemin proposé plus haut, *Donald Trump* apparait. Pourtant, il n'est pas vraiment un acteur. Dans *Home Alone 2*, il apparait moins de 10 secondes et ne dit qu'une seule phrase : « Au bout du hall et à gauche ». Dans *Celebrity*, il n'apparait également que 10 secondes.

Dans `movies.xml`, certains films sont accompagnés de tous leurs figurants, alors que d'autres n'ont que leurs deux acteurs principaux. Par exemple, 192 acteurs sont mentionnés pour *Celebrity.* Pour éviter d'avoir des figurants dans les chemins les plus courts, nous vous demandons de calculer les chemins en favorisant les films ayant peu d'acteurs enregistrés. Vous allez donc considérer maintenant un graphe pondéré où le poids d'un film est le nombre d'acteurs de ce film. C'est lié à l'attribut cout du document xml ci-dessus qui représente la somme des poids des films composant un chemin. Le

chemin présenté a un cout de 309 car il y a 68 acteurs pour *Home Alone 2*, 192 pour *Celebrity* et 49 pour *The Beach*.

Un deuxième objectif de ce projet sera donc de calculer les chemins de poids minimum entre deux acteurs. Il est possible de trouver un chemin entre *Macaulay Culkin* et *Guillaume Canet* avec un cout de 91. Celui-ci passe par 4 films :

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<path cost="91" nbMovies="4">
        <actor>Macaulay Culkin</actor>
        <movie name="Good Son, The" year="1993" />
        <actor>Elijah Wood</actor>
        <movie name="Eternal Sunshine of the Spotless Mind" year="2004" />
        <actor>Tom Wilkinson(I)</actor>
        <movie name="Ghost and the Darkness, The" year="1996" />
        <actor>Alex Ferns</actor>
        <movie name="Joyeux Noël" year="2005" />
        <actor>Guillaume Canet</actor>
</path>
```

2 Sur Moodle

Pour mener à bien votre projet, nous vous fournissons plusieurs fichiers :

- `movies.xml` contenant les informations sur les films.
- `movies.xsd`, un schéma xml permettant de valider `movies.xml`.
- une classe `Main.java` que vous ne pouvez pas modifier. Le code de cette classe est présenté ci-dessous. La classe à construire `Graph` devra contenir deux méthodes pour calculer un chemin. Ces deux méthodes prennent trois paramètres. Les deux premiers paramètres sont les noms des deux acteurs. Le troisième paramètre est le nom du fichier où il faudra sauvegarder le fichier xml de résultat.

```
import java.io.File;
import javax.xml.parsers.SAXParser;
import javax.xml.parsers.SAXParserFactory;

public class Main {
   public static void main(String[] args) {
      try {
         File inputFile = new File("movies.xml");
         SAXParserFactory factory = SAXParserFactory.newInstance();
         SAXParser saxParser = factory.newSAXParser();
         SAXHandler userhandler = new SAXHandler();
         saxParser.parse(inputFile, userhandler);
         Graph g = userhandler.getGraph();
         g.calculerCheminLePlusCourt("Macaulay Culkin", "Guillaume Canet", "output.xml");
         g.calculerCheminCoutMinimum("Macaulay Culkin", "Guillaume Canet", "output2.xml");
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

3 Tâches à effectuer

Nous vous demandons de rendre la classe `Main` fonctionnelle en réalisant les différentes tâches suivantes :

- Implémenter un parseur SAX permettant la lecture du fichier XML et la construction du graphe en mémoire en utilisant les structures de données adéquates.
- Implémenter les deux algorithmes pour calculer les chemins (le plus court et celui de poids minimum) entre deux acteurs. S'il est impossible d'aller d'un acteur à un autre, votre programme lancera une exception.
- Sauvegarder ces chemins dans des fichiers XML.

De plus, nous vous demandons d'écrire la DTD `movies.dtd` qui permet de valider les mêmes documents que `movies.xsd` (tout en étant plus permissive évidemment).

4 Tâches bonus

Si vous avez fini trop tôt et que vous vous embêtez durant les séances, vous pouvez faire une ou plusieurs tâches bonus. L'objectif est de faire le projet durant les séances, ne prenez pas du temps à la maison pour faire ces tâches bonus ; privilégiez les autres cours comme le projet Projet AE.

Voici les différentes tâches bonus :

1. Implémentez un parseur DOM pour créer le graphe.
2. Ajoutez une deuxième façon de calculer un chemin en minimisant d'abord le nombre de films et ensuite le cout.
3. Donnez la DTD permettant de valider les chemins calculés. Pour ajouter la dtd dans un fichier xml, voici une petite aide : https://stackoverflow.com/questions/13553614/how-to-add-doctype-in-xml-document-using-dom-java

5 Organisation et livrables

Ce projet se déroule du 6 mars 2018 au 22 mars 2018 par groupe de deux étudiants. Vous composerez vous-même les groupes en respectant la règle que tous les étudiants d'un même groupe doivent être dans la même série. Vous nous communiquerez la composition des groupes au début de la semaine du 6 mars. Durant tout le projet, la présence aux cours est obligatoire.

Si une série comporte un nombre impair d'étudiant, nous accepterons un seul groupe de 3 étudiants. Ce groupe de 3 étudiants devra également faire les deux premières tâches bonus.

Le projet est à remettre via « Moodle » pour le vendredi 22 mars 2018 à 23h59. Nous ne demandons pas de rapport. Si certaines choses méritent des explications, vous pouvez les fournir en commentaire dans les différents fichiers.

Le plagiat sera lourdement sanctionné (au minimum 0 à l'UE). Tous les projets seront automatiquement testés par un logiciel de détection de plagiat.