Qualité du logiciel et des métriques

Devoir 3

Équipier 1 : Emily Bazile-Supplice (20200848)

Équipier 2 : Eed Flory Jean-Baptiste (20168335)

## Tâche 1

**Objectif :** Concevoir et exécuter des tests boîte noire pour le système de conversion de devises (« Currency Converter ») selon les spécifications suivantes :

- Il doit convertir des montants entre les devises suivantes : USD, CAD, GBP, EUR, CHF, AUD.
- Il doit seulement accepter des montants entre [0, 1 000 000].

### Currency.convert(Double, Double)

#### ➢ Test pour un cas de base

Ce test vise à confirmer que la fonction de conversion fonctionne correctement pour des valeurs attendues. Dans ce scénario, nous utilisons un taux de conversion positif ainsi qu'un montant initial positif.

Valeur initiale : 100$ — Valeur attendue : 150$ 

Taux de conversion : 1.5 — Valeur obtenue : 150$

#### ➢ Test montant initial nul

L'objectif de ce test est de vérifier la conversion d'un montant de 0$, ce qui représente une limite inférieure de nos spécifications.

Valeur initiale : 0$ — Valeur attendue : 0$ 

Taux de conversion : 1.5 — Valeur obtenue : 0$

#### ➢ Test montant initial 1 000 000$ .

L'objectif de ce test est de vérifier la conversion d'un montant de 1 000 000$, ce qui représente une limite supérieure de nos spécifications.

Valeur initiale : 1 000 000$ — Valeur attendue : 1 500 000$ 

Taux de conversion : 1.5 — Valeur obtenue : 1 500 000$

#### ➢ Test pour un taux nul

L'objectif de ce test est de vérifier la conversion d'un certain montant valide avec un taux de conversion nul, ce qui représente une limite inférieure pour le taux de conversion.

Valeur initiale : 100$ — Valeur attendue : 0$ 

Taux de conversion : 0 — Valeur obtenue : 0$

#### ➢ Test pour un montant Négatif

L'objectif de ce test est de vérifier la conversion d'un montant négatif avec un certain taux de conversion valide. Nous ne devrions pas être en mesure de convertir ce montant, car il est hors de nos spécifications.

Valeur initiale : -1$ — Valeur attendue : ThrowException 

Taux de conversion : 1.5 — Valeur obtenue : -1.5$

Le code ne prend pas en compte le cas où le montant entrée est négatif. Il devrait retourner un message d'erreur.

- Test pour un montant supérieur à la limite

Ce test vise à confirmer que la fonction de conversion fonctionne correctement pour des valeurs attendues. Dans ce scénario, nous utilisons un taux de conversion positif ainsi qu'un montant initial positif.

Valeur initiale : 1 000 001$ Valeur attendue : ThrowException

Taux de conversion : 1.5 Valeur obtenue : 1 500 001.5$

Le code ne prend pas en compte le cas où le montant est supérieur à 1 millions . Il devrait retourner un message d'erreur.

- Test pour un taux de conversion négatif

Implémentation de tests pour s'assurer que la conversion est précise et fiable entre chaque paire de devises. Chaque cas de tests convertit le montant de 100$ dans une autre devise.

Valeur initiale : 100$ Valeur attendue : ThrowException

Taux de conversion : -1.5 Valeur obtenue : -150$

Le code ne prend pas en compte le cas où le taux de conversion est invalide (négatif). Il devrait retourner un message d'erreur.

## MainWindow.convert(String, String, ArrayList<Currency>, Double)

- Test pour les cas de base

Ce test vise à confirmer le bon fonctionnement de la fonction de conversion pour chaque paire de devises spécifiées. Nous examinons toutes les combinaisons possibles de devises, en excluant celles qui ne font pas partie des spécifications (comme CNY et JPY). (valeur attendu/valeur obtenu)

| | USD | CAD | GBP | EUR | CHF | AUD |
|---|---|---|---|---|---|---|
| ***USD*** | 100/100 | 137/0 | 66/66 | 93/93 | 101/101 | 152/0 |
| ***CAD*** | 73/0 | 100/0 | 58/0 | 67/0 | 65/0 | 111/0 |
| ***GBP*** | 151/151 | 172/0 | 100/100 | 141/141 | 152/152 | 191/0 |
| ***EUR*** | 107.3/107.3 | 149/0 | 71/71 | 100/100 | 108/108 | 166/0 |
| ***CHF*** | 99/99 | 152/0 | 66/66 | 93/93 | 100/100 | 172/0 |
| ***AUD*** | 66/0 | 111/0 | 52/0 | 60/0 | 58/0 | 100/0 |

Les résultats montrent que la conversion se fait de manière adéquate pour la plupart des paires de devises, conformément aux spécifications. Cependant, les devises CAD et AUD, qui ne sont pas définies dans le tableau des devises, ne sont pas en mesure d'être converti. Il faudrait corriger ce bug.

- Test pour les cas invalides

L'objectif de ce test est de vérifier la conversion d'une devise invalide. Nous ne devrions pas être en mesure de convertir une devise qui n'existe pas.

Valeur initiale : 100$ Valeur attendue : 0

USD vers une Devise inconnue Valeur obtenue : 0

Valeur initiale : 100$

Devise inconnue vers USD

Valeur attendue : 0

Valeur obtenue : 0

## Tâche 2

Créer des tests boîtes blanches.

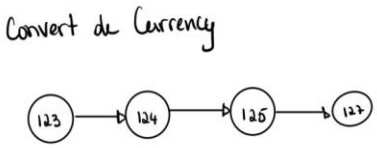
Testez les deux méthodes en utilisant les **5 critères de sélection de jeux de test** quand il fait du sens : couverture des instructions, couverture des arcs du graphe de flot de contrôle, couverture des chemins indépendants du graphe de flot de contrôle, couverture des conditions, couverture des i-chemins.

### Currency.convert(Double, Double) :

**A) Critère de couverture d'instructions :** La méthode est simple, avec seulement trois instructions principales, donc un seul ensemble de tests peut suffire à couvrir toutes les instructions.

**D1** : {(amount, exchangeValue) | Test qui couvre toutes les instructions} :
**Jeu de Test** : (100.25, 1.337) : le montant multiplié par la valeur d'échange nécessite un arrondi, couvrant ainsi toutes les instructions de la méthode convert.

**B) Critère de couverture des arcs du graphe de flot de contrôle :**




**Jeu de Test pour Currency.convert** : {amount = 100.25, exchangeValue = 1.337} couvre l'arc unique de la méthode.

**C) Critère de couverture des chemins indépendants du graphe de flot de contrôle :** C'est une méthode linéaire sans décisions, la complexité cyclomatique V(G) est 1, donc un seul chemin indépendant est nécessaire, qui est déjà couvert par un seul test comme décrit précédemment.

**D) Critère de couverture de conditions** :
Cette méthode ne contient pas de conditions composées. Elle réalise simplement une opération arithmétique suivie d'un arrondi. Par conséquent, il n'y a pas de conditions à tester et le critère de couverture des conditions ne s'applique pas.

**E) Critère de couverture de i chemins** : Ce critère ne s'applique pas car il n'y a pas de boucles.

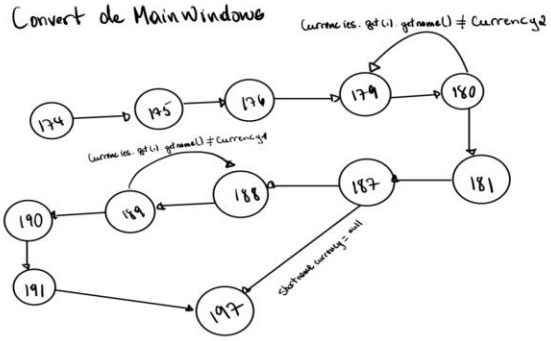
### MainWindow.convert(String, String, ArrayList<Currency>, Double)

A)Criteres de couverture d'instructions :
**D1** : (currency1 = "USD", currency2 = "EUR", amount = 100.0) pour tester la recherche réussie de currency2. Instructions des lignes 179-183.

**D2** : (currency1 = "USD", currency2 = "Fake Currency", amount = 100.0) pour tester le cas où currency2 n'est pas trouvé et donc shortNameCurrency2 reste null. Instructions des lignes 187-193 ne sont pas exécutées.

**D3** : (currency1 = "USD", currency2 = "EUR", amount = 100.0) pour tester le cas où currency1 et currency2 à être des devises valides . Instructions des lignes 187-193.

**B) Critère de couverture des arcs du graphe de flot de contrôle :**




**D1** : Couvre le cas où la devise currency2 n'est pas trouvée dans la liste des devises (currencies).

-Jeu de test pour D1 : {("USD", "ZZZ", currencies, 100.0)}

**D2** : Couvre le cas où la devise currency2 est trouvée, mais currency1 n'est pas trouvé par la seconde boucle for.

-Jeu de test pour D2 : {("ZZZ", "EUR", currencies, 100.0)}

**D3** : Couvre le cas où les devises currency1 et currency2 sont toutes deux trouvées.

-Jeu de test pour D3 : {("USD", "EUR", currencies, 100.0).

**C) Critère de couverture des chemins indépendants du graphe de flot de contrôle :** Selon la méthode MainWindow.convert, la complexité cyclomatique V(G) = nombre de décisions plus 1. Il y a deux boucles for et une instruction if, donc V(G)=3+1=4.

**Ensemble de base B** :

**D1** : Un chemin où aucune devise n'est trouvée.

-Jeu de test : {("XXX", "YYY", currencies, 100.0)}

-Chemin : 174-179-183-197 .

**D2** : Un chemin où currency2 est trouvé, mais currency1 n'est pas trouvé.

-Jeu de test : {("XXX", "EUR", currencies, 100.0)}.

-Chemin : 174-179-182-187-188-194-197 .

-**D3** : Un chemin où currency2 est trouvé et currency1 est aussi trouvé.

-Jeu de test : {("USD", "EUR", currencies, 100.0)}

-Chemin : 174-179-182-187-188-191-197 .

**D) Critère de couverture de conditions** : Dans la méthode convert, il y a des conditions, mais elles ne sont pas composées. Par exemple, nous avons une condition qui vérifie si une devise donnée est égale à une chaîne spécifique. Ces conditions sont indépendantes les unes des autres et ne sont pas combinées avec des opérateurs logiques pour former des conditions composées. Donc, bien que nous puissions tester chaque condition pour vrai et faux, il n'y a pas de conditions composées à décomposer .Par conséquent, Le critère de couverture des conditions ne s'applique pas.

**E) Critère de couverture de i chemins** : nous avons deux boucles for qui itèrent sur une collection de devises (currencies). Ces boucles sont utilisées pour rechercher des correspondances de devises dans la liste, donc elles peuvent être considérées comme des boucles simples non bornées, car leur exécution dépend de la présence de la devise dans la liste, ce qui est une condition externe et non un nombre fixe d'itérations.

1-**Cas où l'on saute la boucle** :

-Si la liste des devises est vide ou si la devise recherchée est en première position, ce qui entraînerait une sortie immédiate de la boucle.

-Jeu de test : Passer une liste vide.

**2-Cas d'une seule itération de la boucle** :

-Si la devise recherchée est trouvée lors de la première itération.

-Jeu de test : Passer une liste où la devise recherchée est le premier élément.

3-**Cas de deux itérations** :

-Si la devise recherchée est trouvée lors de la deuxième itération.

-Jeu de test : Passer une liste où la devise recherchée est le deuxième élément.

4-**Cas de m itérations (m < n)** :

-Tester avec un nombre d'itérations inférieur au nombre total de devises dans la liste.

-Jeu de test : Passer une liste où la devise recherchée est avant la dernière position.