Université Badji Mokhtar annaba
Faculté de technologie

Département d'informatique
2ème année LMD,S4

Année 2023/2024

Durée : 1h30

12/05/2024

Examen : programmation orientée objet

Documents non autorisés.

Exercice 1:(04 points) :

Pour chacune de ces affirmations, indiquez si elle est vraie ou fausse:

1. La surcharge en java :

a) La surcharge de méthode en Java permet à une classe d'avoir plusieurs méthodes avec le même nom mais des signatures différentes.

b) La surcharge de méthode se base uniquement sur le type de retour des méthodes.

c) Il est possible de surcharger des méthodes statiques en Java.

2. Le mot-clé super :

a) Permet d'appeler le constructeur de la classe mère.

b) Permet d'accéder aux membres de la classe mère.

c) Permet de créer une instance de la classe mère.

3. Le mot-clé final peut être utilisé pour :

a) Empêcher la redéfinition d'une méthode.

b) Empêcher l'héritage d'une classe.

c) Empêcher la modification de la valeur d'un champ.

4. Le constructeur en java :

a) Un constructeur peut être surchargé en définissant plusieurs constructeurs avec différentes signatures.

b) Un constructeur peut appeler un autre constructeur de la même classe en utilisant le mot-clé this().

c) Un constructeur par défaut initialise tous les attributs de la classe avec des valeurs par défaut.

Exercice 2: (04 points) :

- Corrigez les erreurs et donnez le résultat produit par ce programme :

```
package p1;
class Classe1 {
private int n = 1;
private int b = 6;
private String s1;
public int getN() {
 return n; }
public void setN(int i) {
 n = i; }
public int methode1(Classe1 c)
{ c.setN(n + c.getN());
return n + c.getN(); }
public static String methode1(String
s2){
                return s1 + s2;
}
    }
```

```
package p2;

public class Classe2 extends Classe1
{
     private int a;
     private String chaine;

   public int methode1(Classe1 c) {
          c.setN(b + c.getN());
          return b + c.getN();
     }
}
```

```
package p2;
public class Test {
public static void main (String []
args){
Classe1 c1 = new Classe1();
Classe1 c2 = new Classe1();
Classe2 c3 = new Classe2();
System.out.println(c2.getN());
System.out.println(c1.methode1(c2));
System.out.println(c2.getN());
System.out.println(c1==c3);
System.out.println(c3.methode1(c3));
          }
     }
```

Exercice 3 : (12 points)

On veut réaliser une application Java qui gère une équipe de football. L'analyse de l'application mène aux développement des classes suivantes:

I. La classe joueur:

1. Écrire la classe joueur caractérisée par : le nom du joueur, son age, sa position sur le terrain (défonceur, attaquant, ...) , et un statut (en reserve ou principal) en respectant le principe de l'encapsulation.
2. Ajoutez à cette classe un constructeur d'initialisation avec tous les paramètres.
3. Écrire une méthode CalculerPrimeJouésGagnés(joués, gagnés) qui admet en paramètre le nombre des matchs joués et gagnés par son équipe pendant le championnat national, et qui retourne la prime mensuelle du joueur comme suit :

$$\textbf{Prime} = \begin{cases} \textbf{100000 } DA \times \frac{\textbf{matchs gagnés}}{\textbf{matchs joués}} & \textbf{, si le joueur est principal.} \\ \textbf{50\% de la prime du joueur principal, si le joueur est en reserve.} \end{cases}$$

II. La classe équipe :

1. Écrire la classe équipe caractérisée par : le nom de l'équipe, le nom de la ville de l'équipe, le nombre de matchs joués, le nombre de match gagnés, le nombre de joueurs de l'équipes en respectant le principe de l'encapsulation.
2. Ajoutez à cette classe un constructeur avec tous les paramètres sachant que le nombre de joueurs initial est 25.
3. Écrire une méthode qui permet de comparer le nombre de victoires entre deux équipes.

III. La classe match de football:

1. Écrire la classe match de football caractérisée par: les deux équipes équipe A et équipe B, le nombre de but marqués par l'équipe A, et le nombre de buts marqués par l'équipe B en respectant le principe de l'encapsulation des données.
2. Ajoutez à la classe match de football le constructeur paramétré qui permet de créer un objet de la classe match de football.
3. Définir la méthode EstMatchNul() qui permet de tester si le match été nul.

IV. La classe capitaine:

1. Écrire la classe capitaine qui hérite de la classe joueur. Le capitaine est caractérisé par le nombre de matchs qu'il a joué dans sa carrière.
2. Ajoutez a cette classe un constructeur d'initialisation avec tous les paramètres. Le nombre de matchs doit être au moins 100.