Rapport Mega Machine à Caoua

Paul Ecoffet Mathieu Seurin

Vendredi 28 Novembre 2014

Nous allons ici détailler les fonctions utilisées dans notre programme (signature et axiome) ainsi que faire l'analyse de leur complexité en notation **O**. Après chaque méthode, nous indiquerons le fichier correspondant aux tests que nous avons effectués, ainsi que les noms des tests.

Fonctions de Machine, mode fonctionnement

order :

- 1. signature : (Monnaie, Commande) $\Rightarrow$ ($Boisson \cup \emptyset \times$ Monnaie)
  2. axiome :
  3. Complexité : ON DOIT FAIRE LES AUTRES AVANT
  4. Test : *test_machine.py*
     - test_order_simple()
     - test_order_complex()
     - test_order_fail_not_enough_cash()
     - test_order_fail_not_drink()
     - test_order_fail_no_stock()
     - test_order_cant_get_maxcash()

Fonctions de Machine, mode Maintenance

edit_prices :

- 1. signature : (dictionnaire_prix) $\Rightarrow \emptyset \cup$ Error
  2. axiome :
     - $\forall$ stock $\in$ {'thé', 'café', 'lait', 'chocolat'},
       $\forall$ prix ≥ 0
       edit_prices(stock=prix) $\Rightarrow$ machine._stock_prices[stock] = prix

- Si stock = 'sucre' $\forall$ 0 $\leq$ prix_i $\leq$ prix_{i+1}, i $\in$ [0,3] edit_prices(sucre= [prix_i])
  machine._stock_prices[stock] = [(prix_i)] $\forall$i $\in$ [0,3]

3. Complexité : $O(n)$ avec n le nombre de produits payant
4. Test : *test_machine.py*
   - test_edit_prices

edit_stocks :

- 1. signature : (dictionnaire_stocks) $\Rightarrow \emptyset \cup$ Error
  2. axiome :
     - $\forall$ stock $\in$ {'thé', 'café', 'lait', 'chocolat', 'sucre'},
       $\forall$ machine.quantite[stock] < quantite $\leq$ machine.quantite_max[stock]
       machine.edit_stock(stock=quantite) $\Rightarrow$ machine.quantite[stock] = quantite
     - $\forall$ stock $\in$ {'thé', 'café', 'lait', 'chocolat', 'sucre'},
       $\forall$ quantite $\leq$ machine.quantite[stock] ou quantite > machine.quantite_max[stock]
       machine.edit_stock(stock, quantite) $\Rightarrow$ machine.quantite[stock] = machine.quantite[stock]
  3. Complexité : $O(n)$ avec n le nombre de stocks différents
  4. Test : *test_machine.py*
     - test_edit_stocks

refill_stocks :

- 1. signature : $\emptyset \Rightarrow \emptyset$
  2. axiome :
     - $\forall$ stock $\in$ {'thé', 'café', 'lait', 'chocolat', 'sucre'},
       machine.refill_stock() $\Rightarrow$ machine.quantite[stock] = machine.quantite_max[stock]
  3. Complexité : $O(n)$ avec n le nombre de stocks différents
  4. Test : *test_machine.py*
     - test_edit_prices

edit_coins :

- 1. signature :
  2. axiome :
  3. Complexité :
  4. Test : *test_machine.py*
     - test_edit_prices

refill_coins :

- 1. signature : 2.axiome :
  2. Complexité :
  3. Test : *test_machine.py*
     - test_edit_prices

_remove_stocks :
1. signature : 2.axiome : 3. Complexité : 4. Test : *test_machine.py* test_edit_prices

-

Fonctions de Coins