

Évaluation du module Programmation Orientée Objet

Code : INF3034

Date : 2009 - 2010

Prof. : A. Gademer

Nb pages : 9

Documents **non** autorisés.

Calculatrice **non** autorisés.

Durée : **1h30**

Nom :

Classe :

Prénom :

*"Porque eu sou do tamanho do que vejo
E não, do tamanho da minha altura..."
(Car je suis de la taille de ce que je vois,
et non de la taille de ma simple hauteur)
- Alberto Caeiro*

Lisez-moi

- Prenez le temps de lire la question afin de ne pas répondre hors sujet,
- Pas de malus, ce qui ne vous autorise pas à répondre n'importe quoi cependant,
- Des questions peuvent avoir plusieurs bonnes réponses (i.e. vous pouvez cocher plusieurs cases par question), et c'est l'ensemble des bonnes réponses qui valide la question,
- Pour valider une question où une justification est demandée, il est indispensable de donner la bonne justification (i.e. une(des) case(s) cochée(s) sans justification quand celle-ci est demandée est considérée comme une mauvaise réponse),
- Écrire toutes les réponses sur ce document dans l'espace prévu à cet effet donc allez à l'essentiel. Il sera tenu compte de la qualité de la rédaction (style, clarté et concision)
- Quand vous cochez une réponse, faites-le franchement. Les cas où il y aura doute, c'est-à-dire quand on ne sait pas de manière évidente si vous avez choisi telle ou telle réponse, seront systématiquement considérés comme Faux.

1 Approche Objet (7 points)

1.1 À part l'Encapsulation, quels sont les deux autres mécanismes majeurs de l'approche objet ? (1 point)

1. _____

2. _____

1.2 Décrivez ce que l'on nomme la surcharge (1 point) :

1.3 Considérant que la classe Voiture hérite de la classe Vehicule. Les expressions ci-dessous sont-elles valides ? (1,5 points)

1. Vehicule veh = new Voiture();

1. oui non

2. Voiture voit = new Vehicule();

2. oui non

```
3. Vehicule veh = ??;  
   Voiture voit;  
   if(veh instanceof Voiture)  
       voit=(Voiture)veh;
```

3. oui non

Justifiez :

1.4 Soit la classe CompteEnBanque (2.5 points)

```
package finance

public class CompteEnBanque {

    private float solde;

    public void deposer(float montant) {
        if(montant > 0)
            solde+=montant;
    }

    public void retirer(float montant) {
        if(solde > montant)
            solde-=montant;
    }

    protected CompteEnBanque() {
        this.solde = 0;
    }

    public CompteEnBanque(int montant) {
        this();
        deposer(montant);
    }
}
```

Dans la méthode main de la classe Programme, qui n'est pas dans le package finance, ai-je le droit d'écrire ?

- | | | | |
|--|----|------------------------------|------------------------------|
| 1. <code>CompteEnBanque c = new CompteEnBanque();</code> | 1. | <input type="checkbox"/> oui | <input type="checkbox"/> non |
| 2. <code>CompteEnBanque c = new CompteEnBanque(1000);</code> | 2. | <input type="checkbox"/> oui | <input type="checkbox"/> non |
| 3. <code>c.solde = 200;</code> | 3. | <input type="checkbox"/> oui | <input type="checkbox"/> non |
| 4. <code>c.retirer(2000);</code> | 4. | <input type="checkbox"/> oui | <input type="checkbox"/> non |
| 5. <code>c.deposer(100);</code> | 5. | <input type="checkbox"/> oui | <input type="checkbox"/> non |

Justifiez :

1.5 Si A hérite de B (1 point)

- | | | | | | |
|----|---|--------------------------|------|--------------------------|------|
| 1. | B spécialise A | <input type="checkbox"/> | Vrai | <input type="checkbox"/> | Faux |
| 2. | A généralise B | <input type="checkbox"/> | Vrai | <input type="checkbox"/> | Faux |
| 3. | B possède au moins tous les champs et les méthodes de A | <input type="checkbox"/> | Vrai | <input type="checkbox"/> | Faux |
| 4. | A possède au moins tous les champs et les méthodes de B | <input type="checkbox"/> | Vrai | <input type="checkbox"/> | Faux |
| 5. | Toute instance de B peut être considérée comme un A | <input type="checkbox"/> | Vrai | <input type="checkbox"/> | Faux |
| 6. | Toute instance de A peut être considérée comme un B | <input type="checkbox"/> | Vrai | <input type="checkbox"/> | Faux |

2 Java (10,5 points)

2.1 Associez les mots-clef à leurs actions (2,5 points) :

- | | | | | |
|----|-------------------------|--------------------------|--------------------------|--|
| 1. | <code>extends</code> | <input type="checkbox"/> | <input type="checkbox"/> | reporte l'implémentation sur la classe fille |
| 2. | <code>throws</code> | <input type="checkbox"/> | <input type="checkbox"/> | permet l'implémentation d'interfaces |
| 3. | <code>final</code> | <input type="checkbox"/> | <input type="checkbox"/> | indique la présence d'exceptions |
| 4. | <code>implements</code> | <input type="checkbox"/> | <input type="checkbox"/> | empêche la modification |
| 5. | <code>abstract</code> | <input type="checkbox"/> | <input type="checkbox"/> | permet l'héritage de classe |

2.2 Entourez l'orthographe conventionnelle (0.5 points).

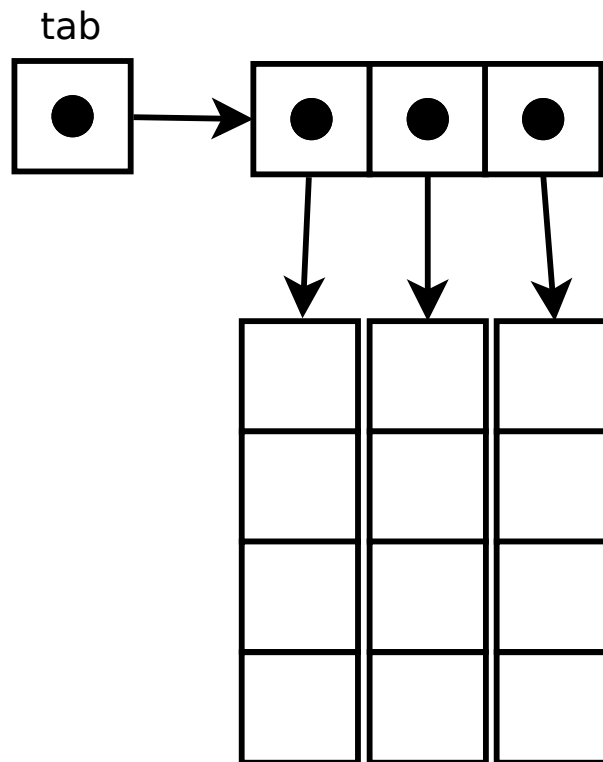
- ```
1. public class zoneDeDessin
2. public class ZoneDeDessin
3. public class zonededessin
```

### 2.3 Quel est le type de la variable `estCarre`? (0,5 point)

```
estCarre = (largeur == hauteur);
```

## 2.4 Remplissez les cases mémoires correspondants au tableau défini dans le programme ci-dessous. (1,5 points)

```
int[][] tab = new int[3][4];
for(int col = 0; col < tab.length; col++)
 for(int lin=0; lin < tab[0].length; lin++)
 tab[col][lin] = (10 * lin + col + 11);
```



## 2.5 Qu'affiche le code suivant ? (1 point)

```
int essais = 3;
String nom = "Jeremy";

StringBuffer strBuf = new StringBuffer("Attention ");
strBuf.append(nom.toUpperCase()).append(" !\n");
strBuf.append("Il ne vous reste");
strBuf.append(10-essais).append("essais.");

System.out.println(strBuf.toString());
```

## 2.6 Que fait le programme suivant ? (2 points)

```
import java.util.Arrays;

public class Programme {
 public static void main(String[] args) {

a. int [] tab = new int [30];

b. for(int i = 0; i < tab.length; i++)
 tab[i] = (int) (100 * Math.random()) ;

c. Arrays.sort(tab);

d. for(int val:tab)
 System.out.print("" + val + " ");
 System.out.println();
 }
}
```

a. \_\_\_\_\_

\_\_\_\_\_

b. \_\_\_\_\_

\_\_\_\_\_

c. \_\_\_\_\_

\_\_\_\_\_

d. \_\_\_\_\_

\_\_\_\_\_

## 2.7 Soient les interfaces (2,5 points)

```
public interface Comparable {
 int compareTo(Object obj);
}

public interface Dessinable {
 void paint(Graphics g);
}

public interface ActionListener {
 void actionPerformed(ActionEvent e);
}
```

## et les classes

```
public abstract class Oeuvre implements Dessinable {
 public abstract String getAuteur();
}
public class Tableau extends Oeuvre implements Comparable {
 public float getPrix() { return prix; }
 [...]
}
```

quelles sont les méthodes accessibles pour ces deux classes ?

|                 | Oeuvre                   | Tableau                  |
|-----------------|--------------------------|--------------------------|
| compareTo       | <input type="checkbox"/> | <input type="checkbox"/> |
| paint           | <input type="checkbox"/> | <input type="checkbox"/> |
| actionPerformed | <input type="checkbox"/> | <input type="checkbox"/> |
| getAuteur       | <input type="checkbox"/> | <input type="checkbox"/> |
| getPrix         | <input type="checkbox"/> | <input type="checkbox"/> |

Justifiez :

---

---

---

---

### 3 TP (2,5 points)

3.1 Dans quel cas utilise-t-on généralement la classe Vector ?  
(1 point)

---

---

---

---

**3.2 Dessinez dans le cadre ci-dessous l'interface obtenue lors de l'exécution du programme suivant. (1,5 points)**

```
public class Main {

 public static void main(String[] args) {

 JFrame fen = new JFrame();
 fen.setPreferredSize(new Dimension(640,480));
 fen.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 fen.setLayout(new BorderLayout());
 JPanel topPanel = new JPanel();
 topPanel.setLayout(new FlowLayout(FlowLayout.LEFT));
 topPanel.add(new JButton("Ok"));
 topPanel.add(new JButton("Cancel"));
 fen.add(topPanel, BorderLayout.NORTH);
 JPanel centerPanel = new JPanel();
 centerPanel.setLayout(new GridLayout(2,2));
 centerPanel.add(new JButton("A"));
 centerPanel.add(new JButton("B"));
 centerPanel.add(new JButton("C"));
 centerPanel.add(new JButton("D"));
 fen.add(centerPanel, BorderLayout.CENTER);
 fen.pack();
 fen.setVisible(true);

 }

}
```

