Commencé le	mercredi 25 septembre 2024, 22:28
État	Terminé
Terminé le	mercredi 25 septembre 2024, 23:13
Temps mis	45 min
Points	4,00/4,00
Note	**10,00** sur 10,00 (**100**%)

Question 1

Correct

Note de 1,00 sur 1,00

:Question 6::

Le code Java suivant provoque une `ClassCastException` lors de son exécution :

```public class Voiture {
    private String marque;
    private int annee;

    public Voiture(String marque, int annee) {
        this.marque = marque;
        this.annee = annee;
    }

    // Getters et setters
}

List<Voiture> voitures = Arrays.asList(
    new Voiture("Toyota", 2010),
    new Voiture("Honda", 2012),
    new Voiture("Ford", 2008)
);

Collections.sort(voitures);
```

Pourquoi ce code provoque-t-il une `ClassCastException` ?

a. Parce que les objets `Voiture` ne peuvent pas être triés car ils ne sont pas des types primitifs.

b. Parce que la méthode `compareTo` de la classe `Voiture` lance une exception non gérée.

c. Parce que la liste `voitures` contient des éléments `null`, ce qui provoque l'exception lors du tri.

d. Parce que la classe `Voiture` n'implémente pas l'interface `Comparable`, et aucun `Comparator n'est fourni pour définir l'ordre de tri` ✔

Votre réponse est correcte.

La réponse correcte est :
Parce que la classe `Voiture` n'implémente pas l'interface `Comparable`, et aucun `Comparator n'est fourni pour définir l'ordre de tri`

Question 2

Correct

Note de 1,00 sur 1,00

```
List<Integer> nombres = Arrays.asList(5, 3, 9, 1);
Collections.sort(nombres, new Comparator<Integer>() {
    @Override
    public int compare(Integer i1, Integer i2) {
        return i1.compareTo(i2);
    }
});
System.out.println(nombres);
```

- a. Le Comparator est inutile ici car Integer implémente déjà Comparable. ✔
- b. Pour trier les nombres en ordre croissant.
- c. Le code ne compile pas.
- d. Pour trier les nombres en ordre décroissant.

La réponse correcte est : Le Comparator est inutile ici car Integer implémente déjà Comparable.

Question 3

Correct

Note de 1,00 sur 1,00

Quel est la principale différence entre `Comparable` et `Comparator` ?

- a. `Comparator` permet de trier des chaînes de caractères seulement.
- b. `Comparable` nécessite deux arguments, `Comparator` un seul.
- c. `Comparable` compare selon l'ordre naturel, `Comparator` permet de définir des ordres personnalisés. ✔
- d. Il n'y a pas de différence significative.

La réponse correcte est : `Comparable` compare selon l'ordre naturel, `Comparator` permet de définir des ordres personnalisés.

Question 4

Correct

Note de 1,00 sur 1,00

Considérez le code Java suivant

```
List<Integer> nombres = Arrays.asList(5, 3, 9, 1);
Collections.sort(nombres, new Comparator<Integer>() {
    @Override
    public int compare(Integer i1, Integer i2) {
        return i1.compareTo(i2);
    }
});
System.out.println(nombres);
```

Quel est l'intérêt d'utiliser un Comparator ici ?

- a. Pour trier les nombres en ordre croissant.
- b. Pour trier les nombres en ordre décroissant.
- c. Le Comparator est inutile ici car Integer implémente déjà Comparable. ✔
- d. Le code ne compile pas.

La réponse correcte est : Le Comparator est inutile ici car Integer implémente déjà Comparable.