Université Libre de Bruxelles — Vendredi 13 juin

Faculté des Sciences Appliquées

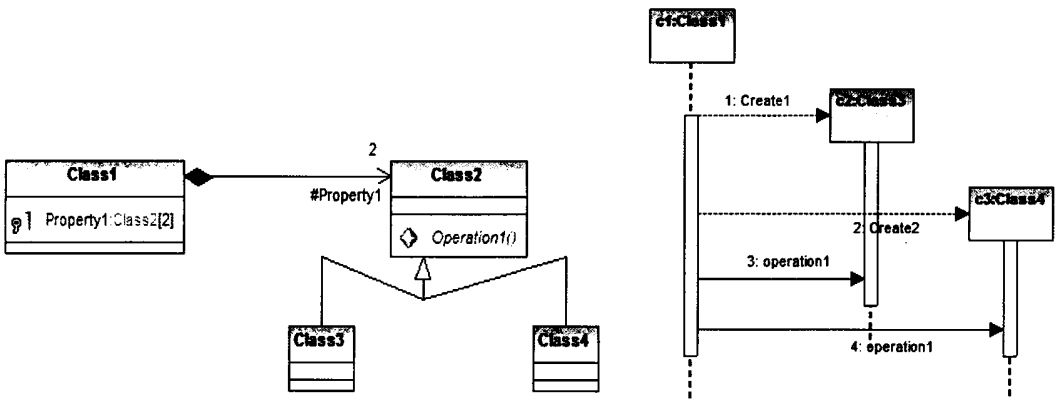
INFO-H-200 : Programmation Orientée Objet

Examen

Remarques préliminaires

- N'oubliez pas d'inscrire nom, prénom et numéro de matricule sur chaque feuille.
- Vous disposez de trois heures et vous ne pouvez pas utiliser de notes.
- Pour les questions où l'on vous demande de justifier ou d'expliquer, la justification ne doit pas faire plus de 4 ou 5 lignes.
- Choisissez bien l'ordre dans lequel vous répondez aux questions : elles n'ont pas toutes le même rapport entre valeur en points et temps nécessaire.
- L'examen est sur 20 points et compte pour 60% de la note finale. Les autres 40% sont donnés par la note du projet.

Question 1 /2



Ecrivez le squelette de code Java qui pourrait être automatiquement généré à partir de ces deux diagrammes UML.

Question 2 /2

Soit deux classes A et B, chacune en possession d'un attribut entier. La deuxième est liée à la première par une relation de composition (A contient un B).

Ensuite deux objets sont créés :

A a1 = new A(5,6)

A a2 = new A(5,6)

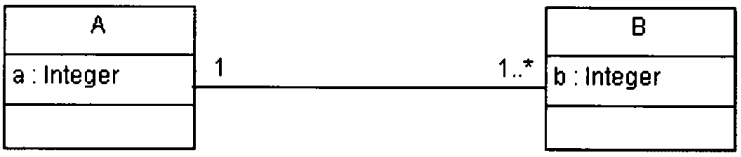
Une instruction apparaît dans le code testant l'égalité de ces deux objets, possédant de fait des attributs de même valeur :

if (a1 == a2) {.....}

L'égalité sera-t-elle vérifiée ou pas ? Pourquoi ?

Si non, comment modifier le test et prévoir le code de ces deux classes de telle manière qu'elle le devienne (au vu de l'égalité des valeurs des deux attributs) ?

Question 3 : /3

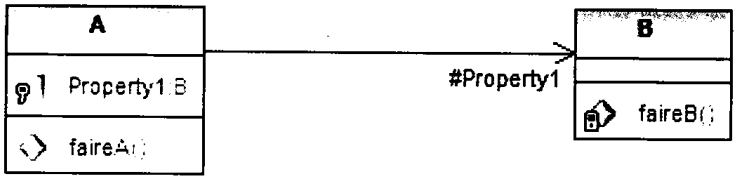


1) Au départ de ce petit diagramme de classe, écrivez les deux constructeurs des classes A et B ainsi que quelques méthodes assez logiques qui pourraient venir compléter la classe A.
2) Faites de la classe A un singleton (classe ne donnant naissance qu'à un objet unique).

Question 4 : /2

Expliquez en quelques lignes les trois manières les plus fréquentes d'assurer la sauvegarde des objets sur le disque dur.

Question 5 : /2



La méthode *faireB()* étant privée dans la classe B, comment la classe A pourrait-elle déclencher cette méthode à partir de sa propre méthode *faireA()* ? Dessinez votre solution sous forme d'un diagramme de séquence correspondant

Que:

Le code Java ci-dessous contient quelques erreurs de logique qui seraient détectées par le compilateur. Veuillez identifier, corriger et expliquer brièvement ces erreurs dans le code directement.

ATTENTION :

ıstification importe autant que l'identification et la correction des erreurs.
es erreurs partout à tort et à travers pourrait être pénalisé !

```
**************************************************************************

abstract class Arbre {
	protected int x, y, hauteur, diametre, branches, terminaisons;
	private String etat, type;
	public Arbre(int x,int y,int h,int d,int b,String e,String t) {
		this.x = x;
		this.y = y;
		this.hauteur = h;
		this.diametre = d;
		this.branches = b;
		this.terminaisons = 0;
		this.etat = etat;
		this.type = type;
	}

	private void afficheToi(){
		System.out.println(type+"\t"+x+"\t"+y+"\t"+hauteur+"\t"+
				diametre+"\t"+branche    "\t"+terminaisons+"\t"+etat);
		this.afficheTerminaisons();
	}
	public abstract void afficheTerminaisons();
	public abstract void meursEnAutomne();
	public void changeEtat(String s){
		etat = s;
	}
	public void pousser(){
		this.branches++;
		this.hauteur+=2;
		this.diametre++;
	}
}

 class Conifere extends Arbre {
	private java.util.ArrayList<Epine> lesEpines;
	public Conifere() {
		lesEpines = new java.util.ArrayList<Epine>();
	}

	public void meursEnAutomne() {
		for(int i=0;i<lesEpines.size();i++)
			lesEpines.get(i).changeDeCouleur("Jaune");
		this.changeEtat(false);
	}
```

```
	public void pousser() {
		super.pousser();
		for(int i=0;i<this.branches;i++){
			lesEpines.add(new Epine(5,"Vert"));
			this.terminaisons++;
		}
	}

	public void afficheTerminaisons(){
		for(int i=0;i<lesEpines.size();i++){
			lesEpines.get(i).afficheToi();
		}
	}
}

 class Feuillus extends Arbre {
	private java.util.ArrayList<Feuille> lesFeuilles;
	public Feuillus() {
		lesFeuilles = new java.util.ArrayList<Feuille>();
	}
	public void meursEnAutomne() {
		lesFeuilles.clear();
		this.changeEtat("Dénudé");
	}
	public void pousser() {
		super.pousser();
		for(int i=0;i<lesFeuilles.size();i++){
			lesFeuilles.get(i).pousser();
		}
		lesFeuilles.add(new Feuille(3,"Vert"));
		this.terminaisons++;
	}
	public void afficheTerminaisons(){
		for(int i=0;i<lesFeuilles.size();i++){
			lesFeuilles.get(i).afficheToi();
		}
	}
	public void changeDeCouleur(String c){
		for(int i=0;i<lesFeuilles.size();i++){
			lesFeuilles.get(i).changeDeCouleur(c);


}

 abstract class Terminaison {
	private String couleur;
	public Terminaison(String c) {
		this.couleur = c;
	}

	public Terminaison(){
		this("Vert");
	}

	public void changeDeCouleur(String c){
		this.couleur = c;
	}
	public abstract void afficheToi();
}
```

```
class Epine extends Terminaison {
	private int longueur;
	public Epine(int l, String c) {
		this.longueur = l;
	}
	public void afficheToi(){
		System.out.println("\tE: "+longueur+"cm - "+this.couleur);
	}
}

class Feuille extends Terminaison {
	private int diametre;
	public Feuille(int d, String c) {
		this.diametre = d;
	}
	public void afficheToi(){
		System.out.println("\tF: "+diametre+"cm");
	}
	public void pousser(){
		this.diametre++;
	}
}

public class Foret {
	private java.util.ArrayList<Arbre> lesArbres;
	public Foret(){
		lesArbres = new java.util.ArrayList<Arbre>();
		lesArbres.add(new Feuillus(15,15,7,3,2,"En pleine forme","Chêne"));
		lesArbres.add(new Conifere(100,20,5,2,1,"En pleine forme","Sapin"));
		lesArbres.add(new Conifere(200,20,10,2,"En pleine forme","Pin"));
		lesArbres.add(new Feuillus(250,15,5,2,1,"En pleine forme","Hêtre"));
		lesArbres.add(new Arbre(10,30,3,2,1,"En pleine forme","Eglantier"));
		lesArbres.add(new Conifere(350,20,3,1,0,"En pleine forme","Pin"));
		lesArbres.add(new Feuillus(40,15,4,2,1,"En pleine forme","Bouleau"));
		lesArbres.get(0).changeDeCouleur("Rouge");
	}

	public void lesArbresPoussent(){
		for(int i=0;i<lesArbres.size();i++)
			lesArbres.get(i).pousser();
	}

	public void lesArbresSAffichent(){
		System.out.println("\nTYPE\tX\tY\tH\tD\tB\tN\tETAT");
		for(int i=0;i<lesArbres.size();i++)
			lesArbres.get(i).afficheToi();
	}


	public void lesArbresMeurentEnAutomne(){
		for(int i=0;i<lesArbres.size();i++)
			lesArbres.get(i).meursEnAutomne();
	}
```

```
        public static void main(String[] args){
            monBeauSapin = new Conifere(20,20,10,2,1,"En pleine forme","Pin")
            Foret f = new Foret();
            for(int i=0;i<5;i++){
                  f.lesArbresPoussent();
            }
            f.lesArbresMeurentEnAutomne();
            lesArbresSAffichent();
        }
}
```

Question 7 /2

Soit en présence d'un mode d'adressage direct, donnez la raison d'être des quatre symboles présents dans cette instruction : *Load D R1 X*

Question 8 /2

Un ordinateur fonctionnant avec mémoire virtuelle possède une mémoire centrale dont la taille permet de contenir 4 pages uniquement. Au départ, cette mémoire est vide. Soit la séquence des pages virtuelles référencées aux instants t=1,2,....., 11 :

P3 / P5 / P6 / P8 / P3 / P9 / P6 / P12 / P3 / P6 / P10

Décrivez et représentez l'évolution des quatre emplacements de la mémoire centrale au fur et à mesure des accès, quand la politique de remplacement des pages est de type : « First In First Out ».