# Étapes précédant les tests

**Ajout aux classes**

**Dans ce travail pratique, nous devons rajouter un attribut type à la classe *Graph* ainsi que les méthodes *getter* et *setter*. Ensuite, nous avons utilisé la méthode *setter* dans la classe *GraphGenerator* pour le type selon la méthode. Subséquemment, nous avons ajouté une méthode à *BipartiteXExtended*. Notre choix s’est arrêté à getPartitonBlack(), qui retourne une des deux partitions constituant le graph bipartie. La couleur Black réfère au code de couleur utilisé dans le code précédemment**

**Tests à refaire**

**Étant donné que le code de différentes classes a été modifiées, nous devons nous interroger sur les tests à refaire. Pour commencer, nous nous référons au tableau suivant réalisé lors du dernier travail pratique:**

|  |  |
| --- | --- |
| **Classe étudiée** | **CFW de la classe étudiée** |
| BipartiteXExtended |  |
| BipartiteX | BipartiteXExtended |
| Graph | BipartiteXExtended  BipartiteX |
| In | BipartiteXExtended  BipartiteX  Graph |
| Queue | BipartiteXExtended  BipartiteX |
| Stack | BipartiteXExtended  BipartiteX  Graph |
| Bag | BipartiteXExtended  BipartiteX  Graph |

Tableau 1: CFW des classes étudiées

Trois constats concernant le tableau 1 doivent être mentionnés :

* *GraphGenerator* ne fait pas partie des classes concernées.
* *Graph* aura un impact sur deux classes : *BipartiteX* et *BipartiteXExtended*.
* Aucune classe sera affectée par la modification de *BipartiteXExtended*.

La prochaine étape consiste à réaliser les tests.

# Tests

**Critère d’adéquation**

Nous retenons le même cirtère d’adéquation que le dernier travail pratique, soit la couverture des branches. Ce choix fut arrêté à cause de sa simplicité. Nous utilisons JaCoCo pour l’évaluer. Il est à noter que les fonctions main sont ignorées.

**Queue & Bag & In & Stack**

Comme mentionné lors de l’analyse du tableau 1, ces classes ne sont pas affectées et ne nécessitent pas de nouveaux tests.

**Graph**

Les anciens tests furent modifiés pour avoir une vérification que l’attribut type reste *NULL* lors des tests. Pour la méthode *setter*, un test additionnel a été nécessaire. Le test consiste à générer un graphe avec un type indéfini pour ensuite le définir en s’assurant que la valeur est bien celle désirée.

**bipartiteX**

Les anciens tests furent modifiés pour avoir une vérification que l’attribut type reste *NULL* lors des tests.

**bipartiteXExtended**

Les anciens tests furent modifiés pour avoir une vérification que l’attribut type reste *NULL* lors des tests. De plus, la nouvelle méthode doit être testée. Nous l’avons à travers tous les tests à regardant que les partitions correspondaient à nos attentes. En effet, en créant les arcs nous-mêmes, nous connaissons les partitions.

# Questions