

TP3-JAVA-AKOBI-Banconlé

Sommaire:

Question1:.....	1
Question2:.....	3
Question3:.....	4
Question 4:.....	6
Question5:.....	10
Question6:.....	10

Question1:

On veut avoir une méthode main permettant les appels suivants :

```
public static void main(String[] args) { Cahier cahier1 = new
Cahier(20, "Jaune");
Cahier cahier2 = new Cahier(40, "Rouge"); Achat achat = new
Achat(); achat.add(cahier1); achat.add(cahier2);
System.out.println(achat); } Écrire les classes Cahier et Achat ainsi
que leurs méthodes nécessaires (constructeurs, getters, setters,
affiche, toString, ... )
```

```
package TP3;
import java.util.HashSet;
import java.util.ArrayList;

import java.util.HashSet;

class Cahier {
    private int nombreDePages;
    private String couleur;

    public Cahier(int nombreDePages, String couleur) {
        this.nombreDePages = nombreDePages;
        this.couleur = couleur;
    }

    public int getNombreDePages() {
```

TP3-JAVA-AKOBI-Banconlé

```
        return nombreDePages;
    }

    public String getCouleur() {
        return couleur;
    }

    public double calculerPrix() {
        return nombreDePages / 2.0;
    }

    public String toString() {
        return getCouleur() + " " + getNombreDePages() + " x 1";
    }
}

class Achat {
    private HashSet<Cahier> cahiers;

    public Achat() {
        cahiers = new HashSet<>();
    }

    public void add(Cahier cahier) {
        cahiers.add(cahier);
    }

    public double calculerPrixTotal() {
        double prixTotal = 0;
        for (Cahier cahier : cahiers) {
            prixTotal += cahier.calculerPrix();
        }
        return prixTotal;
    }

    public String toString() {
        StringBuilder resultat = new StringBuilder();
        for (Cahier cahier : cahiers) {
            resultat.append(cahier.toString()).append("\n");
        }
        resultat.append("prix: ").append(calculerPrixTotal());
        return resultat.toString();
    }
}

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

TP3-JAVA-AKOBI-Banconlé

```
Cahier cahier1 = new Cahier(20, "Jaune");
Cahier cahier2 = new Cahier(40, "Rouge");

Achat achat = new Achat();
achat.add(cahier1);
achat.add(cahier2);
System.out.println(achat);
}
}
```

Résultat:

```
Jaune 20 x 1
Rouge 40 x 1
Prix: 30.0
```

Question2:

Vérifier que le code suivant fonctionne correctement (le résultat doit être vrai); sinon, corriger le code :

```
public static void main(String[] args) { HashSet set = new HashSet<>();
Cahier c = new Cahier(20, "Jaune");
set.add(c); System.out.println(set.contains(new Cahier(20, "Jaune"))); }
```

Réponse: On va essayer de surcharger Cahier pour permettre la comparaison pour le “set.contains”.

Donc on ajoute la méthode equals à Cahier.

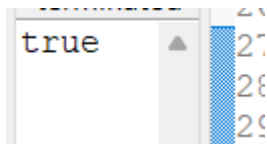
```
public boolean equals(Object objet) {
    if (this == objet) {
        return true;
    }
```

TP3-JAVA-AKOBI-Banconlé

```
}
    if (objet == null || getClass() != objet.getClass()) {
        return false;
    }
    Cahier cahier = (Cahier) objet;
    return nombreDePages == cahier.nombreDePages &&
couleur.equals(cahier.couleur);
}
    public int hashCode() {
        return Objects.hash(nombreDePages, couleur);
    }
}

public class Test {
    public static void main(String[] args) {
        HashSet<Cahier> set = new HashSet<>();
        Cahier c = new Cahier(20, "Jaune") ;
        set.add(c);
        System.out.println(set.contains(c));
    }
}
```

Résultat:

A screenshot of a Java IDE's console window. The word 'true' is displayed in the console, indicating the result of the set.contains(c) operation. The background of the console is light gray, and the text is black.

Question3:

On vend un autre article : le carnet ; un carnet possède juste une valeur de 0 à 9 indiquant un facteur de qualité. Le prix d'un carnet est égal à 3 fois son facteur de qualité. On veut donc que le main suivant fonctionne :

```
public static void main(String[] args) {
    Cahier cahier1 = new Cahier(20, "Jaune"); Cahier cahier2 = new
    Cahier(40, "Rouge"); Carnet carnet = new Carnet (5);
```

TP3-JAVA-AKOBI-Banconlé

```
Achat achat = new Achat();
achat.add(cahier1);
achat.add(cahier2); // un cahier
achat.add(carnet ); // un carnet
System.out.println(achat); }
```

Écrire et modifier les classes et les méthodes nécessaires.

Réponse:

Ici on ajoute la classe Carnet et on modifie donc la classe Achat pour qu'elle prenne en compte la classe Carnet aussi et on calcule le prix des carnets en multipliant leurs note par 3 et on ajoute au prix.

```
class Carnet {
    private int facteurDeQualite;
    public Carnet(int facteurDeQualite) {
        this.facteurDeQualite = facteurDeQualite;
    }
    public int getFacteurDeQualite() {
        return facteurDeQualite;
    }
    public double calculerPrix() {
        return 3 * facteurDeQualite;
    }
    public String toString() {
        return "Carnet (qualité " + facteurDeQualite + ")";
    }
}

class Achat {
    private HashSet<Object> articles;
    public Achat() {
        articles = new HashSet<>();
    }
    public void add(Object article) {
        articles.add(article);
    }
    public double calculerPrixTotal() {
        double prixTotal = 0;
        for (Object article : articles) {
            if (article instanceof Cahier) {
                prixTotal += ((Cahier) article).calculerPrix();
            } else if (article instanceof Carnet) {
                prixTotal += ((Carnet) article).calculerPrix();
            }
        }
    }
}
```

TP3-JAVA-AKOBI-Banconlé

```
        return prixTotal;
    }
    public String toString() {
        StringBuilder resultat = new StringBuilder();
        for (Object article : articles) {
            resultat.append(article.toString()).append("\n");
        }
        resultat.append("Prix total: ").append(calculerPrixTotal());
        return resultat.toString();
    }
}

public class Test {
    public static void main(String[] args) {
        Cahier cahier1 = new Cahier(20, "Jaune");
        Cahier cahier2 = new Cahier(40, "Rouge");
        Carnet carnet = new Carnet (5);
        Achat achat = new Achat();
        achat.add(cahier1);
        achat.add(cahier2); // un cahier
        achat.add(carnet ); // un carnet
        System.out.println(achat);
    }
}
```

Résultat:

```
Jaune 20 x 1
Rouge 40 x 1
Carnet (qualité 5)
Prix total: 45.0
```

Question 4:

On veut permettre d'ajouter plusieurs cahiers ou plusieurs carnets à notre achat d'un coup en indiquant un paramètre supplémentaire à la méthode add, indiquant la quantité des cahiers ou des carnets que l'on veut mettre dans l'achat. L'affichage de l'achat devra indiquer la quantité de chaque cahier et carnet : public static void main(String[]

TP3-JAVA-AKOBI-Banconlé

args) { Cahier cahier1 = new Cahier(20, "Jaune"); Cahier cahier2 = new Cahier(40, "Rouge"); Carnet carnet = new Carnet (5); Achat achat = new Achat(); achat.add(cahier1, 5); // 5 cahiers achat.add(cahier2);
achat.add(carnet , 7); // 7 carnets System.out.println(achat); } Modifiez le code en conséquence, sachant que pour représenter une cahier (ou un carnet) et une quantité, le plus simple est de créer une classe QuantiteArticle qui contient un cahier (ou une carnet) et une quantité (une valeur entière)

Réponse:

Ici on va maintenant créer une classe:QuantiteArticle pour rassembler pour chaque article que ce soit les carnets ou les cahiers, on calcule leurs quantités, et enfin on reporte cela dans Achat qu'on va appeler dans Test.

```
package TP3;
import java.util.HashSet;
import java.util.ArrayList;
import java.util.Objects;
class Cahier {
    private int nombreDePages;
    String couleur;
    public Cahier(int nombreDePages, String couleur) {
        this.nombreDePages = nombreDePages;
        this.couleur = couleur;
    }
    public int getNombreDePages() {
        return nombreDePages;
    }
    public String getCouleur() {
        return couleur;
    }
    public double calculerPrix() {
        return nombreDePages / 2.0;
    }
    public String toString() {
        return getCouleur() + " " + getNombreDePages() + " x ";
    }

    public boolean equals(Object objet) {
        if (this == objet) {
            return true;
        }
        if (objet == null || getClass() != objet.getClass()) {
            return false;
        }
    }
}
```

TP3-JAVA-AKOBI-Banconlé

```
    }
    Cahier cahier = (Cahier) objet;
    return nombreDePages == cahier.nombreDePages &&
couleur.equals(cahier.couleur);
}
public int hashCode() {
    return Objects.hash(nombreDePages, couleur);
}
}

class Carnet {
    private int facteurDeQualite;
    public Carnet(int facteurDeQualite) {
        this.facteurDeQualite = facteurDeQualite;
    }
    public int getFacteurDeQualite() {
        return facteurDeQualite;
    }
    public double calculerPrix() {
        return 3 * facteurDeQualite;
    }
    public String toString() {
        return "Carnet (qualité " + facteurDeQualite + ")";
    }
}

class QuantiteArticle {
    private Object article;
    private int quantite;
    public QuantiteArticle(Object article, int quantite) {
        this.article = article;
        this.quantite = quantite;
    }
    public Object getArticle() {
        return article;
    }
    public int getQuantite() {
        return quantite;
    }
}

class Achat {
    private HashSet<QuantiteArticle> articles;
    public Achat() {
        articles = new HashSet<>();
    }
    public void add(Object article, int quantite) {
        articles.add(new QuantiteArticle(article, quantite));
    }
    public double calculerPrixTotal() {
```


TP3-JAVA-AKOBI-Banconlé

```
double prixTotal = 0;
for (QuantiteArticle quantiteArticle : articles) {
    Object article = quantiteArticle.getArticle();
    int quantite = quantiteArticle.getQuantite();
    if (article instanceof Cahier) {
        prixTotal += ((Cahier) article).calculerPrix() *
quantite;
    } else if (article instanceof Carnet) {
        prixTotal += ((Carnet) article).calculerPrix() *
quantite;
    }
}
return prixTotal;
}

public String toString() {
    StringBuilder resultat = new StringBuilder();
    for (QuantiteArticle quantiteArticle : articles) {
        Object article = quantiteArticle.getArticle();
        int quantite = quantiteArticle.getQuantite();

resultat.append(article.toString()).append(quantite).append("\n");
    }
    resultat.append("Prix total: ").append(calculerPrixTotal());
    return resultat.toString();
}
}

public class Test {
    public static void main(String[] args) {
        Cahier cahier1 = new Cahier(20, "Jaune");
        Cahier cahier2 = new Cahier(40, "Rouge");
        Carnet carnet = new Carnet (5);
        Achat achat = new Achat();
        achat.add(cahier1, 5); // 5 cahiers
        achat.add(cahier2,1);
        achat.add(carnet , 7); // 7 carnets
        System.out.println(achat);
    }
}
```

Résultat:

TP3-JAVA-AKOBI-Banconlé

```
Rouge 40 x 1
Jaune 20 x 5
Carnet (qualité 5) 7
Prix total: 175.0
```

Question5:

Pourquoi la classe QuantiteArticle ne doit pas être déclarée "public" ?

Réponse:

Elle n'a pas besoin d'être publique car elle est utilisée uniquement à l'intérieur de votre package, et d'autres classes en dehors de ce package n'ont pas besoin d'y accéder directement. Une déclaration de classe sans modificateur d'accès est accessible uniquement dans le même package.

Question6:

Modifier le code pour que le main suivant fonctionne :

```
public static void main(String[] args) {
    Cahier cahier1 = new Cahier(20, CahierCouleur.Jaune);
    Cahier cahier2 = new Cahier(40, CahierCouleur.Rouge);
    Carnet carnet = new Carnet (5); Achat achat = new Achat();
    achat.add(cahier1, 5);
    achat.add(cahier2); achat.add(carnet , 7);
    System.out.println(achat); // définir la fonction toString qui retourne le
    montant des achats // dans la classe Achat
```

Réponse: Ici on va déclarer la classe enum et donc modifier la déclaration de la couleur dans Cahier.

TP3-JAVA-AKOBI-Banconlé

```
enum CahierCouleur {
    Jaune, Rouge, Bleu;
}
class Cahier {
    private int nombreDePages;
    CahierCouleur couleur;
    public Cahier(int nombreDePages, CahierCouleur couleur) {
        this.nombreDePages = nombreDePages;
        this.couleur = couleur;
    }
    public int getNombreDePages() {
        return nombreDePages;
    }
    public CahierCouleur getCouleur() {
        return couleur;
    }
    public double calculerPrix() {
        return nombreDePages / 2.0;
    }
    public String toString() {
        return getCouleur() + " " + getNombreDePages() + " x ";
    }

    public boolean equals(Object objet) {
        if (this == objet) {
            return true;
        }
        if (objet == null || getClass() != objet.getClass()) {
            return false;
        }
        Cahier cahier = (Cahier) objet;
        return nombreDePages == cahier.nombreDePages &&
couleur.equals(cahier.couleur);
    }
    public int hashCode() {
        return Objects.hash(nombreDePages, couleur);
    }
}
```

Résultat:

TP3-JAVA-AKOBI-Banconlé

Rouge 40 x 1
Jaune 20 x 5
Carnet (qualité 5) 7
Prix total: 175.0

Ci-joint le code final.