

**DÉPARTEMENT MATHÉMATIQUE
ET INFORMATIQUE**

Filière : Génie du Logiciel et des
Systèmes Informatiques Distribués

Programmation Orientée Objet Java



EXERCICES D'APPLICATION : CLASSES, OBJETS

RÉALISÉ PAR
Achraf TAFFAH

ENCADRANT
M. Abdelmajid BOUSSELHAM

2022/2023

Table des matières

Table des matières	2
Introduction Générale	3
L'énoncé de l'exercice	4
Présentation du programme final	5
Conclusion.....	15

Introduction Générale

La programmation orientée objet (POO) est un paradigme informatique consistant à définir et à faire interagir des objets grâce à différentes technologies, notamment les langages de programmation (Python, Java, C++, Ruby, Visual Basic. NET, Simula...).

L'objectif de cet exercice est d'appliquer quelques notions de bases de la programmation orientée objet avec le langage de programmation.

L'énoncé de l'exercice

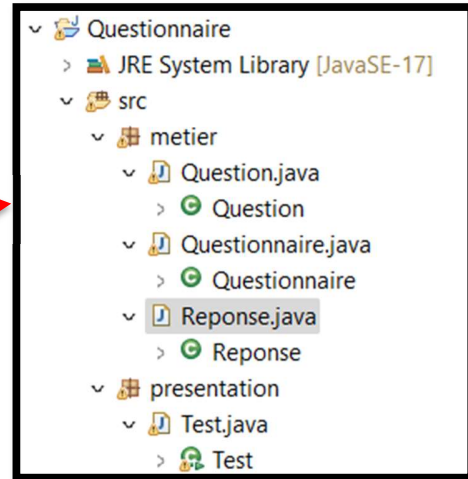
Énoncé :

On souhaite développer une application pour créer un questionnaire de test et d'évaluation. L'application propose un certain nombre de questions avec des options de réponses, L'utilisateur va répondre à chaque question en choisissant une option de réponse. A la fin du Test, l'application doit afficher le score final et la correction de chaque question pour L'utilisateur.


1. Créez la classe Réponse avec deux attributs, titre et correcte (booléen). Ajoutez un Constructeur pour initialiser les paramètres de la réponse et la méthode **toString()** qui Retourne une chaine de caractères contenant les informations de la réponse.
2. Créez la classe Question avec quatre attributs, titre, description, un tableau avec un Maximum de 3 réponses et le score de la question. Ajoutez un constructeur pour Initialiser la question et la méthode **toString()** qui retourne une chaine de caractères Contenant la question avec ses options de réponses.
3. Créez la classe Questionnaire avec trois attributs, titre et un tableau avec un maximum De 5 questions. Ajoutez un constructeur pour initialiser le questionnaire, et la méthode **toString()** qui retourne une chaine de caractères contenant les informations de toutes Les questions avec les options de réponses.
4. Créer une application qui permet de saisir un questionnaire avec l'ensemble des Questions et les options de réponses. Ensuite vous affichez le questionnaire pour L'utilisateur pour répondre aux questions et à la fin affichez le score final avec la Correction de chaque question (l'option correcte de chaque question).

Présentation du programme final

J'ai créé le package **métier** qui contient les classes de bases **Question**, **Réponse** et la classe **questionnaire**, puis j'ai créé le package **présentation** pour tester le fonctionnement de mon programme



J'ai créé la classe réponse avec deux attributs, titre et correcte (booléen). Puis j'ai ajouté un constructeur par défaut et un autre pour initialiser les paramètres de la réponse et j'ai ajouté aussi un constructeur par recopie, enfin j'ai surchargé la méthode **toString()** qui va retourner une chaîne de caractères qui contient les informations de la **réponse**.



```
1 package metier;
2 public class Reponse {
3     private String titre;
4     private boolean correcte;
5
6     public Reponse() {
7         this.titre = "";
8         this.correcte = false;
9     }
10
11     public Reponse(String titre, boolean correcte) {
12         this.titre = titre;
13         this.correcte = correcte;
14     }
15
16     public Reponse(Reponse r) {
17         this.titre = r.titre;
18         this.correcte = r.correcte;
19     }
20
21     public String getTitre() {
22         return titre;
23     }
24     public void setTitre(String titre) {
25         this.titre = titre;
26     }
27     public boolean getCorrecte() {
28         return correcte;
29     }
30     public void setCorrecte(boolean correcte) {
31         this.correcte = correcte;
32     }
33
34     @Override
35     public String toString() {
36         return "Réponse : "+titre;
37     }
38 }
```

J'ai créé la classe **Question** avec quatre attributs, titre, description, un tableau avec un maximum de 3 réponses et le score de la question. Puis j'ai ajouté un constructeur qui va prendre en paramètre juste le nombre des réponses et un autre pour initialiser les paramètres de la réponse et j'ai ajouté aussi un constructeur par copie, ensuite j'ai surchargé la méthode **toString()** qui va retourner une chaîne de caractères qui contient les informations de la réponse. J'ai finalement créé les méthodes **getTrueReponse** et **isTrueReponse** qui m'aideront pour la prochaine utilisation même si ces dernières ne sont pas requises.

```

1 package metier;
2 import java.util.Arrays;
3 public class Question {
4     private String titre;
5     private String description;
6     private float scor;
7
8     private Reponse reponses[] = new Reponse[3];
9     public Question(int nbReponses) {}
23    public Question(String titre, String description, Reponse[] reponses) {}
33
34    public Question(Question q) {}
44
45    public String getTitre() {}
48
49    public void setTitre(String titre) {}
52
53    public String getDescription() {}
56
57    public void setDescription(String description) {}
60
61    public float getScor() {}
64
65    public void setScor(float scor) {}
68
69    public Reponse[] getReponses() {}
72
73    public void setR(Reponse[] reponses) {}
76
78    public String toString() {}
85
86    public String getTrueReponse() {}
96
97    public int isTrueReponse(String reponse) {}
103 }
104

```

C'est le constructeur qui va prendre le nombre des réponses en tant que paramètre.

```

public Question(int nbReponses) {
    if(reponses.length <= 3) {
        this.titre = "";
        this.description = "";
        Reponse[] reponses=new Reponse[nbReponses];
        for(int i = 0; i < nbReponses ; i++){
            reponses[i] = new Reponse();
        }
        this.reponses = reponses;
    }
    else {
        System.out.println("Le tableau des réponses doit contenir un maximum de 3 réponses.");
    }
}

```

Press 'F2' for focus

C'est le constructeur qui va prendre le titre, la description et un tableau des réponses comme des paramètres pour construire un nouvel objet.

```
public Question(String titre, String description, Reponse[] reponses) {  
    if(reponses.length <= 3) {  
        this.titre = titre;  
        this.description = description;  
        this.reponses = reponses;  
    }  
    else {  
        System.out.println("Le tableau des réponses doit contenir un maximum de 3 réponses.");  
    }  
}
```

Press 'F2' for focus

C'est le constructeur qui va prendre un paramètre de type Question pour construire un nouvel objet.

```
public Question(Question q) {  
    if(q.reponses.length <= 3) {  
        this.titre = q.titre;  
        this.description = q.description;  
        this.reponses = q.reponses;  
    }  
    else {  
        System.out.println("Le tableau des réponses doit contenir un maximum de 2 réponses.");  
    }  
}
```

Press 'F2' for focus

Voilà les getters et les setters de tous les attributs privés.

```
45 public String getTitre() {  
46     return titre;  
47 }  
48  
49 public void setTitre(String titre) {  
50     this.titre = titre;  
51 }  
52  
53 public String getDescription() {  
54     return description;  
55 }  
56  
57 public void setDescription(String description) {  
58     this.description = description;  
59 }  
60  
61 public float getScor() {  
62     return scor;  
63 }  
64  
65 public void setScor(float scor) {  
66     this.scor = scor;  
67 }  
68  
69 public Reponse[] getReponses() {  
70     return reponses;  
71 }  
72  
73 public void setR(Reponse[] reponses) {  
74     this.reponses = reponses;  
75 }  
76
```

La méthode **toString()** qui va retourner une chaîne de caractères qui contient les informations de la réponse.

```
@Override  
public String toString() {  
    String str = new String("\t"+titre+"(scor : "+scor+"");  
    for(Reponse rep : reponses) {  
        str+="\n \t\t * "+rep.getTitre();  
    }  
    return str;  
}
```

Press 'F2' for focus

J'ai ajouté cette méthode qui va retourner le titre de la vraie réponse d'une question.

```
public String getTrueReponse() {
    if(reponses.length > 0) {
        for(int i=0;i<reponses.length;i++) {
            if(reponses[i].getCorrecte()) {
                return reponses[i].getTitre();
            }
        }
    }
    return "Le tableau des réponses doit être non vide";
}
```

Press 'F2' for focus

J'ai ajouté cette méthode qui va retourner **True** si et seulement si une réponse donnée en paramètre est vraie.

```
public int isTrueReponse(String reponse) {
    if(this.getTrueReponse().equals(reponse)) {
        return 1;
    }
    return 0;
}
```

Press 'F2' for focus

J'ai créé la classe Questionnaire avec trois attributs, titre et un tableau avec un maximum de 5 questions. Ajoutez un constructeur pour initialiser le questionnaire, et la méthode **toString()** qui retourne une chaîne de caractères contenant les informations de toutes les questions avec les options de réponses. J'ai finalement créé les méthodes **getResult** et **showTrueReponses** qui m'aideront pour la prochaine utilisation même si ces dernières ne sont pas requises.

```
1 package metier;
2 import java.util.Arrays;
3
4 public class Questionnaire {
5     private String titre;
6     private Question questions[] = new Question[5];
7
8     public Questionnaire(int nbQuestions, int nbReponses) {}
9
10    public Questionnaire(String titre, Question[] questions) {}
11
12    public Questionnaire(Questionnaire questionnaire) {}
13
14    public String getTitre() {}
15    public void setTitre(String titre) {}
16    public Question[] getQuestions() {}
17    public void setQuestions(Question[] questions) {}
18
19    public String toString() {}
20
21    public int getResult() {}
22
23    public void showTrueReponses() {}
24 }
```

C'est le constructeur qui va le nombre des questions et le nombre des réponses pour chaque question des paramètres.

```
public Questionnaire(int nbQuestions, int nbReponses) {
    if (this.questions.length <= 5) {
        this.titre = "";
        Question[] questions = new Question[nbQuestions];
        for (int i = 0; i < nbQuestions; i++) {
            questions[i] = new Question(nbReponses);
        }
        this.questions = questions;
    }
    else {
        System.out.println("Le tableau des questions doit contenir un maximum de 5 questions.");
    }
}
```

Press 'F2' for focus

C'est le constructeur qui va prendre le titre et un tableau des questions comme des paramètres pour construire un nouvel objet.

```
public Questionnaire(String titre, Question[] questions) {
    if (questions.length <= 5) {
        this.titre = titre;
        this.questions = questions;
    }
    else {
        System.out.println("Le tableau des questions doit contenir un maximum de 5 questions.");
    }
}
```

Press 'F2' for focus

C'est le constructeur qui va prendre un paramètre de type Questionnaire pour construire un nouvel objet.

```
public Questionnaire(Questionnaire questionnaire) {
    if (questionnaire.questions.length <= 5) {
        this.titre = questionnaire.titre;
        this.questions = questionnaire.questions;
    }
    else {
        System.out.println("Le tableau des questions doit contenir un maximum de 5 questions.");
    }
}
```

Press 'F2' for focus

Voilà les getters et les setters de tous les attributs privés.

```
42 public String getTitre() {
43     return titre;
44 }
45 public void setTitre(String titre) {
46     this.titre = titre;
47 }
48 public Question[] getQuestions() {
49     return questions;
50 }
51 public void setQuestions(Question[] questions) {
52     this.questions = questions;
53 }
54
```

La méthode **toString()** qui va retourner une chaîne de caractères qui contient les informations de la réponse.

```
@Override
public String toString() {
    String str = new String(titre);
    str+="\n";
    int i=0;
    for(Question ques : questions) {
        str+=i+" - Question "+i+"\n";
        str+=ques.toString();
        i++;
    }
    return str;
}
```

Press 'F2' for focus

```
public void showTrueReponses() {
    if(this.questions.length > 0) {
        System.out.println("Correction : ");
        for(int i=0;i<this.questions.length;i++) {
            System.out.println("\t Question "+i+" : "+this.questions[i].getTrueReponse());
        }
    }
    else {
        System.out.println("Le tableau des réponses doit avoir la même longueur de tableau des questions.");
    }
}
```

Press 'F2' for focus

J'ai ajouté cette méthode qui va afficher vraie réponse pour chaque question qui existe dans un objet questionnaire si le tableau des questions est non vide, si non il va afficher un message d'erreur.

J'ai ajouté cette méthode qui va retourner calculer et retourner le résultat final du questionnaire, bien sûr il faut initialiser les questions et les réponses avant d'utiliser cette méthode pour avoir un résultat logique.

```
public int getResult() {
    int result=0;
    for(int i=0;i<this.questions.length;i++) {
        result+=this.questions[i].getScor();
    }
    return result;
}
```

Press 'F2' for focus

```

1 package presentation; import metier.Question; import metier.Questionnaire; import metier.Reponse; import java.util.Scanner;
2 public class Test {
3     public static void main(String args[]) {
4         Scanner sc = new Scanner(System.in);
5         int nbQuestions,nbReponses;
6         System.out.print("Entrer svp le nombre des questions : ");
7         nbQuestions=sc.nextInt();
8
9         System.out.print("Entrer svp le nombre des réponses pour chaque question : ");
10        nbReponses=sc.nextInt();
11
12        Questionnaire questionnaire = new Questionnaire(nbQuestions,nbReponses);
13
14        for(int q=0;q<nbQuestions;q++) {
15            System.out.print("Entrer svp le titre de la question : ");
16            questionnaire.getQuestions()[q].setTitre(sc.nextLine());
17            questionnaire.getQuestions()[q].setTitre(sc.nextLine());
18            System.out.print("Entrer svp la description de la question : ");
19            questionnaire.getQuestions()[q].setDescription(sc.nextLine());
20            questionnaire.getQuestions()[q].setDescription(sc.nextLine());
21            for(int r=0;r<nbReponses;r++){
22                System.out.print("Entrer svp une réponse qui concerne cette question : ");
23                questionnaire.getQuestions()[q].getReponses()[r].setTitre(sc.nextLine());
24                questionnaire.getQuestions()[q].getReponses()[r].setTitre(sc.nextLine());
25
26                System.out.println("Cette réponse est correcte ? : (true ou bien false)");
27                questionnaire.getQuestions()[q].getReponses()[r].setCorrecte(sc.nextBoolean());
28            }
29        }
30        for(int i=0;i<questionnaire.getQuestions().length;i++) {
31            System.out.println("\t\t "+questionnaire.getQuestions()[i].getTitre()+" : ");
32            questionnaire.getQuestions()[i].setScor(questionnaire.getQuestions()[i].isTrueReponse(sc.nextLine()));
33            questionnaire.getQuestions()[i].setScor(questionnaire.getQuestions()[i].isTrueReponse(sc.nextLine()));
34        }
35        System.out.println("Résultat : " +questionnaire.getResult()+"/"+questionnaire.getQuestions().length);
36        questionnaire.showTrueReponses();
37    }
38 }

```

Maintenant j'ai testé le fonctionnement de tous les classes et les méthodes créé.


```

Entrer svp le titre du questionnaire :
QCM Java - Programmation Orientée Objet
Entrer svp le nombre des questions :
2
Entrer svp le nombre des réponses pour chaque question : 3
Entrer svp le titre de la question : Quand la surcharge de méthode est-elle déterminée?
Entrer svp la description de la question : Quand la surcharge de méthode est-elle déterminée?
Quand la surcharge de méthode est-elle déterminée?
Entrer svp une réponse qui concerne cette question : Au moment de l'exécution
Au moment de l'exécution
Cette réponse est correcte ? : (true ou bien false)
false
Entrer svp une réponse qui concerne cette question : Au moment de la compilation
Cette réponse est correcte ? : (true ou bien false)
true
Entrer svp une réponse qui concerne cette question : Au moment du codage
Cette réponse est correcte ? : (true ou bien false)
false
Entrer svp le titre de la question : Quand la surcharge ne se produit pas?
Entrer svp la description de la question : Quand la surcharge ne se produit pas?
Quand la surcharge ne se produit pas?
Entrer svp une réponse qui concerne cette question : Quand il y a plusieurs méthodes avec le même nom mais avec une signature de méthode différente et un nombre ou un typ
Quand il y a plusieurs méthodes avec le même nom mais avec une signature de méthode différente et un nombre ou un type de paramètres différent
Cette réponse est correcte ? : (true ou bien false)
false
Entrer svp une réponse qui concerne cette question : Quand il y a plusieurs méthodes avec le même nom, le même nombre de paramètres et le type mais une signature différen
Cette réponse est correcte ? : (true ou bien false)
true
Entrer svp une réponse qui concerne cette question : Quand il y a plusieurs méthodes avec le même nom, la même signature, le même nombre de paramètres mais un type différen
Cette réponse est correcte ? : (true ou bien false)
false
        Quand la surcharge de méthode est-elle déterminée? :
Au moment de la compilation
        Quand la surcharge ne se produit pas? :
Quand il y a plusieurs méthodes avec le même nom, le même nombre de paramètres et le type mais une signature différente
Quand il y a plusieurs méthodes avec le même nom, le même nombre de paramètres et le type mais une signature différente

```

Dans le package de présentation et plus précisément dans la classe **Test**, j'ai testé mon programme avec 2 questions en Java et 3 réponses pour chaque question.

```

Résultat : 2/2
Correction :
    Question 0 : Au moment de la compilation
    Question 1 : Quand il y a plusieurs méthodes avec le même nom, le même nombre de paramètres et le type mais une signature différente

```

Voilà l'affichage du résultat final et aussi la correction du questionnaire.

Conclusion

Cet exercice m'a aidé d'appliquer plusieurs concepts fondamentales de la programmation orientée objet avec le langage orienté objet java, vraiment c'est un bon exercice qui est besoin plus de concentrations pour le réaliser, je pense que cet exercice est une initiation pour l'utilisation de ArrayList qui vont nous a aidés pour résoudre plusieurs obstacles qui existes dans les tableaux, j'ai ajouté quelques méthodes qui ne sont pas demandé dans l'énoncé juste pour rendre mon code rapide en terme de complexité et aussi plus lisible, comme vous connaissez que c'est pas seulement la complexité en terme de la mémoire et en terme du temps qui sont important mais aussi la complexité cognitive est très importante.