
MÉTHODES ET OUTILS POUR LA QUALITÉ LOGICIELLE

TP 4 – TESTS UNITAIRES (DURÉE 3H)

EXERCICE 1 – COUVERTURE DU CODE

QUESTION 1 – INSTRUMENTATION DU CODE ET GCOV

Nous avons ajouté la ligne au fichier Makefile.mak.

Nous avons ensuite compilé et lancé le programme et les fichiers en .gncv et .gcda apparaissent.

Dans le terminal de commande, nous avons tapé la commande `make gcov` et nous avons obtenu le résultat suivant :

```
X:\Semestre_8\MOQL\TP4\TP4>mingw32-make -f Makefile.mak gcov
gcov Cashregister.c Productdatabase.c Specialoffer.c Userinterface.c Main.c
File 'Cashregister.c'
Lines executed:50.89% of 112
Creating 'Cashregister.c.gcov'

File 'Productdatabase.c'
Lines executed:32.14% of 28
Creating 'Productdatabase.c.gcov'

File 'Specialoffer.c'
Lines executed:0.00% of 12
Creating 'Specialoffer.c.gcov'

File 'Userinterface.c'
Lines executed:77.91% of 86
Creating 'Userinterface.c.gcov'

File 'Main.c'
Lines executed:100.00% of 8
Creating 'Main.c.gcov'

Lines executed:57.32% of 246
```

Comme on peut le constater, toutes les lignes du code ne sont pas utilisées.

En testant un maximum de possibilité, nous obtenons ces résultats :

```
X:\Semestre_8\MOQL\TP4\TP4>mingw32-make -f Makefile.mak gcov
gcov Cashregister.c Productdatabase.c Specialoffer.c Userinterface.c Main.c
File 'Cashregister.c'
Lines executed:58.04% of 112
Creating 'Cashregister.c.gcov'

File 'Productdatabase.c'
Lines executed:32.14% of 28
Creating 'Productdatabase.c.gcov'

File 'Specialoffer.c'
Lines executed:0.00% of 12
Creating 'Specialoffer.c.gcov'

File 'Userinterface.c'
Lines executed:88.37% of 86
Creating 'Userinterface.c.gcov'

File 'Main.c'
Lines executed:100.00% of 8
Creating 'Main.c.gcov'

Lines executed:64.23% of 246
```

On peut voir que le total de lignes exécutées est passé de 57.32% à 64,23%. Cependant, le fichier Specialoffer.c n'est pas utilisé malgré le fait d'avoir 3 kiwis dans le panier qui correspond à une offre de réduction.

La commande make gcov_full teste toutes les fonctions et affiche le pourcentage de lignes utilisées par fonction. Dans la plupart des cas, toutes les lignes sont utilisées mais on peut voir par exemple que dans « Specialoffer.c », aucune ligne n'est exécutée :

```
File 'Specialoffer.c'
Lines executed:0.00% of 12
Branches executed:0.00% of 6
Taken at least once:0.00% of 6
No calls
Creating 'Specialoffer.c.gcov'
```

QUESTION 2 – ANALYSE FINE DES RÉSULTAT DE GCOV

Pour commencer, nous avons copié notre dossier X:\TP4 dans documents.

Ensuite, nous avons précisé le chemin d'accès à python dans le fichier Makefile.mak :

```
37 gcovr: $(GCOVR_DIR) $(OUT)
38     C:\python\Python311\python -m gcovr --output $(GCOVR_DIR)/gcovr.html --html-details
39
40 $(GCOVR_DIR):
41     mkdir $@
42
43 install_gcovr:
44     C:\python\Python311\python -m pip install gcovr
45
```

Nous avons lancé le script Clean.bat puis nous avons relancé le programme Run.bat.

Nous avons exécuté la commande ./make install_gcovr puis ./make gcovr.

Grâce aux fichiers .gcda et .gcno, la commande crée un dossier contenant des fichiers .html :

Ce PC > Documents > TP4 > gcovr_out				
Nom	Modifié le	Type	Taille	
gcovr.Cashregister.c.5925a435f58721b86...	07/02/2024 10:31	Microsoft Edge H...	110 Ko	
gcovr.css	07/02/2024 10:31	Document de feui...	14 Ko	
gcovr.functions.html	07/02/2024 10:31	Microsoft Edge H...	7 Ko	
gcovr.html	07/02/2024 10:31	Microsoft Edge H...	6 Ko	
gcovr.Main.c.3c338acc3e29fe8214f2e57d...	07/02/2024 10:31	Microsoft Edge H...	13 Ko	
gcovr.Productdatabase.c.610187172729d...	07/02/2024 10:31	Microsoft Edge H...	44 Ko	
gcovr.Specialoffer.c.0a63ec7683be3ecf50...	07/02/2024 10:31	Microsoft Edge H...	17 Ko	
gcovr.Userinterface.c.437221eb872171005...	07/02/2024 10:31	Microsoft Edge H...	67 Ko	

Voici les résultats que nous obtenons :

GCC Code Coverage Report

File	Lines		Functions		Branches	
Cashregister.c	<div><div></div></div>	42.9% 48 / 112	61.5% 8 / 13	32.5% 13 / 40		
Main.c	<div><div></div></div>	87.5% 7 / 8	100.0% 1 / 1	50.0% 1 / 2		
Productdatabase.c	<div><div></div></div>	32.1% 9 / 28	25.0% 1 / 4	28.6% 4 / 14		
Specialoffer.c	<div><div></div></div>	0.0% 0 / 12	0.0% 0 / 1	0.0% 0 / 6		
Userinterface.c	<div><div></div></div>	48.8% 42 / 86	66.7% 4 / 6	45.5% 10 / 22		

Directory: ./

Date: 2024-02-07 10:31:45

Coverage: low: ≥ 0% medium: ≥ 75.0% high: ≥ 90.0%

Exec Total Coverage

Lines:	106	246	43.1%
Functions:	14	25	56.0%
Branches:	28	84	33.3%

Le taux de couverture Main.c est de 87.5%.

Le taux de couverture le plus faible est celui du fichier Specialoffer.c.

Le fichier Userinterface.c possède 22 chemins d'exécution.

EXERCICE 2 – TESTS UNITAIRES

QUESTION 1 – DÉTERMINATION DU TAUX DE COUVERTURE DES TESTS D'ORIGINE


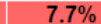


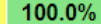


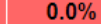

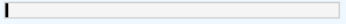
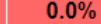

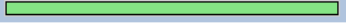
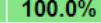

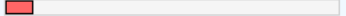


Voici notre résultat après la commande `./make unit_test` :

```
PS C:\Users\23006020\Documents\TP4> ./make unit_test
C:\Users\23006020\Documents\TP4>mingw32-make -f Makefile.mak unit_test
g++ -O0 -g3 -W -I. -MMD -MP --coverage -ftest-coverage -fprofile-arcs -w -fpermissive -I"C:\MOQL\boost_1_78_0"
ter.c Productdatabase.c Specialoffer.c Userinterface.c Test.cpp -o test_cash.exe
test_cash.exe
Running 1 test case...
*** No errors detected
```

Nous lançons la commande `./make gcovr` :

```
PS C:\Users\23006020\Documents\TP4> ./make gcovr
C:\Users\23006020\Documents\TP4>mingw32-make -f Makefile.mak gcovr
C:\python\Python311\python -m gcovr --output gcovr_out/gcovr.html --html-details
(INFO) Reading coverage data...
(INFO) Writing coverage report...
```

Les fichiers `.html` ont bien été créés. Voici nos résultats :

File	Lines		Functions		Branches	
Cashregister.c		2.7% 3 / 112		7.7% 1 / 13		1.5% 1 / 66
Main.c		87.5% 7 / 8		100.0% 1 / 1		50.0% 1 / 2
Productdatabase.c		0.0% 0 / 28		0.0% 0 / 4		0.0% 0 / 14
Specialoffer.c		0.0% 0 / 12		0.0% 0 / 1		0.0% 0 / 6
Test.cpp		100.0% 5 / 5		100.0% 2 / 2		50.0% 61 / 122
Userinterface.c		8.1% 7 / 86		16.7% 1 / 6		13.6% 3 / 22

Commande `./make gcov` :

```
PS C:\Users\23006020\Documents\TP4> ./make gcov
C:\Users\23006020\Documents\TP4>mingw32-make -f Makefile.mak gcov
gcov Cashregister.c Productdatabase.c Specialoffer.c Userinterface.c Main.c
File 'Cashregister.c'
Lines executed:2.68% of 112
Creating 'Cashregister.c.gcov'

File 'Productdatabase.c'
Lines executed:0.00% of 28
Creating 'Productdatabase.c.gcov'

File 'Specialoffer.c'
Lines executed:0.00% of 12
Creating 'Specialoffer.c.gcov'

File 'Userinterface.c'
Lines executed:8.14% of 86
Creating 'Userinterface.c.gcov'

File 'Main.c'
Lines executed:87.50% of 8
Creating 'Main.c.gcov'

Lines executed:6.91% of 246
```

Commande ./make gcovr :

File	Lines	Functions	Branches
Cashregister.c	2.7% 3 / 112	7.7% 1 / 13	1.5% 1 / 66
Main.c	87.5% 7 / 8	100.0% 1 / 1	50.0% 1 / 2
Productdatabase.c	0.0% 0 / 28	0.0% 0 / 4	0.0% 0 / 14
Specialoffer.c	0.0% 0 / 12	0.0% 0 / 1	0.0% 0 / 6
Test.cpp	100.0% 5 / 5	100.0% 2 / 2	50.0% 61 / 122
Userinterface.c	8.1% 7 / 86	16.7% 1 / 6	13.6% 3 / 22

Commande ./make gcov_full :

```
Function 'Userinterface_parse(char)'
Lines executed:14.89% of 47
Branches executed:100.00% of 18
Taken at least once:16.67% of 18
Calls executed:7.14% of 14
```

1	LDRA_uint32_t Userinterface_parse(const LDRA_char_t aChar)
	{
	LDRA_uint32_t menu;
1	if ((aChar >= '0') && (aChar <= '9'))
	{
X	Cashregister_key((const LDRA_uint32_t)aChar - (const LDRA_uint32_t)'0');
X	menu = 0u;
	}
	else
	{
1	switch (aChar)
	{
	case 'b':
1	Cashregister_code();
1	menu = 1u;
1	break;
	case 'n':
	/* Coconuts */
X	Cashregister_barcode(12340U);
X	menu = 2u;
X	break;
	case 'l':
	/* Lychees */
X	Cashregister_barcode(12346U);
X	menu = 3u;
X	break;
	case 'k':
	/* Kiwis */
X	Cashregister_barcode(12347U);
X	menu = 4u;
X	break;

Function (File:Line)	Call count	Block coverage
UT_Userinterface_parse::menu_values::test_method() (Test.cpp:12)	called 1 time, returned 1 time	58.0%
UT_Userinterface_parse::menu_values::invoker() (Test.cpp:12)	called 1 time, returned 1 time	69.0%

Le taux de couverture de la fonction Userinterface_parse est faible, seulement 7.14% des lignes sont exécutées. Comme on peut le voir ci-dessus, uniquement le case 'b' est exécuté. Cela vient du fait que nous avons d'abord utilisé Clean.bat puis en relançant le programme nous avons simplement lancé le random shopping une fois puis nous avons quitté. Cela veut dire que toutes les fonctions n'ont pas été utilisées.

QUESTION 2 - IMPLÉMENTATION DU PLAN DE TEST

```

7
8 // Test-suite
9 BOOST_AUTO_TEST_SUITE( UT_Userinterface_parse )
10
11 // Test-case : valeurs normales du menu
12 BOOST_AUTO_TEST_CASE( menu_values )
13 {
14     BOOST_TEST( Userinterface_parse('b') == 1u );
15     BOOST_TEST( Userinterface_parse('n') == 2u );
16     BOOST_TEST( Userinterface_parse('/') == 14u );
17     BOOST_TEST( Userinterface_parse('\n') == 13u );
18     BOOST_TEST( Userinterface_parse('q') == 12u );
19     BOOST_TEST( Userinterface_parse('r') == 11u );
20     BOOST_TEST( Userinterface_parse('s') == 10u );
21     BOOST_TEST( Userinterface_parse('e') == 9u );
22     BOOST_TEST( Userinterface_parse('c') == 8u );
23     BOOST_TEST( Userinterface_parse('w') == 7u );
24     BOOST_TEST( Userinterface_parse('g') == 6u );
25     BOOST_TEST( Userinterface_parse('p') == 5u );
26     BOOST_TEST( Userinterface_parse('k') == 4u );
27     BOOST_TEST( Userinterface_parse('l') == 3u );
28     BOOST_TEST( Userinterface_parse('n') == 2u );
29     BOOST_TEST( Userinterface_parse('b') == 1u );
30     BOOST_TEST( Userinterface_parse('9') == 0u );
31     BOOST_TEST( Userinterface_parse(':') == 14u );
32     BOOST_TEST( Userinterface_parse('0') == 0u );
33 }
34
35 BOOST_AUTO_TEST_SUITE_END()
36

```

QUESTION 3 - EXÉCUTION DU PLAN DE TESTS ET CORRECTIONS

PS C:\Users\23006020\Documents\TP4> ./make unit_test

*** No errors detected

Dans notre fichier excel PTU-Userinterface_c.xls, tous les Test Case Results sont ok (pass).

90		19	LDRA_uint32_t Userinterface_parse(const LDRA_char_t aChar)
91			{
92			LDRA_uint32_t menu;
93	► 4/4	19	if ((aChar >= '0') && (aChar <= '9'))
94			{
95		2	Cashregister_key((const LDRA_uint32_t)aChar - (const LDRA_uint32_t)'0');
96		2	menu = 0u;
97			}
98			else
99			{
100	► 14/14	17	switch (aChar)
101			{

Toutes les lignes de la fonction Userinterface_parse sont utilisées, comme on peut le voir dans les ronds rouges : 4/4 et 14/14.

EXERCICE 3 – TESTS UNITAIRES - SUITE

```
1 2
2  #define BOOST_TEST_MODULE MOQL
3  #include <boost/test/included/unit_test.hpp>
4
5  // Module sous test
6  #include "Userinterface.h"
7  #include "Specialoffer.h"
8  // #include "Productdatabase.h"
9
10 // Test-suite
11 BOOST_AUTO_TEST_SUITE( UT_Userinterface_parse )
12
13 // Test-case : valeurs normales du menu
14 BOOST_AUTO_TEST_CASE( menu_values )
15 {
16     BOOST_TEST( Userinterface_parse('b') == 1u );
17     BOOST_TEST( Userinterface_parse('n') == 2u );
18     BOOST_TEST( Userinterface_parse('/') == 14u );
19     BOOST_TEST( Userinterface_parse('\n') == 13u );
20     BOOST_TEST( Userinterface_parse('q') == 12u );
21     BOOST_TEST( Userinterface_parse('r') == 11u );
22     BOOST_TEST( Userinterface_parse('s') == 10u );
23     BOOST_TEST( Userinterface_parse('e') == 9u );
24     BOOST_TEST( Userinterface_parse('c') == 8u );
25     BOOST_TEST( Userinterface_parse('w') == 7u );
26     BOOST_TEST( Userinterface_parse('g') == 6u );
27     BOOST_TEST( Userinterface_parse('p') == 5u );
28     BOOST_TEST( Userinterface_parse('k') == 4u );
29     BOOST_TEST( Userinterface_parse('l') == 3u );
30     BOOST_TEST( Userinterface_parse('n') == 2u );
31     BOOST_TEST( Userinterface_parse('b') == 1u );
32     BOOST_TEST( Userinterface_parse('9') == 0u );
33     BOOST_TEST( Userinterface_parse(':') == 14u );
34     BOOST_TEST( Userinterface_parse('0') == 0u );
35 }
36
37 BOOST_AUTO_TEST_SUITE_END()
38
39 BOOST_AUTO_TEST_SUITE( UT_Specialoffer_getPrice )
40     BOOST_AUTO_TEST_CASE( menu_values )
41     {
42         BOOST_TEST( Specialoffer_getPrice(2, 5, NO_OFFER, '') == 10u );
43         BOOST_TEST( Specialoffer_getPrice(2, 5, BUY_ONE_GET_ONE_FREE, '') == 5u );
44         BOOST_TEST( Specialoffer_getPrice(2, 5, TEN_PERCENT_OFF, '') == 9u );
45         BOOST_TEST( Specialoffer_getPrice(1, 50, THREE_FOR_ONE_EURO, '') == 1u );
46     }
47
48
49 BOOST_AUTO_TEST_SUITE_END()
```

Nous avons tenté de modifier le fichier test.cpp pour ajouter des cas de test pour la fonction Specialoffer_getPrice dans le fichier Specialoffer.c. Malgré nos nombreuses tentatives, nous n'avons pas réussi à obtenir un résultat satisfaisant :

```
PS C:\Users\23006020\Documents\TP4> ./make unit_test
C:\Users\23006020\Documents\TP4>mingw32-make -f Makefile.mak unit_test
g++ -O0 -g3 -W -I. -MMD -MP --coverage -fprofile-coverage -fprofile-arcs -w -fpermissive -I"C:\MOQL\boost_1_78_0" Cashregister.c Productdatabase.c Specialoffer.c Userinterface.c Test.cpp -o test_cash.exe
Test.cpp:42:1: error: empty character constant
    BOOST_TEST( Specialoffer_getPrice(2, 5, NO_OFFER, '') == 10u );
    ^
Test.cpp:43:1: error: empty character constant
    BOOST_TEST( Specialoffer_getPrice(2, 5, BUY_ONE_GET_ONE_FREE, '') == 5u );
    ^
Test.cpp:44:1: error: empty character constant
    BOOST_TEST( Specialoffer_getPrice(2, 5, TEN_PERCENT_OFF, '') == 9u );
    ^
Test.cpp:45:1: error: empty character constant
    BOOST_TEST( Specialoffer_getPrice(1, 50, THREE_FOR_ONE_EURO, '') == 1u );
    ^
Makefile.mak:47: recipe for target 'test_cash.exe' failed
mingw32-make: *** [test_cash.exe] Error 1
```

Nous n'avons donc pas pu avoir une couverture de test de 100%. Avec du temps supplémentaire, nous pouvons tenter de débayer avec CPPCheck.