

Correction de l'examen final.**Génie Logiciel – LMD S4.****Cours : (5pts)**

- A- *Vérification* : s'assurer du bon fonctionnement du produit (1,5p)
Validation : s'assurer que le produit réponde au besoin selon le cahier des charges.
- B- Les types des diagrammes pour la modélisation UML (4 parmi 9) : (1,5p)
 Les diagrammes d'objets, de collaboration, de classes, de cas d'utilisation, de séquence, d'états-transitions, d'activités, de composants, de déploiement.
- C- *JRE* : Java Runtime Environment ; c'est la bibliothèque servant de machine virtuelle (2p)
 pour l'exécution des programmes Java.
JDK : Java Development Kit ; c'est l'ensemble des bibliothèques servant à la compilation et l'exécution des programmes Java.

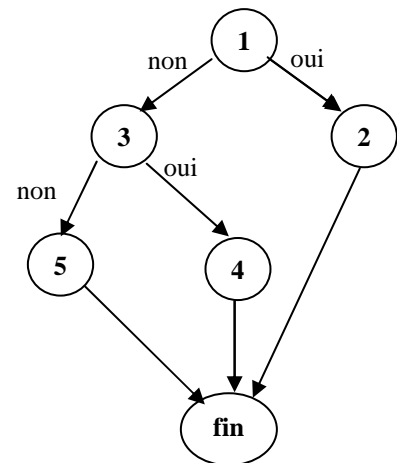
Exercice 01 : (6pts)

1- Graphe de contrôle : (2,5p)

2- Vérification par le jeu de tests T1, T2 (2p)

| Test T1(a=0,b=0) | | | | |
|------------------|---|---|---|-------|
| Instr. | a | b | x | Suiv. |
| 1 | 0 | 0 | ? | 3 |
| 3 | 0 | 0 | ? | 5 |
| 5 | 0 | 0 | 0 | fin |
| Chemin : 1, 3, 5 | | | | |

| Test T2(a=1,b=0) | | | | |
|------------------|---|---|---|-------|
| Instr. | a | b | x | Suiv. |
| 1 | 1 | 0 | ? | 2 |
| 2 | 1 | 0 | 0 | fin |
| Chemin : 1, 2 | | | | |



- 3- Un jeu de tests satisfaisant doit couvrir tous les chemins. Selon le graphe de contrôle, nous avons trois différents chemins. Le jeu de test précédent en couvre deux seulement et nous aurons besoin d'un troisième test pour couvrir le chemin (1, 3, 4). (0,5p)

| Test T3(a=-1,b=0) | | | | |
|-------------------|----|---|-------|-------|
| Instr. | a | b | x | Suiv. |
| 1 | -1 | 0 | ? | 3 |
| 3 | -1 | 0 | ? | 4 |
| 4 | -1 | 0 | Div/0 | ? |
| Chemin : 1, 3, 4 | | | | |

Nous détectons ainsi une erreur de division par 0 non contrôlée.

(1p)

Exercice 02 : (9pts)

1- Les classes : *Crayon*, *CrayonCouleur*

(1p)

Les relations : Héritage de *CrayonCouleur* à partir de *Crayon*

(0,5p)

2- Le diagramme de classes :

(2,5p)



3- La déclaration dans la ligne (1) est appelée *constructeur*

(0,5p)

4- Les re-déclarations dans (2) et (3) sont autorisées puisque les paramètres dans chaque prototype sont différents et que le langage Java permet la surcharge des méthodes (polymorphisme)

(1p)

5- L'instruction (4) n'est pas permise puisque l'attribut *diametre* est déclaré *private* dans une autre classe. il faut donc le déclarer *protected* dans la classe mère *Crayon*.

L'instruction (5), par contre, est tout à fait légale puisque *couleur* est déclarée dans la même classe *CrayonCouleur*.

(1,5p)

6- L'affichage direct d'un objet fait appel à la fonction *toString()* définie dans la classe mère (*Object*) et, éventuellement, surchargée dans les autres classes. Donc nous appliquons le résultat de cette fonction pour chaque objet et nous aurons (après la correction précédente) :

(2p)

```
Crayon : 5
CrayonCouleur : 5, gris
CrayonCouleur : 7, vert
CrayonCouleur : 5, bleu
```