TP7 - The Unit Question

Objectifs

- Suivre un kata guide base sur le repo `the-unit-question`
- Progresser pas a pas dans une base de code existante via les branches Git
- Appliquer les pratiques TDD vues en seances precedentes sur un projet concret
- Lire et comprendre les tests existants avant d'en ecrire de nouveaux

Prerequis

- Git installe et configure
- Node.js et pnpm installes
- Avoir suivi les seances 1 a 6 (TDD, doubles de test, mutation testing)

Mise en place

Clonez le depot du kata :

```
git clone git@github.com:edouard-mangel/the-unit-question.git
cd the-unit-question
```

Installez les dependances :

```
pnpm install
```

Verifiez que les tests passent :

```
pnpm test
```

Explorez les branches disponibles — chaque branche correspond a une etape du kata :

```
git branch -a
```

Deroulement du kata

Le repo est structure comme une serie d'etapes numerotees. Chaque branche Git represente un etat du code a un moment donne du kata.

Convention des branches

```
main           ← point de depart (ou etape 0)
step/01-...    ← premiere etape
step/02-...    ← deuxieme etape
...
solution       ← solution complete (a consulter en dernier recours)
```

Navigation entre les etapes

Pour passer a l'etape suivante :

```
git checkout step/02-...
pnpm install   # si les dependances ont change
pnpm test
```

Lisez toujours le `README.md` de la branche courante avant de commencer a coder.

Regle d'or

> Ne consultez pas la branche `solution` avant d'avoir veritablement essaye de resoudre l'etape.

Exercice 1 — Prise en main (15 min)

Objectif

Comprendre la structure du projet et executer les tests pour la premiere fois.

Instructions

1. Clonez le depot et placez-vous sur `main`
2. Lisez le `README.md` a la racine
3. Lancez les tests et observez le resultat
4. Identifiez les fichiers sources et les fichiers de test

Verification

- [ ] Le depot est clone localement
- [ ] `pnpm test` s'execute sans erreur d'installation
- [ ] Vous avez identifie le fichier source principal et son fichier de test associe
- [ ] Vous avez lu le README et compris l'objectif global du kata

Questions

1. Combien de tests passent sur `main` ? Combien echouent ?
2. Quelle est la responsabilite principale de la classe/fonction centrale du projet ?
3. Quels doublures ou dependances identifiez-vous dans le code ?

Exercice 2 — Suivi pas a pas (45 min)

Objectif

Progresser a travers les etapes du kata en ecrivant le code manquant avant de passer a l'etape suivante.

Instructions

Pour chaque etape numerotee :

1. Lisez les tests existants sur la branche — ils decrivent le comportement attendu
2. Implementez le code manquant pour faire passer les tests au rouge
3. Faites passer les tests au vert avec le minimum de code necessaire
4. Refactorisez si necessaire (cycle Rouge → Vert → Refactor)
5. Passez a la branche suivante uniquement quand tous les tests sont au vert

Contraintes

- Ne modifier aucun fichier de test existant
- Chaque commit doit correspondre a une etape terminee
- Le message de commit doit decrire ce qui a ete implemente

Format de commit suggere

```
step 01: implement <nom de la fonctionnalite>
step 02: add validation for <cas limite>
```

Verification

- [ ] Vous avez progresse sur au moins 3 etapes
- [ ] Chaque etape se termine avec tous les tests au vert
- [ ] Vos commits sont clairs et correspondent aux etapes
- [ ] Vous n'avez pas modifie les tests existants

Exercice 3 — Analyse critique (20 min)

Objectif

Prendre du recul sur le code produit et les choix de design mis en evidence par le kata.

Instructions

Apres avoir termine au moins 3 etapes, repondez aux questions suivantes.

Questions de reflexion

1. Quels patterns de test avez-vous reconnus ?

Identifiez au moins deux patterns parmi : Arrange-Act-Assert, Given-When-Then, test de borne, test d'erreur, test de comportement.

2. Y a-t-il des doublures de test dans ce projet ?

Si oui : sont-elles manuelles ou generees (`vi.fn()` , `jest.fn()`) ? Que revelent-elles sur les dependances du code ?

3. Le design du code a-t-il ete influence par les tests ?

Donnez un exemple concret d'une decision d'implementation qui a ete dictee par la testabilite.

4. Que changeriez-vous dans la suite de tests ?

Identifiez un test manquant ou un test faible. Proposez un cas supplementaire qui renforcerait la suite.

Livrable

Un document texte (`.md` ou `.txt`) avec vos reponses, a remettre avec le depot Git.

Rendu

Ce qu'il faut remettre

Element	Description
Lien vers votre fork	Fork du depot sur GitHub/GitLab avec vos commits
Fichier de reflexion	Reponses a l'exercice 3

Comment forker et remettre votre travail

```
# 1. Forkez le depot sur GitHub (via l'interface web)

# 2. Ajoutez votre fork comme remote
git remote add myfork git@github.com:<votre-login>/the-unit-question.git

# 3. Poussez vos branches
git push myfork main
git push myfork --all   # pour toutes les branches de travail
```

Transmettez l'URL de votre fork via le formulaire de rendu habituel.

Bareme indicatif

Critere	Points
Depot clone et tests passes sur `main`	2 pts
Progression sur au moins 3 etapes (tests au vert)	6 pts
Commits propres et lisibles	2 pts
Analyse critique argumentee (exercice 3)	5 pts
Total	**15 pts**