# Documentez et expliquez comment vous avez choisi les cas de tests que vous avez implémenté.

Pour les tests de boite noire, on a voulu tester que le programme fait correctement la conversion des montants entre les devises suivantes : USD, CAD, GBP, EUR, CHF, CNY. Comme on n’a pas la connaissance de l’exécution des fonctions, on prend deux chemins qui devront mener au bon résultat. Dans notre cas one a choisi de commencer un de chemins de partir : USD -> Euro -> Pound -> Yuan et l’autre chemin de partir de USD -> Franc -> Yen -> Yuan. Ainsi, on suppose que le code va nous retourner le même montant à la fin si on commence avec le même montant.

Maintenant, le choix des montants, on suppose que le code peut prendre un montant de nature Double. Alors, on suppose l’input valide serait entre [0, infinie [. Les choix alors qu’on va tester sont :

Un montant négatif

Un montant normal (20)

Un montant nul (0)

Pour les valeurs frontières, on va prendre la plus grande valeur possible pour un double et ainsi essayer de créer un over flow

Un montant Double.MAX\_VALUE

# Décrivez les résultats de vos tests et donnez vos observations sur les différentes approches et critères de test.

Pour les tests de boite noire :

Un montant négatif : Test : On veut voir si quand on essaye de convertir un montant négatif, est-ce que la fonction donne une exception ou tout simplement retourne zéro.

Résultat : la fonctionne retourne une réponse négatif. Ceci indique que le développeur n’a pas pris en compte les inputs négatifs.

Un montant normal (20) : Test : on voudrait avoir le même résultat après avoir fait le même type de conversion.

Résultat : (Expected :126.82 / Actual :126.55) on n’obtient pas le même résultat. L’erreur est un numéro après la virgule. Ceci nous indique qui se pourrait qu’il ait un manque de précision. Mais, ceci est un problème récurrent, car il est très difficile pour un système informatique de manipuler des nombre réels comme 1/3. A force de manipuler les numéros, il aura des arrondies qui créeront des manques de précision.

Un montant nul (0) : Test : on veut tester le problème de division par 0.

Résultat : le système gère correctement un input de 0.

Pour les valeurs frontières, on va prendre la plus grande valeur possible pour un double et ainsi essayer de créer un over flow

Un montant Double.MAX\_VALUE : Test : on veut voir s’il y a une overflow à cause de valeurs qui sont trop grandes.

Résultat : On remarque que le logiciel ne donne pas d’exception et continue à exécuter le code avec des numéros très grand. Je pense que c’est meilleur lorsque que le chiffre est trop grand, de retourner une exception et indiquer au client que le montant qu’il a mis est trop grand et il n’est pas possible de le convertir correctement.