Université Paris Diderot
Master 1 II

Langages à Objets Avancés
Année 2018–2019

Examen session 2

Lundi 17 Juin 2019

Tous documents autorisés (sauf copie du voisin, appareillage électronique, etc). Les téléphones portables, comme tout autre moyen de communication vers l'extérieur, doivent être éteints. Le temps à disposition est de deux heures. Motivez bien vos réponses.

1 Exercice [7 points]

1. Écrire la déclaration C++ d'un type permettant de représenter un tableau d'objets de taille fixe :
   — la taille et le type des objets sont fixés à l'instanciation d'un tel tableau ;
   — si `t` est un tel tableau, `t[i]` devra être une expression valide permettant d'accéder en lecture et écriture au i-ème élément du tableau. Si i n'est pas un indice valide, une exception devra être levée ;
   — si `t` et `t2` sont deux tels tableaux, `t==t2` devra être une expression valide permettant de tester si les deux tableaux contiennent exactement les mêmes éléments. Y a t-il une contrainte sur le type des objets contenus ?
   — si `t` est un tableau, `o << t` devra être une expression valide permettant d'afficher sur le flux `o` l'ensemble des éléments du tableau
2. Écrire la définition de chacune des méthodes déclarées à la question précédente.

2 Exercice [6 points]

Étant donné le code suivant :

```
class Jeu {
private:
  list<Joueur *> joueurs;
protected :
  virtual void initialiserLeJeu()=0;
  virtual bool partieTerminee()=0;
  virtual void proclamerLeVainqueur(Joueur *)=0;
public :
   Joueur *joueurSuivant() {
    static list<Joueur *>::iterator i = joueurs.begin();
    if (i==joueurs.end()) i=joueurs.begin();
    return *i++;
  }
  void onJoueUnePartie() {
    Joueur *courant;
    this->nombreDeJoueurs = nombreDeJoueurs;
    initialiserLeJeu ();
    while (!partieTerminee() ) {
      courant = joueurSuivant();
      courant->joue();
    }
    proclamerLeVainqueur(courant) ;
  }
};
```

1. pourquoi les méthodes `onJoueUnePartie` et `joueurSuivant` ne sont-elles pas qualifiées par `virtual` ?
2. pourquoi les méthodes `initialiserLeJeu`, `partieTerminee` et `proclamerLeVainqueur` sont-elles qualifiées par `virtual` ?
3. à quoi sert `=0` dans la déclaration de ces mêmes méthodes (celles de la question précédente) ?

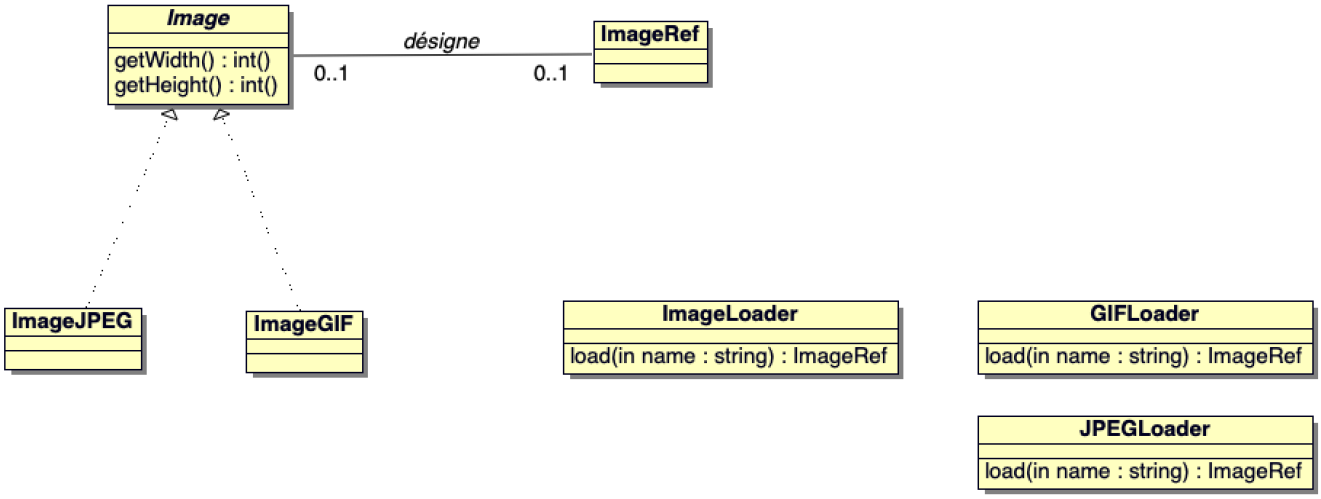
4. que peut-on dire de la classe `Jeu` ?
5. une telle classe correspond au design pattern appelé **template method pattern**. Pourquoi ? Qu'est-ce que cela signifie ?

3 Exercice [7 points]

Étant donné le code suivant :

```
ImageRef myImage = ImageLoader.load("toto.jpeg");
cout << "Taille " << myImage->getWidth() << "x" << myImage->getHeight() << endl;
Image myImage2 = ImageLoader.load("toto.gif");
cout << "Taille " << myImage2->getWidth() << "x" << myImage2->getHeight() << endl;
```

et le diagramme UML suivant :



Écrire les déclarations et définitions des différentes classes et méthodes mentionnées (ainsi que celles manquantes) de sorte que :

1. qu'on ne voit, des `Image`s, rien d'autre que l'*interface*
2. qu'on ne puisse instancier directement des images (quel que soit leur type)
3. qu'on ne puisse instancier les classes `*Loader`, et qu'on ne voit pas les classes `JPEGLoader` et `GIFLoader`
4. les images de type jpeg (resp. gif) soient lues et décodées via `JpegLoader` (resp. `GifLoader`)

Note : On n'écrira pas le code des méthodes effectuant le décodage des images depuis les fichiers...