utbm
université de technologie
Belfort-Montbéliard

LO43 : Programmation Orientée Objet - Final Automne 2008

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

NOM :

Signature :

PRENOM :

Partie 1 : Programmation Orientée Objet

1 Questions de cours :

Question 1 :

Qu'est ce que l'ellipse ?

Réponse :

Question 2 :

Qu'est ce que l'autoboxing des types primitifs ?

Réponse :

2 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 :

3 Exercice : Liste (dé)Chaînée

L'exemple suivant correspond à une liste chaînée dont les éléments sont des doubles. On veut rendre cette liste générique.

Question :
Quel outil de JAVA peut-on utiliser depuis la version 1.5 ?

Réponse :

Question :
Comment devait-on faire avant ?

Réponse :

Indiquez, dans la colonne de droite, les modifications à faire pour rendre cette liste générique.

```
class ListeChaineeDouble {

private Noeud debut = null;
private Noeud fin = null;
private Noeud courant = null;


ListeChaineeDouble() { };


ListeChaineeDouble(double valeur) {
courant = new Noeud(valeur);
fin = courant;
debut = courant; }

void ajouterElement(double valeur) {
Noeud nouvelleFin= new Noeud(valeur);

if (debut == null) {
debut = nouvelleFin;
fin = nouvelleFin;
} else { fin.setSuivant(nouvelleFin);
fin = nouvelleFin; }
}

boolean estFin() {
return (courant == null); }

double valeur() {
if (!estFin()) {
return (courant.getValeur());
} else {
return Double.NaN; }
}

void premier() {
courant = debut; }

Noeud suivant() {
if (!estFin()) {
courant = courant.getSuivant();
}

if (estFin()) {
return null; } else {
return courant; }
}
}
```

Réponse :

```
class Noeud {
private double valeur;
private Noeud suivant;

public Noeud (double valeur) {
this.valeur = valeur;
suivant = null; }

public void setSuivant(Noeud noeud){
suivant = noeud; }

public Noeud getSuivant(){
return suivant; }

public void valeur(double uneValeur){
valeur = uneValeur; }

public double getValeur() {
return valeur; }
}
```

Réponse :

Partie 2 : Conception Orientée Objet

Exercice : FPS

Cet exercice vous invite à réfléchir à la conception d'un mini jeu de type Capture the Flags.
Ce type de jeu met en scène deux équipes de joueurs, chacun appartenant à l'un des camps Nord ou Sud. Nous supposerons que les joueurs évoluent dans un environnement 2D constitué d'obstacles (mûrs, rochers, ...) de formes basiques (rectangle, cercle, ellipse...).
Dans chaque camp se trouve à un certain emplacement un drapeau qui doit être récupéré par un des joueurs de l'équipe adverse et amené à proximité de l'emplacement de son propre drapeau. Le joueur ayant réussi à ramener le drapeau adverse dans son camp donne un point à son équipe. La première équipe ayant N points remporte alors la partie. Si un joueur portant le drapeau adverse se fait tuer, le drapeau tombe par terre. Pour que le drapeau à terre réintègre son emplacement d'origine, il est indispensable qu'un joueur de son camp le touche.
Les joueurs peuvent s'affronter à l'aide de diverses armes (pistolet, sabre, grenade, ...). Un joueur peut blesser uniquement un joueur d'une autre équipe à l'aide d'une arme. Les joueurs sont caractérisés par une barre de points de vie, l'arme actuellement à la main et un inventaire des armes portés. Chaque arme est caractérisée par le nombre de points de vie qu'elle ôte à/aux l'adversaire/adversaires visé(s) et au type de dégâts causés : dégâts ciblés pour un sabre ou un pistolet, dégâts collectifs à l'intérieur d'un certain rayon pour une grenade. Un joueur ayant perdu tous ses points de vie meurt. Il réapparaitra après un certain laps de temps T à une position aléatoire dans son camp.

Questions :

1. Proposez un diagramme de classe permettant de prendre en compte tous les éléments énoncés ci-dessus.

2. Proposez un diagramme d'états pour rendre compte de l'activité associée à un drapeau.

Remarques :
N'oubliez pas de commenter vos diagrammes. La notation tiendra autant compte de la qualité de vos diagrammes et du respect du formalisme UML que des commentaires qui les accompagnent.