École Polytechnique de Montréal

Baccalauréat en Génie Logiciel

Cours - LOG1000

Travail pratique 4

Auteurs :

Maxime Tremblay-Rheault - 1946878
Maude Desrosiers-Carbonneau - 1946801

17 novembre 2019

4.1 Test unitaire

Voici l'ajout du test unitaire trivial. L'affichage 'OK (1 tests)' indique que le code cpp_unit a été écrit correctement et que le Makefile est adéquat.

```
13  void RabaisTest::test_quelconque() {
14      int i = 1;
15      CPPUNIT_ASSERT_EQUAL(1, i);
16  }
```

```
./test/bin/exemple_test
.

OK (1 tests)
```

4.2 Graphe de flot de contrôle du code

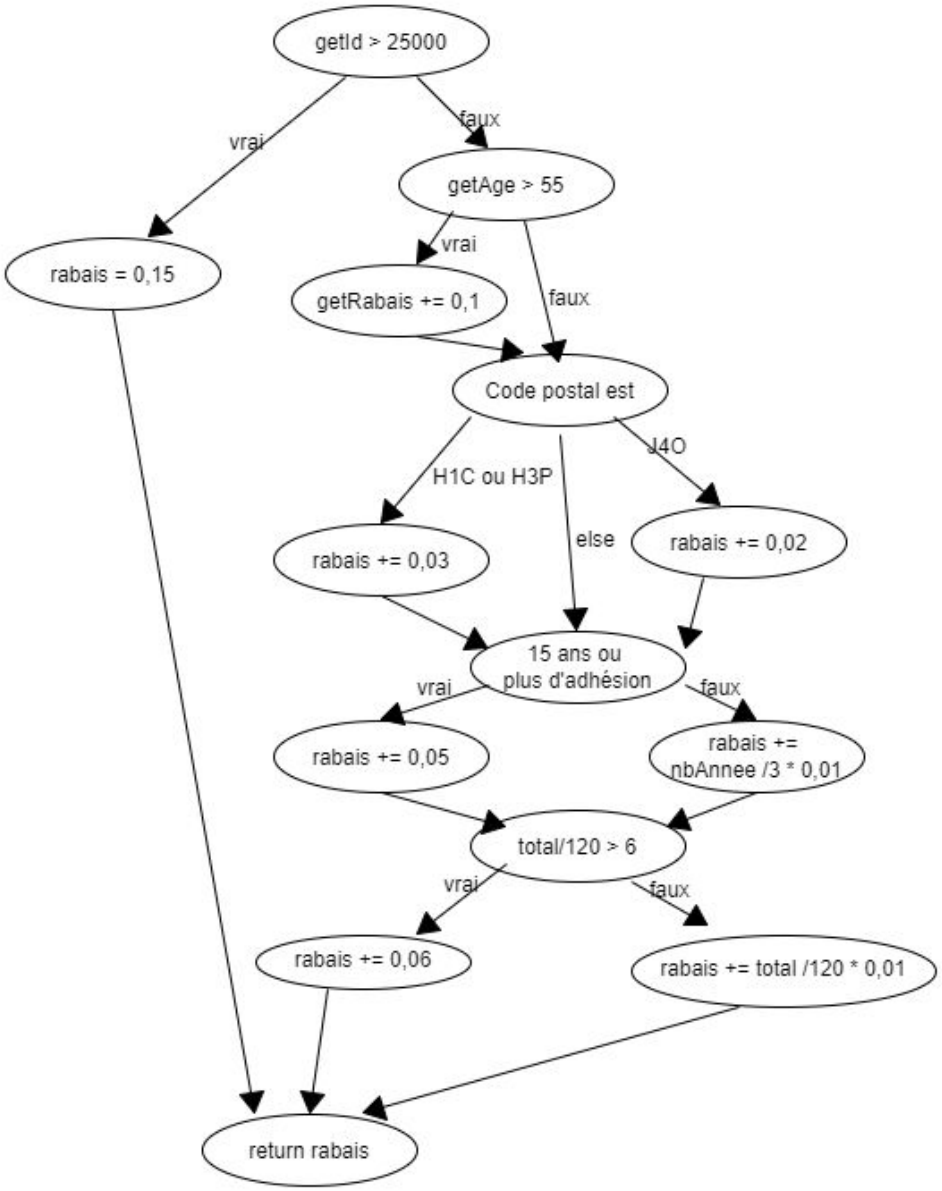

Figure 3 : CFG représentant le code

4.3 Cas de tests pour la couverture des branches

d1 = <{f, 10456 },{0.07}>
f est un objet facture qui a un attribut items qui contient le montant total de 165.30
ce qui équivaut actuellement aux factures ajoutées dans le main.
La valeur attendue est de 0.073775 et est arrondie à 0.07.
d2 = <{f, 14770},{0.17}>
f est un objet facture qui a un attribut items qui contient le montant 120
d3 = <{f,15034}, {0.09}>
f est un objet facture qui a un attribut items qui contient le montant 840
d4 = <{f,25102},{0.15}>
d5 = <{f,11111},{0.22}>
client ajouté 11111 Boquet Bill 99 J40 1980-12-01

4.4 Cas de tests dans le projet CppUnit

La figure suivante présente un exemple de l'implémentation des cas de tests dans CppUnit.

```
18  //Correspond à d1 = <{f, 10456}, {0.07}>
19  //f est une facture qui s eleve a une montant de 165.30$
20  void RabaisTest::d1() {
21      Facture une_facture;
22      une_facture.ajouterItem(50.95);
23      une_facture.ajouterItem(48.95);
24      une_facture.ajouterItem(25.45);
25      une_facture.ajouterItem(39.95);
26
27      float rabais1 = this->rabais_a_tester->getRabais(une_facture, 10456);
28
29      CPPUNIT_ASSERT_EQUAL(float(0.07), rabais1);
30  }
31
32  // Correspond à d2 = <{f, 14770}, {0.17}>
33  // f est une facture avec un item initalise a 120$
34  void RabaisTest::d2() {
35      Facture f = Facture();
36      f.ajouterItem(120);
37      float rabais1 = this->rabais_a_tester->getRabais(f, 14770);
38
39      CPPUNIT_ASSERT_EQUAL(float(0.17), rabais1);
40  }
```

Figure 4: Test d1 et d2 dans CppUnit

4.5 Identification d'un défaut

Plusieurs défauts ont étés identifiés notamment par rapport à l'année d'adhésion et par rapport au traitement du rabais maximal pour une adhésion de longue durée. Le défaut qui sera discuté ici concerne le rabais offert selon le code postal.

En effet, un client qui habite dans la zone H1C obtient actuellement seulement 3% de rabais alors qu'il est exigé qu'il reçoive 4% de rabais. Cette exigence porte le ID SRS05.

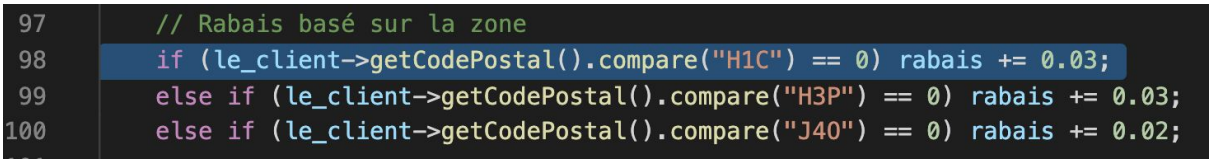

Figure 5: Erreur dans le code

Une correction possible du code serait de remplacer la ligne 98 par la suivante:

if (le_client->getCodePostal().compare(“HIC”)== 0) rabais += 0.04;