Fonctions receivers

Exercice 1 : Définition d'une fonction receiver

Définissez une structure `Car` avec des champs pour `brand` (marque) et `year` (année). Créez une fonction receiver `Age` qui calcule l'âge de la voiture.

Code de départ :

```
package main

import (
    "fmt"
    "time"
)

// Définissez la structure Car ici.

// Définissez la fonction receiver Age ici.

func main() {
    // Créez une instance de Car et appelez la méthode Age.
}
```

Astuces

- Pour obtenir l'année actuelle, utilisez `time.Now().Year()`.

Résultat attendu

Lorsque vous instanciez votre structure avec une année spécifique et appelez la méthode `Age`, vous devez obtenir l'âge de la voiture en années.

Exercice 2 : Modification d'une structure via une fonction receiver

Ajoutez une fonction receiver `UpdateYear` à la structure `Car` qui permet de modifier l'année de la voiture.

Code de départ :

```
// Utilisez le code de départ de l'Exercice 1 en y ajoutant la nouvelle fonction receiver.
```

Résultat attendu

Après avoir appelé `UpdateYear` sur une instance de `Car`, l'année doit être mise à jour selon la valeur passée à la méthode.

Exercice 3 : Utilisation des pointeurs avec les fonctions receiver

Modifiez les fonctions receiver `Age` et `UpdateYear` pour utiliser des pointeurs. Expliquez en commentaires l'impact de cette modification.

Code de départ :

```
// Utilisez le code de départ de l'Exercice 2 en modifiant les fonctions receiver pour qu'elles utilisent des pointeurs.
```

Résultat attendu

Les méthodes doivent maintenant refléter les changements de manière persistante sur les instances de `Car`.

Réflexion

En quelques lignes, décrivez l'utilité des pointeurs avec les fonctions receiver et comment cela impacte la gestion de la mémoire.

Exercice 4 : Comparaison de structures avec une fonction receiver

Implémentez une fonction receiver `Equals` qui prendra en paramètre une autre instance de `Car` et retournera `true` si les deux voitures sont de la même marque et du même modèle.

Code de départ :

```
// Reutilisez la structure Car de l'Exercice 1.

// Définissez la fonction receiver Equals ici.

func main() {
// Créez deux instances de Car et comparez-les avec la méthode Equals.
}
```

Résultat attendu

Quand vous comparez deux instances de `Car` avec la même marque et le même modèle, la méthode `Equals` doit retourner `true`. Sinon, elle doit retourner `false`.

Exercice 5 : Chaînage de méthodes avec les fonctions receiver

Définissez une chaîne de méthodes pour la structure `Car` qui permet de modifier plusieurs champs en une seule ligne d'instruction.

Code de départ :

```
// Utilisez la structure Car et modifiez les fonctions receivers pour permettre le chaînage.

func main() {
// Créez une instance de Car et utilisez le chaînage de méthodes pour modifier plusieurs champs.
}
```

Résultat attendu

Vous devez être capable d'effectuer plusieurs modifications sur une instance de `Car` dans une instruction fluide, comme par exemple `car.UpdateYear(2021).SetColor("red").UpgradeEngine()`.