1. **Boite Noire**
2. Partition du domaine des entrées en classes d’équivalence:

**Partitionnement sur la spécification il doit accepter des montants dans l’intervalle suivant [0, 1 000 000]:**

Classes d’équivalence:

- Montant < 0 ;

- Montant = 250 000 ;

- Montant > 1 000 000;

**Jeu de test** : T= {-10,250000,1 000 060}

**Partitionnement sur la spécification le Converter dois accepter des montants entre les devises suivantes** :

USD, CAD, GBP, EUR, CHF, AUD:

**Classes d'équivalence**:

- Devise différente de [USD, CAD, GBP, EUR, CHF, AUD] ;

- Devise faisant partir de [USD, CAD, GBP, EUR, CHF, AUD] ;

Jeu de test : T= {USD, GNF}

**Hypothèses**:

Pour les montants en dehors de l'intervalle [0, 1 000 000] et pour des devises différentes de celles spécifiées, le code peut lever une exception ou en renvoyer un résultat non défini. Il peut aussi spécifier à l'utilisateur par des logs que les entrées ne respectent pas la spécification.

B- Analyse des valeurs frontières

Spécification il doit accepter des montants dans l’intervalle suivant [0, 1 000 000]

**Jeu de test** :

T= {(typique:10, frontiere:1), (typique:250000, frontiere:500000), (typique:1 000 060, frontiere:1000001)}

Pour la méthode : **currencyConverter.Currency. convert (Double, Double)**

Nous effectuons des vérifications sur des taux de change positif et négatif, sur des montant égale à 0, des taux de change de zéro et sur des valeurs limites et extrêmes.

Nous pouvons remarquer en analysant en boite noire que cette méthode n’exclut pas les éléments qui ne font pas partir de la spécification et elle ne fait pas vraiment de gestion d’erreur.