RAPPORT IFT3913 TP4

Yann-ariel Ananga 20172516 et Aleck Gibbs – 20096971

Repository : https://github.com/Yannari/IFT3913TP2

1- Test Boites Noires

Nous devons tester la méthode convert() en effectuant des tests boîte noire. La spécification exige que les devises soient {USD, CAD, GBP, EUR, CHF, INR, AUD} et le montant dans l'intervalle [0,10000]. Nous avons choisi deux devises de la spécification , respectivement {CAD, EUR} et fait en sorte qu'il renvoie une erreur lorsqu'une devise inconnu est utilisée. Nous avons aussi pris des valeurs de montants et des valeurs frontières. Pour chaque valeur nous avons tester la fonction avec deux fois la même devise valide, deux devises valides différentes, une devise valide et une non valide. Ce qui nous fais 18 tests à effectuer.

Les fonctions sont telles qu'elles :

```
public void currency31() throws ParseException{
        try{
            conversion  = new OfflineJsonWorker().parser();
            double output = CurrencyConvertor.convert(9999, "CAD" , "USD",
conversion);
            assertTrue(output == 9999);
        }
        catch(Exception e)
        {
            fail("currency not currently support");
        }
    }
```

Les résultats obtenus montrent que peu importe les devises et les montants, le calcul manuel donne le même résultat que convert(). Cependant, convert() répond plus large que les spécifications et accepte les nombres négatifs et plus grands que 10 000. Seuls les cas qui respectent les spécifications sont autorisés et les autres doivent renvoyer une exception.

Conclusion :
Les tests n'ont pas tous été couronnés de succès. La méthode va plus loin que ce qui était prévu. Plus précisément, nos résultats démontrent qu'elle prend en compte un nombre plus important de monnaies, tolère les nombres négatifs et les nombres supérieurs à 10 000. Cependant, lorsque les tests sont effectués selon les spécifications demandées, la méthode envoie les bonnes réponses, même pour les frontières.

2.Test boîte blanche

Pour tester la méthode convert(), nous avons utilisés le critère de couverture des instructions ,critère de couverture des arcs du graphe de flot de contrôle et le critère de couverture des

conditions. Car la méthode convert() ne contient aucune boucle et contient seulement un chemin de base. Ce qui rend le critère de couverture des I-chemins et le critère de couverture des chemins indépendants du graphe de flot de contrôle non-applicable.

Critère de couverture des instructions :

Pour ce critère, nous avons fait trois test. Nous avons fais un test de conversion de CAD vers USD de 12.00$. Ensuite, un test de conversion de USD vers CAD de 8.92624 $ pour s'assurer que la conversion était consistante. Finalement, notre dernier test de conversion était de JPY vers JPY de 12.00$ pour s'assurer q'un taux de conversion n'est pas forcément toujours appliqué.

Les hypothèses pour ces tests ont tous été validé. La méthode convert() est bien consistante dans sa conversion, elle n'applique pas forcément une conversion quand elle est utilisé.

critère de couverture des arcs du graphe de flot de contrôle :

Pour ce critère, vue qu'il n'ya qu'une seule instruction conditionnelle et que le fonctionnement de base de la méthode à été vérifier au premier critère nous avons fais 3 tests. Nous avons vérifier que si la variable gauche de l'instruction conditionnelle (if variable_gauche or variable_droite) était vrai et que la variable droite de l'instruction était fausse alors l'instruction serait executé. Ensuite, nous avons vérifier que si la variable droite de l'instruction conditionnelle (if variable_gauche or variable_droite) était vrai et que la variable gauche était fausse alors l'instruction serait executé. Finalement, nous avons vérifier que si les deux variable étaient vrai alors l'instruction serait executé.

Les hypothèses pour ces tests ont tous été validé. L'instruction conditionnelle de la méthode convert() s'exécute correctement.

critère de couverture des conditions :

Pour ce critère, nous avons fais 7 tests. Nous avons vérifier que les erreurs qui pourraient se retrouver dans la méthode étaient pris en compte. Nous avons commencer par passer en paramêtre un string vide au paramêtre gauche et droite de la méthode. Ensuite, nous avons tester de mettre le plus grand bit-double positive et négative que nous pouvions comme montant dans le but de voir si la méthode pourrait les traiter correctement. Finalement, nous avons remplacer les strings de la méthode par des strings de nombre pour s'assurer du bon fonctionnement.

Les hypothèses pour ces tests ont tous été validé. La méthode traite correctement les données entrante.