Text

Description automatically generated with medium confidence

LOG3430 - Méthodes de test validation du logiciel

Hiver 2022

Laboratoire 4

TESTS OO - MADUM

Soumis par :

1956611 – Bryan Junior Ngatshou

1935022– Dat Phi Tran

2016729– Tristan Dea

27 mars 2022

1. **Conception de la matrice MaDUM**

La matrice MaDUM est une matrice de n \* m dont n est le nombre de variables dans la classe et m et le nombre de de méthodes (fonctions) que la classe possède.

Étant donné que la Classe CRUD du fichier crud.py possède environ 20 fonctions, nous avons séparés cette matrice en 4 tableaux à cause de sa grande taille ces tableaux-là sont les suivants :

Tableau 1 : matrice MaDUM pour les fonctions \_init\_, get-new\_user\_id, convert\_to\_unix et add\_new\_user

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fonctions  Variable | \_\_init\_\_ | get\_new\_user\_id | get\_new\_group\_id | Convert\_to\_unix | add\_new\_user |
| users\_data | C | O |  |  | T |
| groups\_data | C |  | O |  |  |
| users\_lookup | C |  |  |  | T |
| Groups\_lookup | C |  |  |  |  |

* La fonction init construis toutes les variables
* La fonction get\_new\_user\_id utilise users\_data mais ne le retourne pas ni ne le transforme pas.
* get\_new\_group\_id utilise groups\_data mais ne le retourne pas ni ne le transforme pas.
* convert\_to\_unix n’a aucun effet sur les variables
* add\_new\_user transforme users\_data et users\_lookup.

Tableau 2: matrice MaDUM pour les fonctions add\_new\_group, read\_users\_file, read\_group\_files,get\_user\_data et get\_groups\_data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fonctions  Variable | add\_new\_group | read\_users\_file | read\_group s\_file | get\_user\_data | get\_groups\_data |
| users\_data | O |  |  | R |  |
| groups\_data | T |  |  |  | R |
| users\_lookup |  |  |  |  |  |
| Groups\_lookup | T |  |  |  |  |

* La fonction add\_new\_group tranforme(modifie) les variables groups\_data et groups\_lookup et utilise la variable users\_data sans ne la transformer ni la retourner
* La fonction read\_users\_file n’a aucun effet sur les variables
* La fonction read\_groups\_file n’a aucun effet sur les variables
* get\_user\_data retourne la variable users\_data
* get\_groups\_data retourne la variable groups\_data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fonctions  Variable | get\_user \_id | get\_group\_id | modify\_users \_file | modify\_groups  \_file | update\_ users |
| users\_data |  |  |  |  | T |
| groups\_data |  |  |  |  |  |
| users\_lookup | R |  |  |  | T |
| Groups\_lookup |  | R |  |  | O |

* La fonction get\_user\_id retourne la variable users\_lookup
* La fonction get\_group\_id retourne la variable goups\_lookup
* La fonction modify\_users\_file n’a aucun effet sur les variables
* La fonction modify\_groups\_file retourne la variable users\_data
* La function update\_users transforme users\_data, users\_lookup et utilise groups\_lookup sans ne la transformer ni la retourner

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fonctions  Variable | update\_groups | remove  \_user | remove\_user\_group | remove\_group | remove\_group\_member |
| users\_data | O | T | T |  |  |
| groups\_data | T |  |  | T | T |
| users\_lookup | O | T |  |  |  |
| Groups\_lookup | T |  |  | T |  |

* La fonction update\_groups transforme groups\_data, groups\_lookup, et utilise users\_data et users\_lookup sans ne les transformer ni les retourner
* La fonction remove\_user transforme users\_data, users\_lookup
* La fonction remove\_user\_ group transforme users\_data
* La fonction remove\_group transforme groups\_data et groups\_lookup
* La fonction remove\_group\_member tranforme groups\_data

1. **Séquence d’appels faits pour la variable groups\_data selon la méthode MaDUM**
2. Les constructeurs (C)

Une seule séquence pour le constructeur :

d1: test\_constructor = {\_init\_ -> get\_groups\_data}

1. Les rapporteurs (R)

Une seule séquence pour les rapporteurs :

d1: test\_get\_groups\_data = {\_init\_ -> get\_groups\_data}

1. Les tranformatteurs (T)

La classe CRUD possède 4 transformateurs pour groups\_data dont ”add new group”, ”update groups”, ”remove group”, ”remove group member”, donc nous auront au total 24 combinaisons de tests

en prenant A pour add\_new\_group , U pour update\_groups, R pour remove \_group et M pour remover\_group\_member, on obtient avec l’aide de l’outil en ligne Free-Online\_Calculator\_Use le tableau des différents combinaisons de séquences suivant :

Table

Description automatically generated

Figure2.c.1 séquences pour les transformateurs de l’attribut groups\_data obtenues grâce au site dont le lien est <https://www.free-online-calculator-use.com/combination-calculator.html>

Selon la figure2. c.1 , nous pouvons donc avoir comme séquences les cas suivants :

* d1 : test\_add\_new\_group\_update\_groups\_remove\_group\_remove\_group\_member =

\_\_init\_\_ -> add\_new\_group -> update\_groups -> remove\_group -> remove\_group\_member -> get\_groups\_data

* d2 : test\_update\_groups\_add\_new\_group\_remove\_group\_remove\_group\_member =

\_\_init\_\_ -> update\_groups -> add\_new\_group -> remove\_group -> remove\_group\_member -> get\_groups\_data

* d3: test\_remove\_group\_add\_new\_group\_update\_groups\_remove\_group\_member =

{\_\_init\_\_-> remove\_group -> add\_new\_group -> update\_groups -> remove\_group\_member-> get\_groups\_data}

* d4: test\_add\_new\_group\_remove\_group\_update\_groups\_remove\_group\_member =

{\_\_init\_\_ -> add\_new\_group -> remove\_group ->update\_groups -> remove\_group\_member -> get\_groups\_data }

continuer jusqu`à d24

1. Les autres (O)

Une seule séquence pour la fonction get\_new\_group\_id

d1: test\_get\_new\_group\_id = {\_init\_ -> get\_new\_group\_id}