LO43 - 2012A - FS01 - 01

# LO43 : Programmation Orientée Objet - Final Automne 2012

Durée : 2h. Documents non autorisés, Traducteurs autorisés, Les réponses de la partie 1 se feront sur le sujet.

NOM : Signature :

PRENOM :

## Questions de cours :

**Question 1 :**

Comment sont passées les références dans les méthodes ? Expliquer pourquoi, dans certains cas, cela peut poser problème.

**Réponse :**

**Question 2 :**

Qu'est ce qu'un Thread ? Comment peut-on en créer un ?

**Réponse :**

# Exercice : Statiquement abstrait

Qu'affiche exactement le programme suivant ?

```
class AbstractStatic {
public static void main(String[] args) {
N un = new N();
N deux = new N(true);
Z trois = new Z();
Z quatre = new Z(false);
Q cinq = new Q();

un.m1();
deux.m1();
deux.m3();
trois.m1();
trois.m3();
quatre.m3();
cinq.m2();

System.out.println(un.n);
System.out.println(un.b);
System.out.println(un.i);
System.out.println(deux.n);
System.out.println(deux.b);
System.out.println(deux.i);
System.out.println(trois.n);
System.out.println(trois.b);
System.out.println(trois.o);
System.out.println(quatre.n);
System.out.println(quatre.b);
System.out.println(quatre.o);
System.out.println(cinq.n);
System.out.println(cinq.d);
} }


abstract class C {
static int n = 0;

C() {
n++; }

C(boolean b) {
if (b) {
n++; } } }


abstract class R extends C {
boolean b = false;

R() {
super(); }

R(boolean b) {
super(b);
n++; }

abstract boolean m1();
}


class Q extends C {
int d = 0;

Q() {
super(true); }

boolean m1() {
if (d == 1) {
return true; }
d++;
return false; }

void m2() {
if (d == 2) {
d++; } }
}


class N extends R {
int i = 1;

N() {
super();
b = false; }

N(boolean b) {
super(b);
n++;
i++; }

boolean m1() {
if (b) {
b = false; } else { b = true; }
return !b; }
```

```
void m3() {
i++ ; }
}

class Z extends R {
int o ;

Z() {
super() ;
o++ ; }

Z(boolean b) {
super(b) ;
o = 4 ; }

boolean m1() {
return !b ; }

void m3() {
o++ ; } }
```

**Réponse :**

---

## Exercice : Un petit tour d'ascenseur

L'objectif est de modéliser le système logiciel d'un ascenseur, dont le fonctionnement est le suivant : L'ascenseur relie trois niveaux : rez-de-chaussée, 1er et 2eme étage. A chaque étage est situé un bouton d'appel de l'ascenseur. Dans la cabine d'ascenseur, trois boutons permettent de sélectionner les trois niveaux différents. Un quatrième bouton permet de réaliser un appel d'urgence et d'arrêter automatiquement la cabine (bouton stop). Lorsqu'un utilisateur appuie sur un bouton d'appel, et si l'ascenseur est libre, alors l'ascenseur se rend à l'étage correspondant à l'appel, ouvre ses portes et attend pendant 10 unités de temps une pression sur un des boutons intérieurs. Si aucune pression n'a lieu, l'ascenseur referme ses portes et attend le prochain appel ou une pression sur un bouton interne. Si une pression a lieu sur un bouton interne, alors l'ascenseur referme ses portes (si nécessaire), met le moteur en marche puis se rend à l'étage indiqué. Lorsque l'étage est atteint, l'ascenseur arrête le moteur, ouvre ses portes pendant 10 unités de temps, referme ses portes et attend une nouvelle requête.

**Question 1 : Proposez un diagramme de classe.**

**Question 2 : Proposez un diagramme d'activités de cet ascenseur.**

**Question 3 : Parmi ces classes, quelles sont celles qui peuvent ou doivent être des Thread ? Y-a-t-il une ressource critique ? Comment peut-on contrôler les accès à celle-ci ?**

**Question 4 : Ecrivez les méthodes des classes impliquées dans l'utilisation de ce(s) Thread(s)**

*Remarque : N'oubliez pas de commenter vos diagrammes. La notation tiendra autant compte de la qualité de vos diagrammes que des commentaires qui les accompagnent.*

---

## Exercice : Maintenant que l'on a un bel ascenseur, il faut le protéger

Le but est de protéger l'accès à l'ascenseur en limitant l'accès à certaines salles du bâtiment.

L'ouverture de chacune des portes de ces salles est commandée par un lecteur de badges placé à proximité. Les badges qui permettent l'ouverture des portes ne sont délivrés qu'aux personnes autorisées. Les droits d'accès sont donnés entre les groupes de personnes et les groupes de portes. (une personne ou une porte doit toujours être au moins dans un groupe). Un groupe de portes peut contenir des portes dispersées dans tout le bâtiment. Une porte donnée ne peut appartenir qu'à un seul groupe de portes. Une même personne peut appartenir à plusieurs groupes. La définition des droits d'accès est effectuée en décrivant pour chaque groupe de personnes les différents groupes de portes qui sont accessibles. Le système de contrôle d'accès doit fonctionner de la manière la plus autonome possible. Un superviseur est responsable de la configuration initiale et de la mise à jour des différentes informations de définition des groupes de personnes et de portes. Un gardien dispose d'un écran de contrôle et est informé des tentatives de passage non autorisées. Les alarmes sont transmises en temps légèrement différé : la mise à jour de l'information sur l'écran de contrôle est effectuée toutes les minutes.

**Question 1 : Donnez le diagramme des cas d'utilisation.**

**Question 2 : Donnez le diagramme de séquence associé au scénario : "identification avec un badge".**

**Question 3 : Donnez le diagramme de classe du système.**