ISIA – PPO – T.P. 3c

Résumé

On souhaite modéliser et planifier l'exécution d'un **workflow de tâches**. Un workflow est un ensemble de tâches avec des dépendances (certaines tâches doivent être terminées avant d'autres). Ce type de problème se retrouve dans la gestion de projets, la compilation de programmes, ou encore l'ordonnancement de calculs en parallèle.

L'objectif est double :

1) **Modéliser** un workflow avec une hiérarchie de classes en Java.
2) **Implémenter des algorithmes de graphes** (détection de cycle, tri topologique).
3) **Gérer les erreurs** avec des exceptions personnalisées.

N'hésitez pas à consulter la documentation en ligne sur le site d'oracle : `https://docs.oracle.com/javase/8/docs/api/` (regardez la version correspondant la version de java que vous utilisez).

I Préparation du TP (5 min)

I.1. Créer un dossier spécifique pour les TP de PPO qui vous appellerez `TP_PPO` par exemple.

I.2. Dans ce dossier, créer un dossier `TP3c`. C'est dans ce dossier que vous travaillerez aujourd'hui. Durant les prochaines séances de TP, essayez de garder cette pratique.

II Modélisation objet (50 min)

II.1. Créez une classe abstraite `Tache` avec :

- un identifiant (`String id`),
- une durée en minutes (`int duree`),
- une méthode abstraite `int cout()`.

II.2. Implémentez deux sous-classes :

- `TacheSimple` : une tâche élémentaire avec un coût fixe.
- `TacheComposee` : une tâche composée de plusieurs sous-tâches ; sa durée est la somme des durées de ses sous-tâches.

II.3. Implémentez une classe `Arc` représentant une dépendance entre deux tâches (`from → to`).

II.4. Implémentez les exceptions suivantes :

- `TacheDejaExistanteException` : une tâche avec le même identifiant existe déjà.
- `TacheInconnueException` : un arc fait référence à une tâche inexistante.
- `ArcInvalideException` : tentative d'ajout d'un arc invalide (ex : boucle sur soi-même).
- `CycleDetecteException` : le workflow contient un cycle.

II.5. Implémentez une classe `Workflow` qui contient :

- une collection de tâches (par identifiant),
- une structure de dépendances (successeurs et prédécesseurs).

Méthodes attendues :

```
void ajouterTache(Tache t) throws TacheDejaExistanteException
void ajouterArc(String from, String to)
    throws TacheInconnueException, ArcInvalideException, CycleDetecteException
List<String> ordreTopologique() throws CycleDetecteException
```

où

1) **Détection de cycle** : à l'ajout d'un arc, ou bien lors du calcul d'ordre topologique, détecter s'il existe un cycle.
2) **Ordre topologique** : Un tri topologique est un ordonnancement linéaire des sommets d'un graphe orienté acyclique (DAG) tel que pour tout arc (u, v), le sommet u apparaisse avant le sommet v dans l'ordre.

Temps total du sujet : 0 heures et 55 minutes