TP 4

IFT-3913 Qualité de Logiciel et Métriques

Remis par:

Olivier Lemay-Caron
20162110

et

Derrick Chung
20188514

16 Décembre 2022

Tests à boîte noire:

Pour les tests boîte noire, nous avons testé le domaine du montant et le domaine des devises. Nous avons utilisé le OfflineJsonWorker pour la table de conversion. Nous n'avons pas vérifié que les valeurs retournées par le convertor soient bonnes, car ce n'était pas dans les spécifications. Pour nos tests, nous considérons que si la devise entrée n'est pas dans les spécifications ou du même format que ceux des spécifications, ça ne devrait pas fonctionner/devrait lancer une exception. Nous considérons aussi que si le montant ne se trouve pas entre 0 et 10000 inclusivement, la conversion ne devrait pas fonctionner/devrait lancer une exception.

Pour les cas acceptants, nous avons testé toutes les valeurs au centième près entre 0 et 10000, inclusivement, pour chaque devise à tous les autres devises acceptées. Les conversions devraient fonctionner et le *Convertor* a effectivement retourné des valeurs sans erreurs, donc ce test a passé.

Pour les cas rejetants du domaine du montant, nous avons testé les valeurs -1 et 10001 pour toutes les conversions de devises acceptées. Les conversions ne devraient pas fonctionner selon les spécifications, mais le *Convertor* retourne quand même des valeurs, donc le test a échoué.

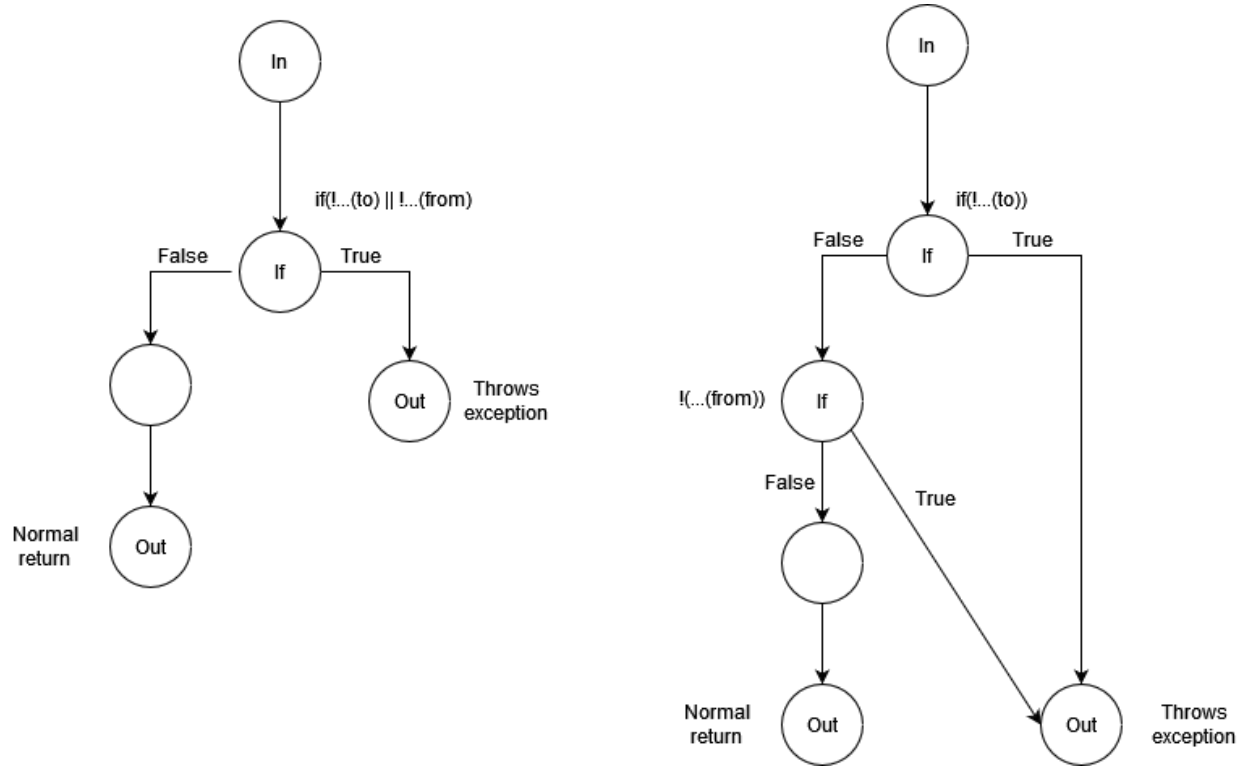
Pour les tests suivants, la valeur du montant est de 1.
Pour les cas rejetants du domaine des devises, nous avons testé les conversions vers et à partir de "bidon", qui n'est pas une devise acceptée, avec toutes les autres devises acceptées par les spécifications. Ce cas a échoué et a été géré par le programme, donc ce test a passé.

Nous avons aussi testé les devises acceptées en lettre minuscule. Ce cas a échoué et a été géré par le programme, donc ce test a passé.

Nous avons aussi testé les devises acceptées selon le code numérique du pays, selon https://www.iban.com/country-codes et investopedia. Ce cas a échoué et a été géré par le programme, donc ce test a passé.

Nous avons aussi testé les conversions vers et à partir de FJD, qui n'est pas une devise acceptée selon les spécifications, avec toutes les autres devises acceptées par les spécifications. Les conversions ne devraient pas fonctionner selon les spécifications, mais le *Convertor* retourne quand même des valeurs, donc ce test a échoué.

Tests de boîte blanche:
Nous avons commencé par analyser le code à tester pour trouver les critères de tests pertinents. Pour cela nous avons créé le premier flot de contrôle (celui de gauche).



Ce premier graphe était suffisant pour couvrir les 3 premiers critères de sélection, soit la couverture des instructions, la couverture des arcs du graphe de flot de contrôle et, comme il n'y a pas de boucle dans le code, le critère des chemins indépendants du graphe. Il n'y a alors que deux chemins a couvrir: le coté true du if vers la sortie avec exception, et la sortie false du if, qui est le flot normal du code vers un return. On voit aussi facilement dans le code que si on obtient un true sur le if dans au moins un chemin, et dans un autre test un false, on a couverture des instructions, car il n'y a pas de saut de ligne ou d'arrêt prématuré possible.

Cependant, au critère de couverture des conditions, on voit que notre graphe n'est pas adéquat, car sélecteur du if contient une condition OU de deux termes. Il faut donc décomposer notre condition pour s'assurer que les deux côtés sont calculés au moins une fois chaque. Cette décomposition est illustrée dans le graphe de droite. Nous nous retrouvons donc avec un graphe avec 3 chemins possible, 1 pour chaque réussite de nos prédicats, et un pour l'échec double qui nous dirige vers le out normal.

Comme il n'y a pas de boucles, nous pouvons ignorer le critère de couverture des i-chemins.

Nous avons donc créé cet ensemble de test pour les entrées de forme {amount, from, to, conversion}:

Chemin de la première condition:
T1 = {1, "USD", "pouet", conversion}
Parcoure le chemin IN-True-Out(Exception).

Chemin de la deuxième condition:
T2= {1, “pouet”, “CAD”, conversion}
Parcoure le chemin IN-False–True-Out(Exception)

Chemin normal
T3={0, “USD”, “CAD”, conversion}
Parcoure le chemin IN-False-False-Out(Normal Return)

Pour ces jeux de données, le seul décideur du chemin étaient les attributs “to” et “from” qui sont nos critères de conditions pour les if. Les autres entrées sont des valeurs arbitraires qui respectent la spécification. Les valeurs pour obtenir des False dans le if sont aussi des valeurs sélectionnées dans les spécifications: on fait l’hypothèse que le code respecte les spécifications et donc que le comportement sera comme on le veut. Les données pour déclencher notre condition if sont donc des valeurs qui sortent du domaine de la spécification. Elles ont été spécialement choisie pour être d’un format autre, et en minuscule, dans l’idée que si le programme est étendu a de nouvelles devises, le test fonctionne encore, car les noms de devises sont en majuscules et sont longues de 3 lettres, alors que nos valeurs sont une suite de 5 lettres minuscules.

On est heureux de constater que nos tests de boîte blanche réussissent et retourne bien les exceptions spécifiées quand les conditions du if sont déclenchées.

Lien vers le repo:
https://github.com/derrick-chung/IFT3913_TP4