Université de Caen Normandie
Département d'informatique
L3 informatique, 2023-2024
Unité SINFL5A1, Session 1
Le 5 décembre 2023, 10h30-12h
Documents autorisés

NB : on peut répondre à une question sans avoir répondu aux questions précédentes

Exercices de conception autour de quelques patterns

On dispose de l'interface **Personne** définie ainsi :

```
public interface Personne {
   String getNom();
   int getAge();
   void setAge(int age);
   String marcher();
}
```

Question 1. Implémentation de base de Personne (2 points)

Ecrire une classe nommée **PersonneImpl**, implémentation simple de Personne (comportant classiquement les attributs, constructeur, et implémentations nécessaires). La méthode marcher() renverra simplement « Je marche ».

Question 2. Utilisation des patterns State et template method (7 points)

Ecrire une classe nommée **PersonneEvolutive** (et les classes ou interfaces associées) qui est une sous-classe de PersonneImpl, et qui utilise le pattern State de sorte que :

- Trois états sont prévus, PremierAge (lorsque age<=2), DeuxiemeAge (lorsque 2<age<80) et TroisiemeAge (age>=80).
- Lorsque le setter de l'âge est utilisé, on bascule automatiquement dans l'état correspondant au nouvel âge (si nécessaire)
- La méthode « marcher() » utilise maintenant la stratégie associée à son état, et renvoie, en prenant en compte l'âge, respectivement « J'ai <age> ans et je marche à 4 pattes », « J'ai <age> ans et je marche normalement » ou « J'ai <age> ans et je marche avec un canne ». Par exemple pour new PersonneEvolutive("Dupont", 82), la méthode marcher() renverra « J'ai 82 ans et je marche avec une canne ». Puis si l'on invoque setAge(20), marcher() reverra alors « J'ai 20 ans et je marche normalement ». On essaiera d'éliminer toute redondance de code dans les trois classes d'état (suggestion : template method).

Question 3. Utilisation du pattern decorator (4 points)

Ecrire un decorator de Personne nommé **PersonneLente** qui permet d'ajouter compléter ce que renvoie la méthode marcher() en y ajoutant « en prenant mon temps ». La version décorée de l'instance précédente aurait sa méthode marcher() qui renverrait alors « J'ai 20 ans et je marche normalement en prenant mon temps ».

Question 4. Utilisation du pattern adapter (5 points)

On dispose d'un traitement qui prend en entrée le type Humain, défini ainsi :

```
public interface Humain {
   public String faireQuelqueChose();
   public boolean estMajeur();
}
```

On souhaite lui soumettre des instances de notre type Personne.
Ecrire l'Adapter correspondant. La méthode faireQuelqueChose() renverra ce que renvoie la méthode marcher(), et estMajeur() renverra *true* si l'âge de la personne est supérieur ou égal à 18, et *false* dans le cas contraire.
Ecrire un exemple de ligne de code qui permet ainsi de créer un Humain à partir d'une Personne.

Question 5. Execution (2 points)

Indiquer ce qui s'affiche à l'exécution du code suivant :

```
Personne p = new PersonneEvolutive("Dupont", 30);
Personne pLente = new PersonneLente(p);
System.out.println(pLente.marcher());
pLente.setAge(90);
System.out.println(pLente.marcher());
System.out.println(p.getAge());
System.out.println(p.marcher());
```