Institut Paul Lambin Session de Aout 2020

Examen de Design Patterns

Christophe Damas, José Vander Meulen

Date et heure : lundi 17 aout à 08h30

Durée de l’examen : 2 h ;

Contenu

[1. Questions sur machine 3](#_Toc43372090)

[a) Graph [6 pts] 3](#_Toc43372091)

[b) MediaPlayer [6 pts] 4](#_Toc43372092)

[2. Question sur papier [8 pts] 5](#_Toc43372093)

**Total : /20**

Vous devez avoir dans votre fichier zip, les répertoires suivants :

* **Q1**
* **Q2**
* **Q3**

A la fin de l’examen, vous devez soumettre les deux projets Q1 et Q2 ainsi que le fichier word d’énoncé complété. Si vous n’aimez pas utiliser word, vous pouvez donner les correspondances des patterns dans un fichier txt.

# Questions sur machine

## Graph [6 pts]

Le projet Q1 contient un fichier xml flight.xml contenant des informations à propos de vols aériens. A partir de ces informations, il est possible de créer un graphe. Le projet propose trois implémentations différentes de graphe : une liste d’arc, une liste d’adjacence et une matrice d’adjacence. Dans les 3 classes, nous avons implémenté une méthode bfs() qui affiche l’itinéraire le plus court entre deux aéroports.

Regardez les 3 méthodes bfs (dans ListeDArc, ListeDAdjacence et MatriceDAdjacence). Les trois codes sont très similaires (les différences se situent uniquement dans le début du premier while). Ce n’est pas très propre.

Par refactoring, améliorez les méthodes bfs() des trois implémentations en utilisant un pattern afin de ne plus avoir de redondance de code.

Vous pouvez modifier les classes Graph, ListeDArc, ListeDAdjacence et MatriceDAdjacence mais vous ne pouvez pas modifier le Main. Il faut que le Main fonctionne de la même manière qu’avant.

Faites le lien entre vos classes/méthodes et le diagramme de classe du pattern choisi.

**Nom du Pattern : Template method**

**Participants théoriques Noms utilisés dans le code**

Abstract Class Graph

ConcreteClass ListeDArci,ListeDAdjacence,MatriceDadjacence

**Nom théorique des méthodes Nom utilisé dans le code**

Templace method bfs

PrimitiveOperation getSet()

## MediaPlayer [6 pts]

Le répertoire Q2 contient un programme fonctionnel mais mal écrit manipulant un media player. En effet, les méthodes onPlay(), onLock(), onNext() et onPrevious() de Player sont remplis de if qui va rendre ce programme vite non maintenable. Par refactoring, améliorez ce programme en utilisant un pattern. Lequel ?

Faites le lien entre vos classes/méthodes et le diagramme de classe du pattern choisi.

**Nom du Pattern :**

**Participants théoriques Noms utilisés dans le code**

**Nom théorique des méthodes Nom utilisé dans le code**

# Question sur papier [8 pts]

Le répertoire Q3 contient un projet contenant 5 patterns différents. Parmi ces 5 patterns, il y a le pattern Singleton. Donnez les 4 autres patterns utilisés. Faites les liens entre les classes/méthodes et le diagramme de classe des différents patterns. Si un pattern apparait plusieurs fois, il compte pour un seul pattern.

**Nom du Pattern 1:**

**Participants théoriques Noms utilisés dans le code**

**Nom théorique des méthodes Nom utilisé dans le code**

**Nom du Pattern 2:**

**Participants théoriques Noms utilisés dans le code**

**Nom théorique des méthodes Nom utilisé dans le code**

**Nom du Pattern 3:**

**Participants théoriques Noms utilisés dans le code**

**Nom théorique des méthodes Nom utilisé dans le code**

**Nom du Pattern 4:**

**Participants théoriques Noms utilisés dans le code**

**Nom théorique des méthodes Nom utilisé dans le code**