Université d'Orléans
Master MIAGE M1 2020-2021
Test Qualité

Feuille de TD3 "Mock et TDD"

Le but de cette feuille est de s'initier aux mocks (bouchons) et le test driven development (TDD).

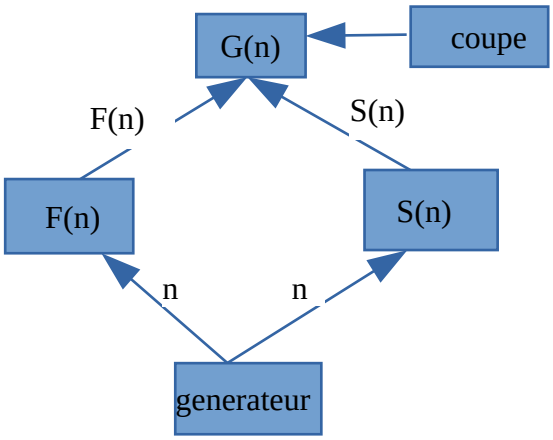
Exercice 1

Téléchargez les classes *categorie* et *employe*.
Notons *an* le nombre d'années passé dans l'entreprise, si (0<=an<=10) alors la valeur de la catégorie (valCategorie) est 1, sinon si (11<=an<=20) alors la valeur de la catégorie est 2, sinon la valeur de la catégorie est 3. Si la catégorie d'un employé est 1, son salaire effectif est son salaire de base, sinon si sa catégorie est 2 son salaire est 1.1 fois son salaire de base, sinon si sa catégorie est 3 son salaire effectif est 1.2 fois son salaire de base.

1. En attendant que votre camarade à qui on a confié le codage de la fonction *valCategorie* monte en compétence en Java, on vous demande de coder les tests pour *salaire* en mockant *categorie*.
2. Codez la fonction *salaire* validant les tests
3. Codez les tests unitaires pour *valCategorie* de la classe *categorie*
4. Codez la fonction *valCategorie* validant les tests

Exercice 2

Un camarade a effacé par inattention les programmes Java qu'il utilisait sur son système 32 bit ci-dessous :



Description fonctionnelle du système

- Le générateur génère un nombre entier aléatoire *n* entre 0 et 20 l'envoie simultanément vers les systèmes F et G.
- Le système F est une "suite de fibonacci". Il va calculer le *n*-ieme terme de Fibonacci *F(n)* puis l'envoie vers G.
- Le système S est une "suite de Syracuse" de permier terme 1. Il va calculer le *n*-ieme terme de la suite de Syracuse *S(n)* puis l'envoie vers G.
- Le système G reçoit puis traite F(n) et S(n) selon la valeur de la "coupe" (1 ou 40) qu'on lui a *donnée* et selon l'algo suivant : si la somme *so* de F(n) et G(n) est strictement plus petite que la valeur de la *coupe* alors G(n) vaut *so* sinon G(n) faut la plus grande valeur entre F(n) et S(n).

Vous devez donc (re-)développer ce système.

1. Le temps de réviser vos cours de maths pour retrouver les définitions des suites de Fibonacci et de Syracuse, vous pouvez commencer à coder G en Java selon l'approche TDD en utilisant les valeurs que votre camarade a relevées et enregistrées dans les tableaux ci-après. Utilisez des mokcs (Easymock).

G avec coupe=1			
n	F(n)	S(n)	G(n)
0	1	1	1
1	1	4	4
2	2	2	2
10	89	4	89
19	6765	4	6765
20	10946	2	10946

G avec coupe=40			
n	F(n)	S(n)	G(n)
0	1	1	2
1	1	4	5
2	2	2	4
10	89	4	89
19	6765	4	6765
20	10946	2	10946

2. Codez la fonction G validant le test.
3. Codez les tests unitaires (TU) pour S
4. Codez la fonction S validant le test
5. Codez les TU pour F
6. Codez la fonction F validant le test
7. Le camarade voudrait s'*acheter* un générateur de nombre aléatoire plus "puissant" qui peut générer un nombre entre 0 et 100. Qu'en dites vous ? Le cas écheant, mettez à jour vos tests et programmes.