

# Examen

Mardi 14 Janvier 2014

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 trois heures. Motivez bien vos réponses.

## 1 Exercice [2 points]

Quels sont les affichages produits à l'exécution du code suivant :

```
#include <iostream>
using namespace std;

class T {
private:
    int i;
public:
    T(int)          { i = 2; }
    T(double)       { i = 3; cout << 25 << ' '; }
    ~T()            { cout << -25 << ' '; }
    void operator=(int) { i = 4; }
    int v() const   { return i; }
    virtual int f() { return 11; }
    friend int main();
};

class B : public T {
public:
    B() : T('a') { }
    B(int): T(1.0) { cout << 20 << ' '; }
    ~B()          { cout << -20 << ' '; }
    int f()       { return 33; }
    int g() const { return 44; }
    int g()       { return 444; }
};

ostream &operator<<(ostream &s, const B& b) { return s << 23 << ' ' << b.v(); }
void f(const B &t) { cout << t << endl; }

int main()
{
    T x(1);
    cout << "a: " << x.i << endl;
    T y = 2;
    cout << "b: " << y.i << endl;
    B z;
    T &r = z;
    cout << "c: " << r.f() << endl;
    cout << "d: " << B(1) << endl;
    cout << "e: " << z.g() << endl;
    f(z);
    return 0;
}
```

## 2 Exercice [4 points]

- Écrire la déclaration C++ du type :
  - tableau d'objets** de taille fixe. La taille et le type des objets étant fixés à la création du 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
- Écrire la définition de chacune des méthodes déclarées à la question précédente.

## 3 Exercice [4 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) ;
    }
};
```

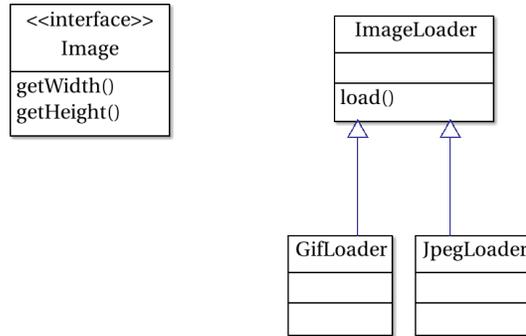
- pourquoi les méthodes `onJoueUnePartie` et `joueurSuivant` ne sont-elles pas qualifiées par `virtual` ?
- pourquoi les méthodes `initialiserLeJeu`, `partieTerminee` et `proclamerLeVainqueur` sont-elles qualifiées par `virtual` ?
- à quoi sert `=0` dans la déclaration de ces mêmes méthodes (celles de la question précédente) ?
- que peut-on dire de la classe `Jeu` ?
- une telle classe correspond au design pattern appelé **template method pattern**. Pourquoi ? Qu'est-ce que cela signifie ?

## 4 Exercice [8 points]

Étant donné le code suivant :

```
Image 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 (incomplet) UML suivant :



Compléter le diagramme et écrire les déclarations et définitions des différentes classes et méthodes mentionnées (ainsi que celles manquantes) de sorte que :

1. on ne puisse créer directement d'instances de `GifLoader` ou `JpegLoader`
2. qu'on ne voit, des `Images`, rien d'autre que l'*interface*
3. qu'on ne puisse instancier la classe `ImageLoader`
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...