FACULTÉ DES SCIENCES
DÉPARTEMENT D'INFORMATIQUE

Auteur : Yves Roggeman

ÉNONCÉ DE PROJET

ULB
UNIVERSITÉ LIBRE DE BRUXELLES

Date : 17/11/2014
Document : Java_2015-01
N° de version : 4
Destinataires : Inscrits INFO-F-202

Confidentialité :
Public ☐
Interne ☒
Confidentiel ☐
Ne pas diffuser sans autorisation ☐
Autre ☐

Réf. : 2014-013-001

Objet : Langages de progammation 2

Pièces jointes :

Commentaires : Projet de Java

Ceci est le second énoncé de l'épreuve de première session qui se déroule en janvier. Il sert de base à l'épreuve orale ; l'évaluation finale porte sur l'acquisition des concepts démontrée à cette occasion.

Le but de cet exercice de programmation est de démontrer une bonne connaissance et un usage adéquat des constructions syntaxiques du langage de programmation Java et de sa bibliothèque standard (*Java API*), ainsi que des mécanismes standard de la programmation concurrente et des *threads*. L'évaluation portera donc essentiellement sur l'écriture, la codification, la présentation, les constructions et la pertinence des choix effectués dans ce contexte.

A. PROBLÈME

Il est demandé de concevoir, en Java, une classe réalisant un parcours en parallèle d'un graphe orienté, plus précisément de la composante connexe accessible à partir d'un sommet initial donné.

La méthode de parcours consiste à lancer un *thread* pour chaque sommet successeur du sommet courant, *i.e.* celui traité par le *thread* courant. Afin d'éviter les cycles, un sommet déjà rencontré durant le parcours ne doit plus être traité de la sorte, évidemment.

L'ordre d'examen des sommets successeurs (*i.e.* directement accessibles en un pas) est aléatoire et n'est lancé qu'après un délai d'attente également aléatoire.

Pour illustrer ce parcours, chaque sommet sera marqué par son numéro d'ordre de traitement (0 pouvant indiquer un sommet non encore traité). De plus, chaque sommet comptera le nombre de fois qu'il aura été examiné.

De manière à pouvoir vérifier le bon fonctionnement de l'algorithme, il devra être possible d'afficher le contenu des sommets du graphe, mais aussi de le réinitialiser pour pouvoir commencer un nouveau parcours. Plusieurs parcours successifs devront être effectués pour tenter d'illustrer la variabilité de l'ordre de parcours.

Les paramètres de l'algorithme de parcours sont un graphe et un sommet initial. Plusieurs parcours d'un même graphe à partir de différents sommets initiaux seront testés.

B. RÉALISATION

Comme indiqué en introduction, l'objet de l'exercice est l'écriture du parcours et le bon usage des *threads* Java.

Pour les arcs du graphe, vous pouvez choisir une représentation par matrice d'incidence globale ou par liste de successeurs de chaque sommet. D'un ou l'autre cas, vous êtes invités, si nécessaire, à utiliser les interfaces et classes génériques du « *Java Collection Framework* » (`java.util.ArrayList<E>` ou `java.util.AbstractSequentialList<E>`, par exemple).

Les sommets du graphe portent un identificateur propre et fixe (**`int`**, `String`, ou autre, au choix), afin de les reconnaître lors de l'affichage du résultat du parcours. Afin d'inclure les attributs supplémentaires nécessaires pour ce parcours, ceux-ci seront définis dans une classe dérivée de celle des sommets originaux du graphe.

Enfin, l'ordre aléatoire de traitement des successeurs se fera par usage d'un générateur pseudo-aléatoire, instance de la classe « `java.util.Random` ».

C. REMISE

Votre travail doit être réalisé pour le jeudi 18 décembre 2014 à 18 heures au plus tard. Vous remettrez tous vos codes sources empaquetés en un seul fichier compacté (« .zip » ou autre) *via* le site du cours (`INFO-F202`) sur l'Université virtuelle (http://uv.ulb.ac.be/). Ceux-ci devront contenir en commentaire vos matricule, nom, prénom et année d'études.

Le jour de l'examen, vous viendrez avec une version imprimée — un *listing* — de ces divers fichiers. À titre d'illustration, une impression du résultat d'une exécution du programme est également demandée.