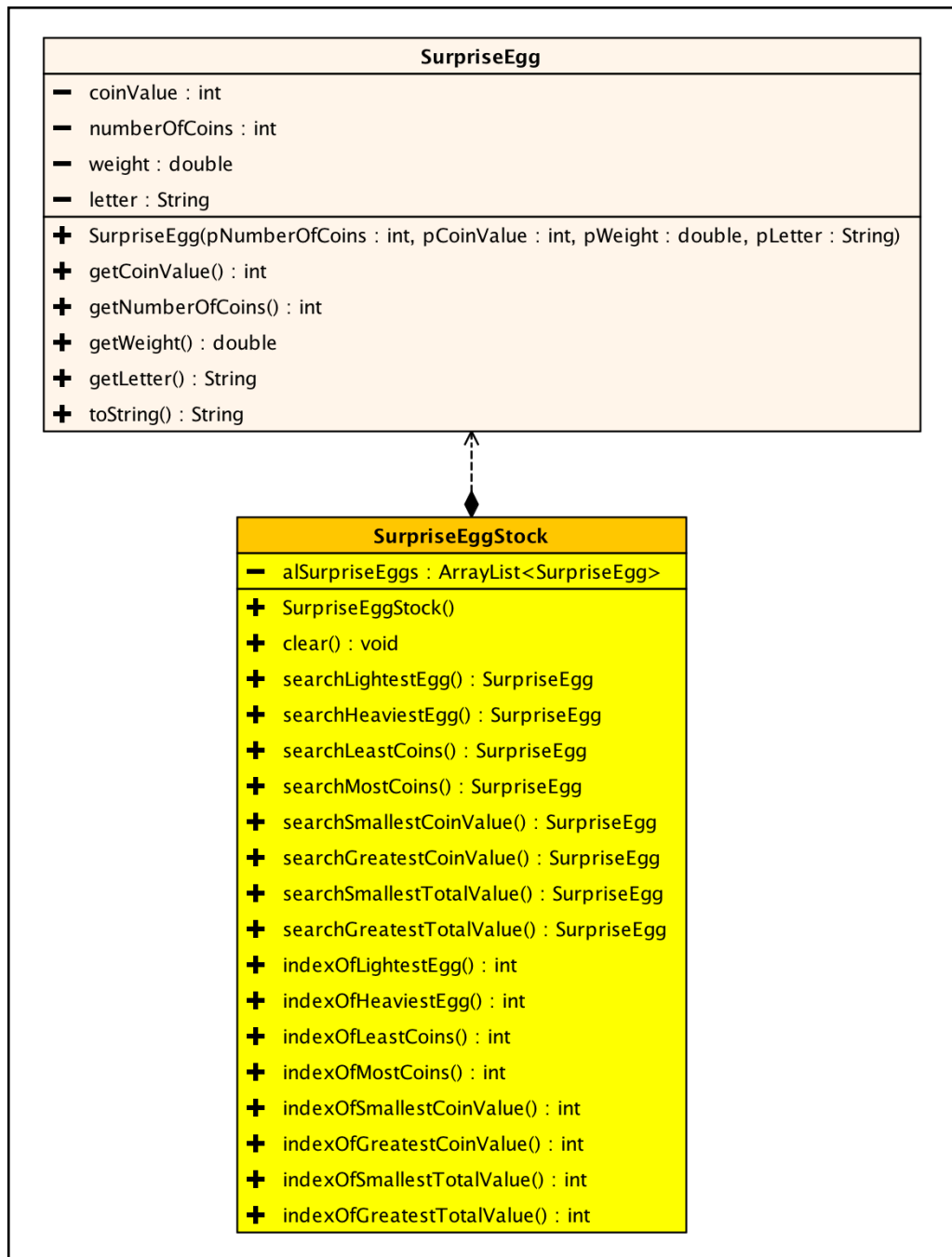


Exercice N1 - Recherche du minimum

Le but de cet exercice est de compléter une classe `SurpriseEggStock` qui permet de gérer une liste d'œufs surprises représentés eux par des objets de la classe `SurpriseEgg`. Les œufs sont marqués par des lettres et contiennent des pièces de monnaie. D'un œuf à l'autre les pièces de monnaie diffèrent en valeur et en nombre ce qui entraîne que le poids varie aussi d'un œuf à l'autre.



Un modèle vous est fourni. Il contient déjà la classe `SurpriseEgg` toute entière ainsi que la classe `SurpriseEggStock` avec son constructeur et une méthode `clear` qui permet de vider la liste. Le constructeur crée une liste contenant quelques œufs avec des valeurs d'attributs aléatoires, pour vous permettre de tester plus facilement vos méthodes.

0. Copiez le modèle nommé **ExerciceN1_M** du répertoire de classe vers votre répertoire personnel.
1. Ajoutez la méthode `searchLightestEgg` qui retourne l'œuf le plus léger.
2. Ajoutez la méthode `searchHeaviestEgg` qui retourne l'œuf le plus lourd.

Réfléchissez à la question suivante avant d'écrire le code de la méthode :

Quelle est l'unique partie de l'algorithme qui change par rapport au point 1 de l'exercice ?



3. Ajoutez la méthode `searchLeastCoins` qui retourne l'œuf contenant le moins de pièces de monnaie.
4. Ajoutez la méthode `searchMostCoins` qui retourne l'œuf contenant le plus de pièces de monnaie.
5. Ajoutez la méthode `searchSmallestCoinValue` qui retourne l'œuf avec les pièces de monnaie de la plus petite valeur.
6. Ajoutez la méthode `searchGreatestCoinValue` qui retourne l'œuf avec les pièces de monnaie de la plus grande valeur.
7. Ajoutez la méthode `searchSmallestTotalValue` qui retourne l'œuf avec la plus petite somme d'argent.

Exemple :

Index	Value
0	Egg B: 8.2g, 2 x 10c = 20c
1	Egg F: 3.92g, 1 x 5c = 5c
2	Egg D: 3.06g, 1 x 2c = 2c
3	Egg E: 15.3g, 5 x 2c = 10c
4	Egg C: 6.12g, 2 x 2c = 4c
5	Egg A: 3.92g, 1 x 5c = 5c

Pour la liste sur l'image la méthode retourne l'œuf à l'index 2, comme il contient la plus petite somme d'argent (2c).

8. Ajoutez la méthode `searchGreatestTotalValue` qui retourne l'œuf avec la plus grande somme d'argent.
9. Ajoutez la méthode `indexOfLightestEgg` qui retourne l'index de l'œuf le plus léger.

10. Ajoutez la méthode `indexOfHeaviestEgg` qui retourne l'index de l'œuf le plus lourd.
11. Ajoutez la méthode `indexOfLeastCoins` qui retourne l'index de l'œuf contenant le moins de pièces de monnaie.
12. Ajoutez la méthode `indexOfMostCoins` qui retourne l'index de l'œuf contenant le plus de pièces de monnaie.
13. Ajoutez la méthode `indexOfSmallestCoinValue` qui retourne l'index de l'œuf avec les pièces de monnaie de la plus petite valeur.
14. Ajoutez la méthode `indexOfGreatestCoinValue` qui retourne l'index de l'œuf avec les pièces de monnaie de la plus grande valeur.
15. Ajoutez la méthode `indexOfSmallestTotalValue` qui retourne l'index de l'œuf avec la plus petite somme d'argent.

Exemple :

Index	Value
0	Egg B: 8.2g, 2 x 10c = 20c
1	Egg F: 3.92g, 1 x 5c = 5c
2	Egg D: 3.06g, 1 x 2c = 2c
3	Egg E: 15.3g, 5 x 2c = 10c
4	Egg C: 6.12g, 2 x 2c = 4c
5	Egg A: 3.92g, 1 x 5c = 5c

Pour la liste sur l'image la méthode retourne 2, comme l'œuf à l'index 2 contient la plus petite valeur d'argent (2c).

16. Ajoutez la méthode `indexOfGreatestTotalValue` qui retourne l'index de l'œuf avec la plus grande somme d'argent.