2022-2023 - R4-02 Qualité de développement - CC1 Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom : Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher. Question 1 🌲 Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies Les TA peuvent contenir des bugs. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. TDD - Cochez les assertions vraies Question 2 🌲 Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue. Le refactoring consiste à remanier le code d'un programme. En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe. Le TDD améliore la conception d'un programme.

En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.

Tests unitaires vs tests d'intégration - Cochez les assertions vraies

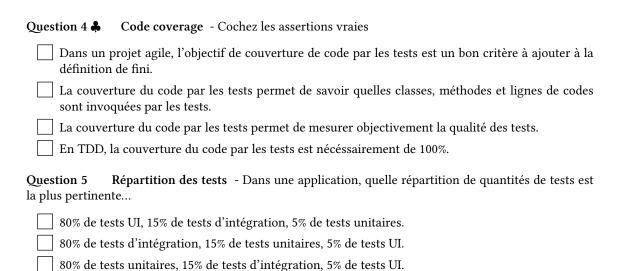
Les tests unitaires consomment en général moins de ressources que les tests d'intégration.

Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.

Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.

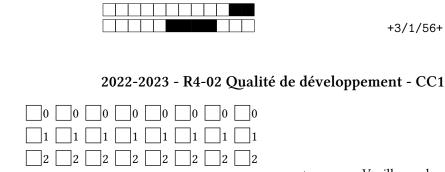
Question 3 🌲





\square_2 \square_2 \square_2 \square_2 \square_2 \square_2 \square_2	
	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	uans la case el dessous.
	Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est auto Le symbole "trèfle" sur une question indique qu'il pe Les réponses doivent être fournies uniquement sur le prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une réponse re-dessiner la case à cocher.	orisé. out être attendu plus d'une réponse correcte. s feuilles d'énoncé. Aucune autre réponse ne sera entièrement coloriées en noires pour être prises
Question 1 & Code coverage - Cochez les assertions	s vraies
Dans un projet agile, l'objectif de couverture de c	ode par les tests est un bon critère à ajouter à la
définition de fini.	
La couverture du code par les tests permet de mes	
En TDD, la couverture du code par les tests est né	
La couverture du code par les tests permet de sav sont invoquées par les tests.	voir quelles classes, méthodes et lignes de codes
Question 2 🌲 Tests unitaires vs tests d'intégration	- Cochez les assertions vraies
Les tests d'intégration vérifient le fonctionnement	de "vrais" objets en interactions.
Les tests unitaires créent toujours les vrais collabo	orateurs pour tester un objet.
Les tests unitaires consomment en général moins	de ressources que les tests d'intégration.
Les tests d'intégration s'exécutent en général plus	rapidement que les tests unitaires.
Question 3 Répartition des tests - Dans une applic la plus pertinente	ation, quelle répartition de quantités de tests est
80% de tests d'intégration, 15% de tests unitaires, 5	5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de t	ests unitaires.
80% de tests unitaires, 15% de tests d'intégration, 5	5% de tests UI.
Question 4 🌲 Bénéfices et limites des tests automat	isés (TA) - Cochez les assertions vraies
Les TA permettent d'effectuer des changements de	ans le code en confiance.
Les TA permettent de montrer qu'un programme	fonctionne comme attendu.
Les TA permettent de limiter le nombre de bugs.	
Les TA peuvent contenir des bugs.	

Question 5 🌲	TDD - Cochez les assertions vraies
En TDD, on	code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
En TDD, dè	s qu'un test passe, on passe à l'écriture d'un nouveau test.
Le TDD am	éliore la conception d'un programme.
Le refactori	ng est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue
Le refactori	ng consiste à remanier le code d'un programme



Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.

Nom e	prénom :	

Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points.

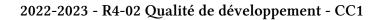
Aucun document ni dispositif électronique n'est autorisé.

Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte.

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.

Question 1 ♣ TDD - Cochez les assertions vraies
Le TDD améliore la conception d'un programme.
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Le refactoring consiste à remanier le code d'un programme.
Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.
En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
Question 2 - Cochez les assertions vraies
Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
Les tests unitaires consomment en général moins de ressources que les tests d'intégration.
Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
Question 3 & Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les TA permettent d'effectuer des changements dans le code en confiance.
Les TA permettent de montrer qu'un programme fonctionne comme attendu.
Les TA peuvent contenir des bugs.
Les TA permettent de limiter le nombre de bugs.

Questio	n 4 🌲 Code coverage - Cochez les assertions vraies
	ans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à l Finition de fini.
La	a couverture du code par les tests permet de mesurer objectivement la qualité des tests.
Er.	n TDD, la couverture du code par les tests est nécéssairement de 100%.
_	a couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de code ent invoquées par les tests.
Questio la plus p	n 5 Répartition des tests - Dans une application, quelle répartition de quantités de tests es pertinente
80	% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
80	% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.
80	% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.



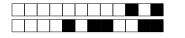
0 0	
prise en compte dans la notation. Les cases doivent	autorisé.
Question 1 ♣ Tests unitaires vs tests d'intégration Les tests unitaires consomment en général mon Les tests d'intégration vérifient le fonctionnem Les tests unitaires créent toujours les vrais col Les tests d'intégration s'exécutent en général production de la company de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests unitaires vs tests d'intégration de la colonnem Les tests d'intégration vérifient le fonctionnem Les tests unitaires créent toujours les vrais colonnem Les tests d'intégration s'exécutent en général montion de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration s'exécutent en général production de la colonnem Les tests d'intégration de	oins de ressources que les tests d'intégration. nent de "vrais" objets en interactions. llaborateurs pour tester un objet.
Question 2 . TDD - Cochez les assertions vraie	S
☐ En TDD, on code tous les tests d'une classe av ☐ Le TDD améliore la conception d'un programm ☐ En TDD, dès qu'un test passe, on passe à l'écri ☐ Le refactoring consiste à remanier le code d'un	me. iture d'un nouveau test.
Le refactoring est une étape du TDD qu'un dé	veloppeur doit entreprendre après qu'un test échoue.
	oplication, quelle répartition de quantités de tests est
80% de tests d'intégration, 15% de tests unitair	res, 5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5%	de tests unitaires.
80% de tests unitaires, 15% de tests d'intégration	



Code coverage - Cochez les assertions vraies
projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la de fini.
rture du code par les tests permet de savoir quelles classes, méthodes et lignes de code quées par les tests.
la couverture du code par les tests est nécéssairement de 100%.
ture du code par les tests permet de mesurer objectivement la qualité des tests.
Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
ermettent de limiter le nombre de bugs.
ermettent de montrer qu'un programme fonctionne comme attendu.
ermettent d'effectuer des changements dans le code en confiance.
euvent contenir des bugs.



\square 2	
	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	dans la case et dessous.
	Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps). I Aucun document ni dispositif électronique n'est autor Le symbole "trèfle" sur une question indique qu'il peu Les réponses doivent être fournies uniquement sur les prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une réponse, re-dessiner la case à cocher.	risé. It être attendu plus d'une réponse correcte. feuilles d'énoncé. Aucune autre réponse ne sera entièrement coloriées en noires pour être prises
Question 1 4 Tests unitaires vs tests d'intégration -	Cochez les assertions vraies
Les tests d'intégration vérifient le fonctionnement	de "vrais" objets en interactions.
Les tests unitaires créent toujours les vrais collabor	rateurs pour tester un objet.
Les tests unitaires consomment en général moins d	e ressources que les tests d'intégration.
Les tests d'intégration s'exécutent en général plus l	rapidement que les tests unitaires.
Question 2 ♣ TDD - Cochez les assertions vraies	
Le TDD améliore la conception d'un programme.	
En TDD, dès qu'un test passe, on passe à l'écriture	d'un nouveau test.
Le refactoring consiste à remanier le code d'un pro	gramme.
En TDD, on code tous les tests d'une classe avant d	le coder toutes les méthodes de la classe.
Le refactoring est une étape du TDD qu'un dévelop	peur doit entreprendre après qu'un test échoue.
Question 3 Répartition des tests - Dans une applica la plus pertinente	tion, quelle répartition de quantités de tests est
80% de tests d'intégration, 15% de tests unitaires, 59	% de tests UI.
80% de tests unitaires, 15% de tests d'intégration, 59	% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de te	sts unitaires.



Question 4 & Code coverage - Cochez les assertions vraies
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de code sont invoquées par les tests.
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à l définition de fini.
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
En TDD, la couverture du code par les tests est nécéssairement de 100%.
Question 5 & Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les TA permettent de limiter le nombre de bugs.
Les TA permettent d'effectuer des changements dans le code en confiance.
Les TA permettent de montrer qu'un programme fonctionne comme attendu.
Les TA peuvent contenir des bugs.



	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	dans la case ci-dessous.
	Nom et prénom :
Les réponses doivent être fournies uniquement prise en compte dans la notation. Les cases doiver	- · · · · · · · · · · · · · · · · · · ·
Question 1 & TDD - Cochez les assertions vra	nies
Le refactoring est une étape du TDD qu'un o	développeur doit entreprendre après qu'un test échoue.
	avant de coder toutes les méthodes de la classe.
En TDD, dès qu'un test passe, on passe à l'é	criture d'un nouveau test.
Le TDD améliore la conception d'un progra	mme.
Le refactoring consiste à remanier le code d	'un programme.
Question 2 - Tests unitaires vs tests d'intégra	ation - Cochez les assertions vraies
Les tests unitaires créent toujours les vrais c	collaborateurs pour tester un objet.
	noins de ressources que les tests d'intégration.
Les tests d'intégration vérifient le fonctionn	
Les tests d'intégration s'exécutent en généra	·
_	tomatisés (TA) - Cochez les assertions vraies
Les TA permettent d'effectuer des changeme	ents dans le code en confiance.
Les TA permettent de limiter le nombre de l	
Les TA peuvent contenir des bugs.	
Les TA permettent de montrer qu'un progra	ımme fonctionne comme attendu.

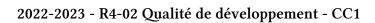


Question 4 & Code coverage - Cochez les assertions vraies
En TDD, la couverture du code par les tests est nécéssairement de 100%.
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
Question 5 Répartition des tests - Dans une application, quelle répartition de quantités de tests es la plus pertinente
80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
80% de tests III 15% de tests d'intégration 5% de tests unitaires

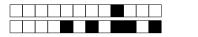
+7/1/48+ 2022-2023 - R4-02 Qualité de développement - CC1 $\bigcirc 0 \bigcirc 0$ Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom : Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher. Question 1 🐥 Tests unitaires vs tests d'intégration - Cochez les assertions vraies Les tests unitaires consomment en général moins de ressources que les tests d'intégration. Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires. Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet. Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

Ouestion 2 🌲 TDD - Cochez les assertions vraies Le TDD améliore la conception d'un programme. En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe. Le refactoring consiste à remanier le code d'un programme. Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue. En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test. Question 3 🌲 Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies Les TA peuvent contenir des bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de limiter le nombre de bugs. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Répartition des tests - Dans une application, quelle répartition de quantités de tests est Question 4 la plus pertinente... 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI. 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires. 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.

Question 5 ૈ	Code coverage - Cochez les assertions vraies
Dans un p	projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la de fini.
En TDD, l	a couverture du code par les tests est nécéssairement de 100%.
La couver	ture du code par les tests permet de mesurer objectivement la qualité des tests.
	ture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes quées par les tests.

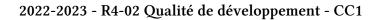


\square_2 \square_2 \square_2 \square_2 \square_2 \square_2 \square_2	
	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	ualis la case ci-dessous.
	Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est auto Le symbole "trèfle" sur une question indique qu'il per Les réponses doivent être fournies uniquement sur les prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une réponse re-dessiner la case à cocher.	orisé. ut être attendu plus d'une réponse correcte. s feuilles d'énoncé. Aucune autre réponse ne sera entièrement coloriées en noires pour être prises
Question 1 & TDD - Cochez les assertions vraies	
Le TDD améliore la conception d'un programme.	
En TDD, dès qu'un test passe, on passe à l'écriture	
En TDD, on code tous les tests d'une classe avant d	
Le refactoring consiste à remanier le code d'un pro	
Le refactoring est une étape du TDD qu'un dévelop	ppeur doit entreprendre après qu'un test échoue.
Question 2 4 Tests unitaires vs tests d'intégration	Cochez les assertions vraies
Les tests unitaires consomment en général moins d	de ressources que les tests d'intégration.
Les tests d'intégration vérifient le fonctionnement	de "vrais" objets en interactions.
Les tests d'intégration s'exécutent en général plus	rapidement que les tests unitaires.
Les tests unitaires créent toujours les vrais collabo	rateurs pour tester un objet.
Question 3 🌲 Bénéfices et limites des tests automati	sés (TA) - Cochez les assertions vraies
Les TA permettent d'effectuer des changements da	ns le code en confiance.
Les TA permettent de limiter le nombre de bugs.	
Les TA peuvent contenir des bugs.	
Les TA permettent de montrer qu'un programme f	onctionne comme attendu.
Question 4 Répartition des tests - Dans une applica la plus pertinente	ation, quelle répartition de quantités de tests est
80% de tests d'intégration, 15% de tests unitaires, 5	% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de te	
80% de tests unitaires, 15% de tests d'intégration, 5	% de tests UI.



Question 5 ૈ	Code coverage	- Cochez les	assertions	vraies

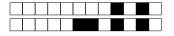
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
En TDD, la couverture du code par les tests est nécéssairement de 100%.
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini



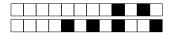
0 0	← Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom:
prise en compte dans la notation. Les cases doivent	autorisé.
Ouestion 1 ♣ Bénéfices et limites des tests auto Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bug Les TA permettent de montrer qu'un program Les TA permettent d'effectuer des changemen	me fonctionne comme attendu.
Question 2 ♣ Tests unitaires vs tests d'intégration Les tests d'intégration s'exécutent en général monte de la test unitaires consomment en général monte de la test unitaires créent toujours les vrais colonnes de la test d'intégration vérifient le fonctionnes de la test de la test d'intégration vérifient le fonctionnes de la test d'intégration vérifient le fonctionnes de la test de la test d'intégration vérifient le fonctionnes de la test de la test d'intégration vérifient le fonctionnes de la test de la test d'intégration vérifient le fonctionnes de la test de la test d'intégration vérifient le fonctionnes de la test de la test d'intégration vérifient le fonctionnes de la test	plus rapidement que les tests unitaires. sins de ressources que les tests d'intégration. llaborateurs pour tester un objet.
Question 3 - Code coverage - Cochez les assert	tions vraies
 La couverture du code par les tests permet de Dans un projet agile, l'objectif de couverture définition de fini. En TDD, la couverture du code par les tests es 	de code par les tests est un bon critère à ajouter à la
La couverture du code par les tests permet de sont invoquées par les tests.	e savoir quelles classes, méthodes et lignes de codes



Question 4 . TDD - Cochez les assertions vraies
Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue
Le refactoring consiste à remanier le code d'un programme.
En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Le TDD améliore la conception d'un programme.
Question 5 Répartition des tests - Dans une application, quelle répartition de quantités de tests es la plus pertinente
80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.
80% de tests d'intégration 15% de tests unitaires 5% de tests III



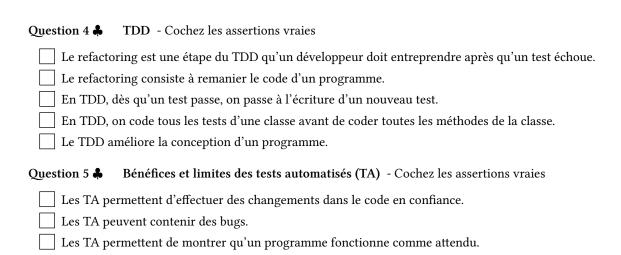
0 0	✓ Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est auto Le symbole "trèfle" sur une question indique qu'il per Les réponses doivent être fournies uniquement sur les prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une réponse re-dessiner la case à cocher.	risé. ut être attendu plus d'une réponse correcte. feuilles d'énoncé. Aucune autre réponse ne sera entièrement coloriées en noires pour être prises
Question 1 🌲 Bénéfices et limites des tests automati	sés (TA) - Cochez les assertions vraies
Les TA permettent de montrer qu'un programme f	onctionne comme attendu.
Les TA permettent d'effectuer des changements da	ns le code en confiance.
Les TA permettent de limiter le nombre de bugs.	
Les TA peuvent contenir des bugs.	
Question 2 Répartition des tests - Dans une applica la plus pertinente	ation, quelle répartition de quantités de tests est
80% de tests unitaires, 15% de tests d'intégration, 5	% de tests UI.
80% de tests d'intégration, 15% de tests unitaires, 5	% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de te	ests unitaires.
Question 3 ♣ TDD - Cochez les assertions vraies	
En TDD, dès qu'un test passe, on passe à l'écriture	d'un nouveau test.
Le refactoring est une étape du TDD qu'un dévelop	ppeur doit entreprendre après qu'un test échoue.
Le TDD améliore la conception d'un programme.	
En TDD, on code tous les tests d'une classe avant d	
Le refactoring consiste à remanier le code d'un pro	gramme.
Question 4 🌲 Tests unitaires vs tests d'intégration	
Les tests d'intégration s'exécutent en général plus	rapidement que les tests unitaires.
Les tests unitaires consomment en général moins d	
Les tests unitaires créent toujours les vrais collabo	
Les tests d'intégration vérifient le fonctionnement	de "vrais" objets en interactions.



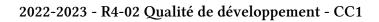
Question 5 ૈ	Code coverage	- Cochez les assertions vraies	,

La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
En TDD, la couverture du code par les tests est nécéssairement de 100%.
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la
définition de fini

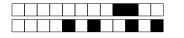
0 0	← Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temp Aucun document ni dispositif électronique n'est a Le symbole "trèfle" sur une question indique qu'i Les réponses doivent être fournies uniquement su prise en compte dans la notation. Les cases doivent é en compte dans la notation. Pour modifier une répo re-dessiner la case à cocher.	autorisé. I peut être attendu plus d'une réponse correcte. r les feuilles d'énoncé. Aucune autre réponse ne sera être entièrement coloriées en noires pour être prises
Question 1 Tests unitaires vs tests d'intégration Les tests unitaires créent toujours les vrais coll Les tests d'intégration s'exécutent en général pur les tests unitaires consomment en général mon les tests d'intégration vérifient le fonctionnement	aborateurs pour tester un objet. blus rapidement que les tests unitaires. ins de ressources que les tests d'intégration.
Question 2 Répartition des tests - Dans une appla plus pertinente	plication, quelle répartition de quantités de tests est
80% de tests unitaires, 15% de tests d'intégration	n, 5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% d	de tests unitaires.
80% de tests d'intégration, 15% de tests unitaire	es, 5% de tests UI.
Question 3 & Code coverage - Cochez les asserti	ions vraies
La couverture du code par les tests permet de sont invoquées par les tests.	savoir quelles classes, méthodes et lignes de codes
En TDD, la couverture du code par les tests est	nécéssairement de 100%.
Dans un projet agile, l'objectif de couverture définition de fini.	le code par les tests est un bon critère à ajouter à la
La couverture du code par les tests permet de r	nesurer objectivement la qualité des tests.



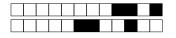
Les TA permettent de limiter le nombre de bugs.



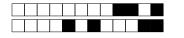
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
d'étudiant ci-contre, et écrire votre nom
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.
Question 1 & Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Question 1 & Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies Les TA peuvent contenir des bugs.
Les TA peuvent contenir des bugs.
Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs.
Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance.
 Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Question 2 ♣ Code coverage - Cochez les assertions vraies
Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de montrer qu'un programme fonctionne comme attendu.
 Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Question 2 ♣ Code coverage - Cochez les assertions vraies En TDD, la couverture du code par les tests est nécéssairement de 100%. La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes
 Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Question 2 ♣ Code coverage - Cochez les assertions vraies En TDD, la couverture du code par les tests est nécéssairement de 100%. La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la
 Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Question 2 ♣ Code coverage - Cochez les assertions vraies En TDD, la couverture du code par les tests est nécéssairement de 100%. La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
 Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Question 2 ♣ Code coverage - Cochez les assertions vraies En TDD, la couverture du code par les tests est nécéssairement de 100%. La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini. La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Question 2 ♣ Code coverage - Cochez les assertions vraies En TDD, la couverture du code par les tests est nécéssairement de 100%. La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini. La couverture du code par les tests permet de mesurer objectivement la qualité des tests. Question 3 ♣ Tests unitaires vs tests d'intégration - Cochez les assertions vraies
 Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Question 2 ♣ Code coverage - Cochez les assertions vraies En TDD, la couverture du code par les tests est nécéssairement de 100%. La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini. La couverture du code par les tests permet de mesurer objectivement la qualité des tests. Question 3 ♣ Tests unitaires vs tests d'intégration - Cochez les assertions vraies Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.



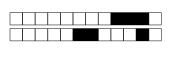
Question 4 - TDD - Cochez les assertions vraies
Le TDD améliore la conception d'un programme.
Le refactoring consiste à remanier le code d'un programme.
Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.
En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Question 5 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente
80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.



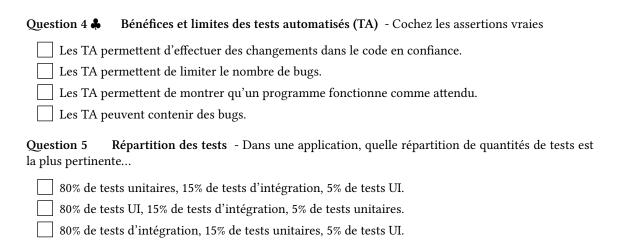
0 0	 ✓ Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom :
8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9	
Durée totale : 51 mn (1h08 mn pour les tiers temps) Aucun document ni dispositif électronique n'est aut Le symbole "trèfle" sur une question indique qu'il p Les réponses doivent être fournies uniquement sur le prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une répons re-dessiner la case à cocher.	torisé. eut être attendu plus d'une réponse correcte. es feuilles d'énoncé. Aucune autre réponse ne sera e entièrement coloriées en noires pour être prises
Question 1 Code coverage - Cochez les assertion	
Dans un projet agile, l'objectif de couverture de définition de fini.	code par les tests est un bon critère à ajouter à la
En TDD, la couverture du code par les tests est no	
La couverture du code par les tests permet de me	
sont invoquées par les tests.	avoir quelles classes, méthodes et lignes de codes
Question 2 🌲 Bénéfices et limites des tests automa	tisés (TA) - Cochez les assertions vraies
Les TA permettent d'effectuer des changements d	lans le code en confiance.
Les TA permettent de limiter le nombre de bugs.	
Les TA peuvent contenir des bugs.	
Les TA permettent de montrer qu'un programme	fonctionne comme attendu.
Question 3 4 Tests unitaires vs tests d'intégration	- Cochez les assertions vraies
Les tests d'intégration vérifient le fonctionnemen	it de "vrais" objets en interactions.
Les tests unitaires créent toujours les vrais collab	orateurs pour tester un objet.
Les tests unitaires consomment en général moins	de ressources que les tests d'intégration.
Les tests d'intégration s'exécutent en général plu	s rapidement que les tests unitaires.



Question 4 ♣ TDD - Cochez les assertions vraies
Le refactoring consiste à remanier le code d'un programme.
Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Le TDD améliore la conception d'un programme.
En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
Question 5 Répartition des tests - Dans une application, quelle répartition de quantités de tests es la plus pertinente
80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.



0 0	 ← Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom : 			
Durée totale : 51 mn (1h08 mn pour les tiers temps)				
Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.				
Question 1 Tests unitaires vs tests d'intégration - Cochez les assertions vraies Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.				
Les tests unitaires créent toujours les vrais collab Les tests unitaires consomment en général moins Les tests d'intégration s'exécutent en général plu	de ressources que les tests d'intégration.			
Question 2 🌲 TDD - Cochez les assertions vraies				
En TDD, dès qu'un test passe, on passe à l'écritur	re d'un nouveau test.			
En TDD, on code tous les tests d'une classe avant	de coder toutes les méthodes de la classe.			
	oppeur doit entreprendre après qu'un test échoue.			
Le refactoring consiste à remanier le code d'un pr				
Le TDD améliore la conception d'un programme.				
Question 3 & Code coverage - Cochez les assertion	ns vraies			
La couverture du code par les tests permet de me	surer objectivement la qualité des tests.			
La couverture du code par les tests permet de sa sont invoquées par les tests.	avoir quelles classes, méthodes et lignes de codes			
En TDD, la couverture du code par les tests est no	écéssairement de 100%.			
Dans un projet agile, l'objectif de couverture de définition de fini.	code par les tests est un bon critère à ajouter à la			



2022-2023 - R4-02 Qualité de développement - CC1 Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom : Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher. Question 1 🌲 Code coverage - Cochez les assertions vraies En TDD, la couverture du code par les tests est nécéssairement de 100%. La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests. La couverture du code par les tests permet de mesurer objectivement la qualité des tests. Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini. Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies Question 2 🐥 Les TA permettent d'effectuer des changements dans le code en confiance. Les TA permettent de limiter le nombre de bugs. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Les TA peuvent contenir des bugs. Question 3 🌲 Tests unitaires vs tests d'intégration - Cochez les assertions vraies Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires. Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions. Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet. Les tests unitaires consomment en général moins de ressources que les tests d'intégration.

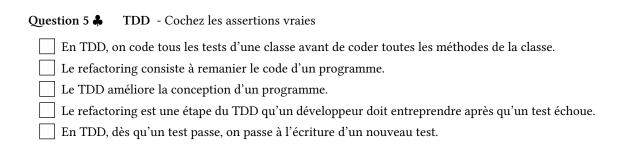
Question 4 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente...

80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.

80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.

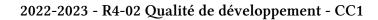
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.

80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.

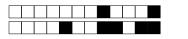


0 0	
Durée totale : 51 mn (1h08 mn pour les tiers tem Aucun document ni dispositif électronique n'est Le symbole "trèfle" sur une question indique qu'i Les réponses doivent être fournies uniquement su prise en compte dans la notation. Les cases doivent en compte dans la notation. Pour modifier une répo re-dessiner la case à cocher.	autorisé. l peut être attendu plus d'une réponse correcte. ir les feuilles d'énoncé. Aucune autre réponse ne sera être entièrement coloriées en noires pour être prises
sont invoquées par les tests.	mesurer objectivement la qualité des tests.
Question 2 ♣ TDD - Cochez les assertions vraies	3
En TDD, dès qu'un test passe, on passe à l'écri En TDD, on code tous les tests d'une classe ava Le refactoring consiste à remanier le code d'ur Le refactoring est une étape du TDD qu'un dév Le TDD améliore la conception d'un programm	ant de coder toutes les méthodes de la classe. n programme. veloppeur doit entreprendre après qu'un test échoue.
Question 3 🌲 Bénéfices et limites des tests autor	matisés (TA) - Cochez les assertions vraies
Les TA permettent de limiter le nombre de bug	gs.
Les TA peuvent contenir des bugs.	
Les TA permettent d'effectuer des changement	ts dans le code en confiance.
Les TA permettent de montrer qu'un programme	me fonctionne comme attendu.

Question 4 🌲	Tests unitaires vs tests d'intégration - Cochez les assertions vraies
Les tests	unitaires consomment en général moins de ressources que les tests d'intégration.
Les tests	unitaires créent toujours les vrais collaborateurs pour tester un objet.
Les tests	d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
Les tests	d'intégration s'exécutent en général plus rapidement que les tests unitaires.
Question 5 la plus pertiner	Répartition des tests - Dans une application, quelle répartition de quantités de tests es ate
80% de te	ests d'intégration, 15% de tests unitaires, 5% de tests UI.
80% de te	ests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de te	ests UI, 15% de tests d'intégration, 5% de tests unitaires.



0 0	← Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est autor Le symbole "trèfle" sur une question indique qu'il per Les réponses doivent être fournies uniquement sur le prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une réponse re-dessiner la case à cocher.	orisé. eut être attendu plus d'une réponse correcte. es feuilles d'énoncé. Aucune autre réponse ne sera e entièrement coloriées en noires pour être prises
Question 1 ♣ Tests unitaires vs tests d'intégration Les tests d'intégration s'exécutent en général plus Les tests d'intégration vérifient le fonctionnement Les tests unitaires consomment en général moins Les tests unitaires créent toujours les vrais collaborations	rapidement que les tests unitaires. t de "vrais" objets en interactions. de ressources que les tests d'intégration.
Question 2 ♣ Bénéfices et limites des tests automat	
Question 3 - Code coverage - Cochez les assertions	s vraies
 En TDD, la couverture du code par les tests est né La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de c définition de fini. La couverture du code par les tests permet de mes 	voir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la
	cation, quelle répartition de quantités de tests est
la plus pertinente 80% de tests unitaires, 15% de tests d'intégration, 5 80% de tests d'intégration, 15% de tests unitaires, 5 80% de tests UI, 15% de tests d'intégration, 5% de t	5% de tests UI.

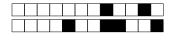


Ouestion 5 🐣	TDD -	Cochez les	assertions	vraies

En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
Le refactoring consiste à remanier le code d'un programme.
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Le TDD améliore la conception d'un programme.
Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.



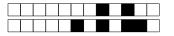
	llez coder votre numéro
	ntre, et écrire votre nom
444444 dans la case ci-d	essous.
	n:
999999	
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu pl Les réponses doivent être fournies uniquement sur les feuilles d'énonce prise en compte dans la notation. Les cases doivent être entièrement colo en compte dans la notation. Pour modifier une réponse, vous pouvez ut re-dessiner la case à cocher.	us d'une réponse correcte. É. Aucune autre réponse ne sera riées en noires pour être prises
Question 1 & TDD - Cochez les assertions vraies	
Le refactoring consiste à remanier le code d'un programme.	
Le refactoring est une étape du TDD qu'un développeur doit entrep	rendre après qu'un test échoue.
En TDD, on code tous les tests d'une classe avant de coder toutes le	s méthodes de la classe.
Le TDD améliore la conception d'un programme.	
En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau te	st.
Question 2 - Cochez les asser	tions vraies
Les tests unitaires créent toujours les vrais collaborateurs pour teste	er un objet.
Les tests d'intégration vérifient le fonctionnement de "vrais" objets	en interactions.
Les tests unitaires consomment en général moins de ressources que	les tests d'intégration.
Les tests d'intégration s'exécutent en général plus rapidement que l	es tests unitaires.
Question 3 Répartition des tests - Dans une application, quelle répa la plus pertinente	rtition de quantités de tests est
80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.	
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.	
80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.	



Question 4	4 & Code coverage - Cochez les assertions vraies
	s un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la nition de fini.
	ouverture du code par les tests permet de savoir quelles classes, méthodes et lignes de code invoquées par les tests.
La co	ouverture du code par les tests permet de mesurer objectivement la qualité des tests.
En T	TDD, la couverture du code par les tests est nécéssairement de 100%.
Question 5	5 & Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les	TA permettent d'effectuer des changements dans le code en confiance.
Les	TA permettent de limiter le nombre de bugs.
Les	TA peuvent contenir des bugs.
Les	TA permettent de montrer qu'un programme fonctionne comme attendu.

$\begin{array}{c ccccccccccccccccccccccccccccccccccc$				
	← Veuillez coder votre numéro			
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.			
	dans it case of desisous.			
	Nom et prénom :			
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.				
Question 1 . TDD - Cochez les assertions vraies				
Le refactoring consiste à remanier le code d'un pr	rogramme			
Le refactoring est une étape du TDD qu'un dévelo				
En TDD, dès qu'un test passe, on passe à l'écritur				
Le TDD améliore la conception d'un programme.				
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.				
Question 2 . Tests unitaires vs tests d'intégration	- Cocnez les assertions vraies			
Les tests unitaires consomment en général moins				
Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.				
Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.				
Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.				
Question 3 & Code coverage - Cochez les assertions vraies				
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.				
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.				
En TDD, la couverture du code par les tests est nécéssairement de 100%.				
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.				

Question 4 🐥	Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les TA po	ermettent de montrer qu'un programme fonctionne comme attendu.
Les TA pe	ermettent de limiter le nombre de bugs.
Les TA pe	euvent contenir des bugs.
Les TA po	ermettent d'effectuer des changements dans le code en confiance.
Question 5 la plus pertinen	Répartition des tests - Dans une application, quelle répartition de quantités de tests es te
80% de te	sts UI, 15% de tests d'intégration, 5% de tests unitaires.
80% de te	sts unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de te	sts d'intégration, 15% de tests unitaires, 5% de tests UI.

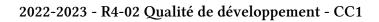


□ 0 □ 0			
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.			
Question 1 ♣ Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies □ Les TA permettent de limiter le nombre de bugs. □ Les TA permettent de montrer qu'un programme fonctionne comme attendu. □ Les TA peuvent contenir des bugs. □ Les TA permettent d'effectuer des changements dans le code en confiance.			
Question 2 ♣ TDD - Cochez les assertions vraies □ En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test. □ En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe. □ Le TDD améliore la conception d'un programme. □ Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue. □ Le refactoring consiste à remanier le code d'un programme.			
Question 3 ♣ Tests unitaires vs tests d'intégration - Cochez les assertions vraies Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires. Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet. Les tests unitaires consomment en général moins de ressources que les tests d'intégration. Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.			
Question 4 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires. 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI. 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.			

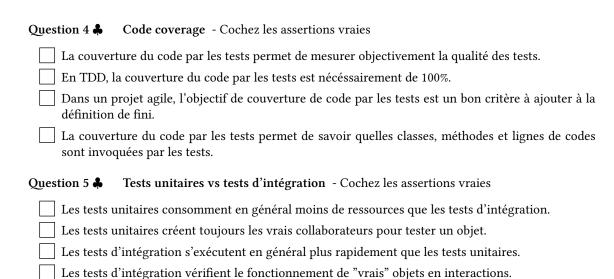


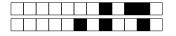
Question 5 ૈ	Code coverage	- Cochez l	es assertions	vraies

La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
En TDD, la couverture du code par les tests est nécéssairement de 100%.
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes

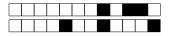


□2 □2 <t< th=""></t<>			
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.			
Question 1 ♣ TDD - Cochez les assertions vraies □ Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue. □ En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test. □ Le TDD améliore la conception d'un programme. □ En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe. □ Le refactoring consiste à remanier le code d'un programme.			
Question 2 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI. 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires. 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.			
Question 3 ♣ Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies □ Les TA permettent de montrer qu'un programme fonctionne comme attendu. □ Les TA peuvent contenir des bugs. □ Les TA permettent de limiter le nombre de bugs. □ Les TA permettent d'effectuer des changements dans le code en confiance.			





3 3 3 3 3 3 3 3 4 Veuillez coder votre numéro			
d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.			
5 5 5 5 5 5 5			
Nom et prénom :			
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.			
Question 1 & Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies			
Les TA permettent de limiter le nombre de bugs.			
Les TA permettent d'effectuer des changements dans le code en confiance.			
Les TA permettent de montrer qu'un programme fonctionne comme attendu.			
Les TA peuvent contenir des bugs.			
Question 2 - Cochez les assertions vraies			
Les tests unitaires consomment en général moins de ressources que les tests d'intégration.			
Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.			
Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.			
Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.			
Question 3 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente			
80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.			
80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.			
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.			
Question 4 & Code coverage - Cochez les assertions vraies			
En TDD, la couverture du code par les tests est nécéssairement de 100%.			
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.			
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.			
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.			



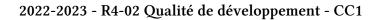
Ouestion 5 🌲	TDD	- Cochez les	assertions	vraies

_	
	En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
	Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue
	En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
	Le refactoring consiste à remanier le code d'un programme.
	Le TDD améliore la conception d'un programme.



$egin{array}{cccccccccccccccccccccccccccccccccccc$				
	← Veuillez coder votre numéro			
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.			
	ualis la case ci-uessous.			
	Nom et prénom :			
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.				
Question 1 & Code coverage - Cochez les assertion	ns vraies			
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.				
En TDD, la couverture du code par les tests est nécéssairement de 100%.				
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.				
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.				
Question 2 ♣ TDD - Cochez les assertions vraies				
Le TDD améliore la conception d'un programme.				
Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.				
En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.				
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.				
Le refactoring consiste à remanier le code d'un programme.				
Question 3 4 Tests unitaires vs tests d'intégration	- Cochez les assertions vraies			
Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.				
Les tests unitaires consomment en général moins de ressources que les tests d'intégration.				
Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.				
Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.				

Question 4 la plus pertine	Répartition des tests - Dans une application, quelle répartition de quantités de tests es nte			
80% de t	ests UI, 15% de tests d'intégration, 5% de tests unitaires.			
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.				
80% de t	ests unitaires, 15% de tests d'intégration, 5% de tests UI.			
Question 5 ૈ	Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies			
Les TA p	permettent de montrer qu'un programme fonctionne comme attendu.			
Les TA p	peuvent contenir des bugs.			
Les TA p	permettent de limiter le nombre de bugs.			
Les TA r	permettent d'effectuer des changements dans le code en confiance.			

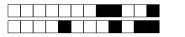


8888888				
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.				
Question 1 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI. 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires. 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.				
Question 2 4 TDD - Cochez les assertions vraies				
Le refactoring consiste à remanier le code d'un programme. Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue. En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test. Le TDD améliore la conception d'un programme. En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.				
Question 3 - Code coverage - Cochez les assertions vraies				
En TDD, la couverture du code par les tests est nécéssairement de 100%.				
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.				
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.				
La couverture du code par les tests permet de mes	urer objectivement la qualité des tests.			

Question 4 🌲	Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les TA pe	rmettent d'effectuer des changements dans le code en confiance.
Les TA pe	rmettent de montrer qu'un programme fonctionne comme attendu.
Les TA pe	uvent contenir des bugs.
Les TA pe	rmettent de limiter le nombre de bugs.
Question 5 🌲	Tests unitaires vs tests d'intégration - Cochez les assertions vraies
Les tests ı	unitaires consomment en général moins de ressources que les tests d'intégration.
Les tests o	l'intégration vérifient le fonctionnement de "vrais" objets en interactions.
Les tests u	unitaires créent toujours les vrais collaborateurs pour tester un objet.
Les tests o	



$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	
	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	duns it case of dessous.
	Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps) Aucun document ni dispositif électronique n'est au Le symbole "trèfle" sur une question indique qu'il p Les réponses doivent être fournies uniquement sur l	torisé. Jeut être attendu plus d'une réponse correcte.
prise en compte dans la notation. Les cases doivent êtr	-
en compte dans la notation. Pour modifier une répon-	se, vous pouvez utiliser un correcteur blanc, sans
re-dessiner la case à cocher.	
Question 1 🌲 Bénéfices et limites des tests automa	tisés (TA) - Cochez les assertions vraies
Les TA permettent d'effectuer des changements d	dans le code en confiance.
Les TA peuvent contenir des bugs.	
Les TA permettent de limiter le nombre de bugs.	
Les TA permettent de montrer qu'un programme	e fonctionne comme attendu.
Les TA permettent de montrer qu'un programme Question 2 Code coverage - Cochez les assertion	
 Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. 	ns vraies
 Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. 	ns vraies avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la
 Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de 	ns vraies avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%.
 Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. En TDD, la couverture du code par les tests est n La couverture du code par les tests permet de me 	ns vraies avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%. esurer objectivement la qualité des tests.
 Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. En TDD, la couverture du code par les tests est n La couverture du code par les tests permet de me Question 3 ♣ Tests unitaires vs tests d'intégration 	ns vraies avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%. esurer objectivement la qualité des tests. - Cochez les assertions vraies
 Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. En TDD, la couverture du code par les tests est n La couverture du code par les tests permet de me Question 3 ♣ Tests unitaires vs tests d'intégration Les tests unitaires créent toujours les vrais collaboration 	ns vraies avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%. esurer objectivement la qualité des tests. - Cochez les assertions vraies porateurs pour tester un objet.
 Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. En TDD, la couverture du code par les tests est n La couverture du code par les tests permet de me Question 3 ♣ Tests unitaires vs tests d'intégration Les tests unitaires créent toujours les vrais collable Les tests d'intégration vérifient le fonctionnemer 	avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%. esurer objectivement la qualité des tests. - Cochez les assertions vraies porateurs pour tester un objet. nt de "vrais" objets en interactions.
 Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. En TDD, la couverture du code par les tests est n La couverture du code par les tests permet de me Question 3 ♣ Tests unitaires vs tests d'intégration Les tests unitaires créent toujours les vrais collable Les tests d'intégration vérifient le fonctionnemer Les tests d'intégration s'exécutent en général plus 	ns vraies avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%. esurer objectivement la qualité des tests. - Cochez les assertions vraies corateurs pour tester un objet. ent de "vrais" objets en interactions. es rapidement que les tests unitaires.
 Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. En TDD, la couverture du code par les tests est n La couverture du code par les tests permet de me Question 3 ♣ Tests unitaires vs tests d'intégration Les tests unitaires créent toujours les vrais collable Les tests d'intégration vérifient le fonctionnemer 	ns vraies avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%. esurer objectivement la qualité des tests. - Cochez les assertions vraies corateurs pour tester un objet. ent de "vrais" objets en interactions. es rapidement que les tests unitaires.
Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. En TDD, la couverture du code par les tests est n La couverture du code par les tests permet de me Question 3 ♣ Tests unitaires vs tests d'intégration Les tests unitaires créent toujours les vrais collab Les tests d'intégration vérifient le fonctionnemer Les tests d'intégration s'exécutent en général plu Les tests unitaires consomment en général moins	ns vraies avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%. esurer objectivement la qualité des tests. - Cochez les assertions vraies corateurs pour tester un objet. ent de "vrais" objets en interactions. es rapidement que les tests unitaires.
Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. En TDD, la couverture du code par les tests est n La couverture du code par les tests permet de me Question 3 ♣ Tests unitaires vs tests d'intégration Les tests unitaires créent toujours les vrais collabement les tests d'intégration vérifient le fonctionnemer Les tests d'intégration s'exécutent en général plu Les tests unitaires consomment en général moins Question 4 Répartition des tests - Dans une appli	avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%. Esurer objectivement la qualité des tests. - Cochez les assertions vraies porateurs pour tester un objet. Int de "vrais" objets en interactions. Its rapidement que les tests unitaires. Its de ressources que les tests d'intégration. Its de codes
Les TA permettent de montrer qu'un programme Question 2 ♣ Code coverage - Cochez les assertion La couverture du code par les tests permet de sa sont invoquées par les tests. Dans un projet agile, l'objectif de couverture de définition de fini. En TDD, la couverture du code par les tests est n La couverture du code par les tests permet de me Question 3 ♣ Tests unitaires vs tests d'intégration Les tests unitaires créent toujours les vrais collaber les tests d'intégration vérifient le fonctionnemer les tests d'intégration s'exécutent en général plus Les tests unitaires consomment en général moins Question 4 Répartition des tests - Dans une applit la plus pertinente	avoir quelles classes, méthodes et lignes de codes code par les tests est un bon critère à ajouter à la écéssairement de 100%. Esurer objectivement la qualité des tests. - Cochez les assertions vraies porateurs pour tester un objet. Ent de "vrais" objets en interactions. Es rapidement que les tests unitaires. Es de ressources que les tests d'intégration. Estation, quelle répartition de quantités de tests est

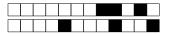


Ouestion 5 🌲	TDD	- Cochez les	assertions	vraies

Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Le refactoring consiste à remanier le code d'un programme.
En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
Le TDD améliore la conception d'un programme.

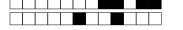


□ 0 □ 0
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.
Question 1 ♣ Tests unitaires vs tests d'intégration - Cochez les assertions vraies Les tests unitaires consomment en général moins de ressources que les tests d'intégration. Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires. Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions. Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
 Question 2 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires. 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI. 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
Question 3 ♣ TDD - Cochez les assertions vraies Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue. En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test. Le refactoring consiste à remanier le code d'un programme. Le TDD améliore la conception d'un programme. En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Question 4 ♣ Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements dans le code en confiance. Les TA peuvent contenir des bugs. Les TA permettent de montrer qu'un programme fonctionne comme attendu.

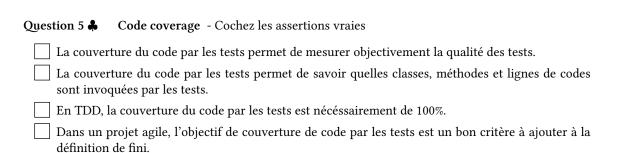


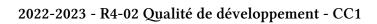
Question 5 🌲	Code coverage	- Cochez les	assertions	vraies

La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
En TDD, la couverture du code par les tests est nécéssairement de 100%.

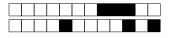


0 0	
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est auto Le symbole "trèfle" sur une question indique qu'il per Les réponses doivent être fournies uniquement sur les prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une réponse re-dessiner la case à cocher.	risé. ut être attendu plus d'une réponse correcte. s feuilles d'énoncé. Aucune autre réponse ne sera entièrement coloriées en noires pour être prises
Question 1 ♣ TDD - Cochez les assertions vraies	
 □ En TDD, on code tous les tests d'une classe avant d □ Le refactoring consiste à remanier le code d'un pro □ Le TDD améliore la conception d'un programme. □ En TDD, dès qu'un test passe, on passe à l'écriture □ Le refactoring est une étape du TDD qu'un dévelopment de la conception d'un programme. 	ogramme. d'un nouveau test.
Question 2 Répartition des tests - Dans une applica la plus pertinente	ation, quelle répartition de quantités de tests est
80% de tests UI, 15% de tests d'intégration, 5% de te	ests unitaires.
80% de tests d'intégration, 15% de tests unitaires, 5	% de tests UI.
80% de tests unitaires, 15% de tests d'intégration, 5	% de tests UI.
Question 3 4 Tests unitaires vs tests d'intégration	Cochez les assertions vraies
Les tests d'intégration s'exécutent en général plus Les tests d'intégration vérifient le fonctionnement Les tests unitaires créent toujours les vrais collabo Les tests unitaires consomment en général moins of	de "vrais" objets en interactions. rateurs pour tester un objet.
Question 4 🌲 Bénéfices et limites des tests automati	sés (TA) - Cochez les assertions vraies
Les TA peuvent contenir des bugs. Les TA permettent d'effectuer des changements da Les TA permettent de limiter le nombre de bugs. Les TA permettent de montrer qu'un programme f	ns le code en confiance.





□ 0 □ 0
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.
Question 1 ♣ Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies Les TA peuvent contenir des bugs. Les TA permettent de limiter le nombre de bugs. Les TA permettent de montrer qu'un programme fonctionne comme attendu. Les TA permettent d'effectuer des changements dans le code en confiance.
Question 2 ♣ Tests unitaires vs tests d'intégration - Cochez les assertions vraies Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet. Les tests unitaires consomment en général moins de ressources que les tests d'intégration. Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions. Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
Question 3 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI. 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires. 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
Question 4 ♣ TDD - Cochez les assertions vraies □ En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test. □ Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue. □ En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe. □ Le TDD améliore la conception d'un programme. □ Le refactoring consiste à remanier le code d'un programme.

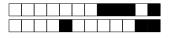


Question 5 ૈ	Code coverage	- Cochez les	assertions v	raies

Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
En TDD, la couverture du code par les tests est nécéssairement de 100%.
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.

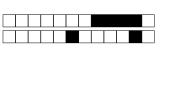


\square 3	← Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom
$\square 4 \ \square 4$	dans la case ci-dessous.
5555555	N /
	Nom et prénom :
8888888	
9999999	
Durée totale : 51 mn (1h08 mn pour les tiers temps Aucun document ni dispositif électronique n'est au Le symbole "trèfle" sur une question indique qu'il p Les réponses doivent être fournies uniquement sur prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une répon re-dessiner la case à cocher.	torisé. Deut être attendu plus d'une réponse correcte. Les feuilles d'énoncé. Aucune autre réponse ne sera re entièrement coloriées en noires pour être prises
Les TA permettent de limiter le nombre de bugs.	atisés (TA) - Cochez les assertions vraies
Les TA peuvent contenir des bugs. Les TA permettent de montrer qu'un programme	e fonctionne comme attendu
	t fonetionne comme attenda.
Les 1A permettent d'effectuer des changements	
Les TA permettent d'effectuer des changements Question 2 Répartition des tests - Dans une appl la plus pertinente	
Question 2 Répartition des tests - Dans une appl	dans le code en confiance. ication, quelle répartition de quantités de tests est
Question 2 Répartition des tests - Dans une appl la plus pertinente	dans le code en confiance. ication, quelle répartition de quantités de tests est , 5% de tests UI.
Question 2 Répartition des tests - Dans une appl la plus pertinente 80% de tests unitaires, 15% de tests d'intégration	dans le code en confiance. ication, quelle répartition de quantités de tests est , 5% de tests UI. , 5% de tests UI.
Question 2 Répartition des tests - Dans une appl la plus pertinente 80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaires	dans le code en confiance. ication, quelle répartition de quantités de tests est , 5% de tests UI. , 5% de tests UI. tests unitaires.
Question 2 Répartition des tests - Dans une appl la plus pertinente 80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaires 80% de tests UI, 15% de tests d'intégration, 5% de Question 3 ♣ Tests unitaires vs tests d'intégration Les tests d'intégration s'exécutent en général plu	dans le code en confiance. ication, quelle répartition de quantités de tests est 5% de tests UI. 5% de tests UI. tests unitaires. - Cochez les assertions vraies as rapidement que les tests unitaires.
Question 2 Répartition des tests - Dans une appl la plus pertinente 80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaires 80% de tests UI, 15% de tests d'intégration, 5% de Question 3 ♣ Tests unitaires vs tests d'intégration Les tests d'intégration s'exécutent en général plu Les tests d'intégration vérifient le fonctionneme	dans le code en confiance. ication, quelle répartition de quantités de tests est , 5% de tests UI. , 5% de tests UI. tests unitaires. - Cochez les assertions vraies as rapidement que les tests unitaires. and de "vrais" objets en interactions.
Question 2 Répartition des tests - Dans une appl la plus pertinente 80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaires 80% de tests UI, 15% de tests d'intégration, 5% de Question 3 ♣ Tests unitaires vs tests d'intégration Les tests d'intégration s'exécutent en général plu Les tests d'intégration vérifient le fonctionneme Les tests unitaires créent toujours les vrais collaires	dans le code en confiance. ication, quelle répartition de quantités de tests est 5% de tests UI. 5% de tests UI. tests unitaires. - Cochez les assertions vraies as rapidement que les tests unitaires. nt de "vrais" objets en interactions. porateurs pour tester un objet.
Question 2 Répartition des tests - Dans une appl la plus pertinente 80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaires 80% de tests UI, 15% de tests d'intégration, 5% de Question 3 ♣ Tests unitaires vs tests d'intégration Les tests d'intégration s'exécutent en général plu Les tests d'intégration vérifient le fonctionneme	dans le code en confiance. ication, quelle répartition de quantités de tests est 5% de tests UI. 5% de tests UI. tests unitaires. - Cochez les assertions vraies as rapidement que les tests unitaires. nt de "vrais" objets en interactions. porateurs pour tester un objet.
Question 2 Répartition des tests - Dans une appl la plus pertinente 80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaires 80% de tests UI, 15% de tests d'intégration, 5% de Question 3 ♣ Tests unitaires vs tests d'intégration Les tests d'intégration s'exécutent en général plu Les tests d'intégration vérifient le fonctionneme Les tests unitaires créent toujours les vrais collaires	dans le code en confiance. ication, quelle répartition de quantités de tests est 5% de tests UI. 5% de tests UI. tests unitaires. - Cochez les assertions vraies as rapidement que les tests unitaires. Int de "vrais" objets en interactions. corateurs pour tester un objet. s de ressources que les tests d'intégration.
Question 2 Répartition des tests - Dans une appl la plus pertinente 80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaires 80% de tests UI, 15% de tests d'intégration, 5% de Question 3 ♣ Tests unitaires vs tests d'intégration Les tests d'intégration s'exécutent en général plu Les tests d'intégration vérifient le fonctionneme Les tests unitaires créent toujours les vrais collai Les tests unitaires consomment en général moin	dans le code en confiance. ication, quelle répartition de quantités de tests est 5% de tests UI. 5% de tests UI. tests unitaires. - Cochez les assertions vraies as rapidement que les tests unitaires. Int de "vrais" objets en interactions. corateurs pour tester un objet. s de ressources que les tests d'intégration. Ins vraies
Question 2 Répartition des tests - Dans une applia plus pertinente ■ 80% de tests unitaires, 15% de tests d'intégration ■ 80% de tests d'intégration, 15% de tests unitaires ■ 80% de tests UI, 15% de tests d'intégration, 5% de Question 3 ♣ Tests unitaires vs tests d'intégration ■ Les tests d'intégration s'exécutent en général plu ■ Les tests d'intégration vérifient le fonctionneme ■ Les tests unitaires créent toujours les vrais collal ■ Les tests unitaires consomment en général moin Question 4 ♣ Code coverage - Cochez les assertio ■ En TDD, la couverture du code par les tests est rest	dans le code en confiance. ication, quelle répartition de quantités de tests est 5% de tests UI. 5% de tests UI. tests unitaires. - Cochez les assertions vraies as rapidement que les tests unitaires. Int de "vrais" objets en interactions. corateurs pour tester un objet. s de ressources que les tests d'intégration. Ins vraies
Question 2 Répartition des tests - Dans une applia plus pertinente 80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaires 80% de tests UI, 15% de tests d'intégration, 5% de Question 3 ♣ Tests unitaires vs tests d'intégration Les tests d'intégration s'exécutent en général plu Les tests d'intégration vérifient le fonctionneme Les tests unitaires créent toujours les vrais collai Les tests unitaires consomment en général moin Question 4 ♣ Code coverage - Cochez les assertio En TDD, la couverture du code par les tests est r La couverture du code par les tests permet de s sont invoquées par les tests.	dans le code en confiance. ication, quelle répartition de quantités de tests est 5% de tests UI. 5% de tests UI. tests unitaires. - Cochez les assertions vraies as rapidement que les tests unitaires. Int de "vrais" objets en interactions. porateurs pour tester un objet. Is de ressources que les tests d'intégration. Ins vraies décéssairement de 100%.

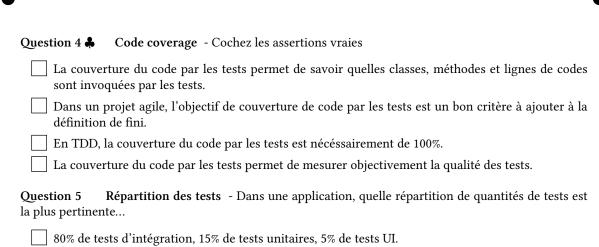


Ouestion 5 🌲	TDD -	Cochez les	assertions	vraies

Le refactoring consiste à remanier le code d'un programme.
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue
En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
Le TDD améliore la conception d'un programme.

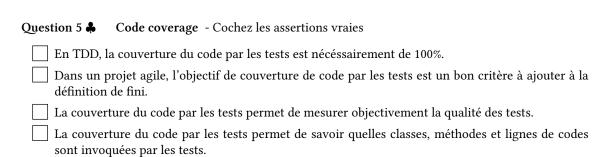


\square 2	
	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	dans la case ci-dessous.
	Nom et prénom :
8 8 8 8 8 8 8 8	
9999999	
Durée totale : 51 mn (1h08 mn pour les tiers temps) Aucun document ni dispositif électronique n'est au Le symbole "trèfle" sur une question indique qu'il p Les réponses doivent être fournies uniquement sur l prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une répons re-dessiner la case à cocher.	torisé. Jeut être attendu plus d'une réponse correcte. les feuilles d'énoncé. Aucune autre réponse ne sera le entièrement coloriées en noires pour être prises
Question 1 ♣ Bénéfices et limites des tests automa Les TA permettent de limiter le nombre de bugs.	tisés (TA) - Cochez les assertions vraies
	fonctionne comme ettendu
Les TA permettent de montrer qu'un programme	
Les TA permettent d'effectuer des changements d Les TA peuvent contenir des bugs.	ians le coue en comfance.
Question 2 4 TDD - Cochez les assertions vraies	
En TDD, on code tous les tests d'une classe avant	t de coder toutes les méthodes de la classe.
Le refactoring est une étape du TDD qu'un dével	oppeur doit entreprendre après qu'un test échoue.
Le refactoring consiste à remanier le code d'un p	rogramme.
En TDD, dès qu'un test passe, on passe à l'écritur	re d'un nouveau test.
Le TDD améliore la conception d'un programme	
Question 3 4 Tests unitaires vs tests d'intégration	- Cochez les assertions vraies
Les tests d'intégration vérifient le fonctionnemer	nt de "vrais" objets en interactions.
Les tests unitaires créent toujours les vrais collab	oorateurs pour tester un objet.
Les tests d'intégration s'exécutent en général plu	s rapidement que les tests unitaires.
Les tests unitaires consomment en général moins	s de ressources que les tests d'intégration.



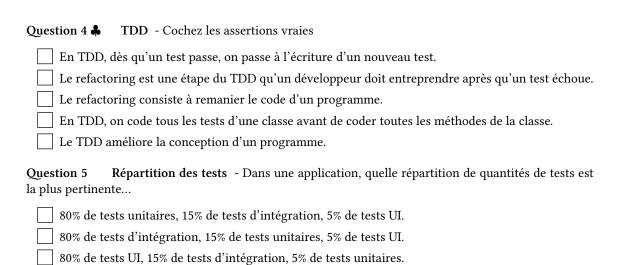
80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.

	\longleftarrow Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	dans la case ci-dessous.
	Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est aut Le symbole "trèfle" sur une question indique qu'il pe Les réponses doivent être fournies uniquement sur le prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une réponse	orisé. eut être attendu plus d'une réponse correcte. es feuilles d'énoncé. Aucune autre réponse ne sera e entièrement coloriées en noires pour être prises
re-dessiner la case à cocher.	e, reac pearez annoez un correcteur zame, same
Question 1 Répartition des tests - Dans une applic la plus pertinente	cation, quelle répartition de quantités de tests est
80% de tests d'intégration, 15% de tests unitaires,	5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de t	ests unitaires.
80% de tests unitaires, 15% de tests d'intégration,	5% de tests UI.
Question 2 🌲 Tests unitaires vs tests d'intégration	- Cochez les assertions vraies
Les tests unitaires créent toujours les vrais collabo	orateurs pour tester un objet.
Les tests d'intégration vérifient le fonctionnement	t de "vrais" objets en interactions.
Les tests d'intégration s'exécutent en général plus	rapidement que les tests unitaires.
Les tests unitaires consomment en général moins	de ressources que les tests d'intégration.
Question 3 & Bénéfices et limites des tests automat	isés (TA) - Cochez les assertions vraies
Les TA permettent de montrer qu'un programme	fonctionne comme attendu.
Les TA peuvent contenir des bugs.	
Les TA permettent d'effectuer des changements d	ans le code en confiance.
Les TA permettent de limiter le nombre de bugs.	
Question 4 & TDD - Cochez les assertions vraies	
En TDD, on code tous les tests d'une classe avant	de coder toutes les méthodes de la classe.
En TDD, dès qu'un test passe, on passe à l'écriture	e d'un nouveau test.
Le refactoring est une étape du TDD qu'un dévelo	ppeur doit entreprendre après qu'un test échoue.
Le TDD améliore la conception d'un programme.	
Le refactoring consiste à remanier le code d'un pr	ogramme.



2022-2023 - R4-02 Qualit

	← Veuillez coder votre numéro			
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.			
	ualis la case ci-dessous.			
	Nom et prénom :			
9 9 9 9 9 9 9				
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.				
Question 1 & Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies				
Les TA peuvent contenir des bugs.	C 1			
Les TA permettent de montrer qu'un program				
Les TA permettent de limiter le nombre de bu				
Les TA permettent d'effectuer des changements dans le code en confiance.				
Question 2 🌲 Tests unitaires vs tests d'intégration - Cochez les assertions vraies				
Les tests unitaires consomment en général moins de ressources que les tests d'intégration.				
Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.				
Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.				
Les tests d'intégration vérifient le fonctionner	nent de "vrais" objets en interactions.			
Question 3 - Code coverage - Cochez les assertions vraies				
En TDD, la couverture du code par les tests es	st nécéssairement de 100%.			
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.				
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.				
La couverture du code par les tests permet de	mesurer objectivement la qualité des tests.			



2022-2023 - R4-02 Qualité de développement - CC1 Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom : Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher. TDD - Cochez les assertions vraies Question 1 🌲 Le TDD améliore la conception d'un programme. En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test. En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe. Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue. Le refactoring consiste à remanier le code d'un programme. Question 2 🚓 Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies Les TA peuvent contenir des bugs. Les TA permettent d'effectuer des changements dans le code en confiance.

Question 3 ♣ Code coverage - Cochez les assertions vraies
 □ Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
 □ La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
 □ La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
 □ En TDD, la couverture du code par les tests est nécéssairement de 100%.

Les TA permettent de montrer qu'un programme fonctionne comme attendu.

Les TA permettent de limiter le nombre de bugs.

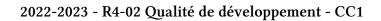
Question 4 la plus pertine	Répartition des tests - Dans une application, quelle répartition de quantités de tests es ente
80% de t	rests UI, 15% de tests d'intégration, 5% de tests unitaires.
80% de t	ests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de 1	ests d'intégration, 15% de tests unitaires, 5% de tests UI.
Question 5 🌲	Tests unitaires vs tests d'intégration - Cochez les assertions vraies
Les test	s d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
Les test	s unitaires créent toujours les vrais collaborateurs pour tester un objet.
Les test	s d'intégration s'exécutent en général plus rapidement que les tests unitaires.
Les test	s unitaires consomment en général moins de ressources que les tests d'intégration.



\square_2 \square_2 \square_2 \square_2 \square_2 \square_2 \square_2				
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	Veuillez coder votre numéro			
	at ci-contre, et écrire votre nom			
	ase ci-dessous.			
Nom et	prénom :			
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QC Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être atte Les réponses doivent être fournies uniquement sur les feuilles d prise en compte dans la notation. Les cases doivent être entièreme en compte dans la notation. Pour modifier une réponse, vous pour re-dessiner la case à cocher.	endu plus d'une réponse correcte. 'énoncé. Aucune autre réponse ne sera ent coloriées en noires pour être prises			
Question 1 & Code coverage - Cochez les assertions vraies				
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.				
En TDD, la couverture du code par les tests est nécéssaireme	ent de 100%.			
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.				
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.				
Question 2 4 Tests unitaires vs tests d'intégration - Cochez le	es assertions vraies			
Les tests d'intégration vérifient le fonctionnement de "vrais"	objets en interactions.			
Les tests unitaires consomment en général moins de ressour				
Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.				
Les tests d'intégration s'exécutent en général plus rapidemen	-			
	-			
Question 3 4 Bénéfices et limites des tests automatisés (TA)	- Cochez les assertions vraies			
Les TA permettent de limiter le nombre de bugs.				
Les TA peuvent contenir des bugs.				
Les TA permettent de montrer qu'un programme fonctionne	comme attendu.			
Les TA permettent d'effectuer des changements dans le code	en confiance.			



Question 4 . TDD - Cochez les assertions vraies
Le TDD améliore la conception d'un programme.
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue
En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
Le refactoring consiste à remanier le code d'un programme.
Question 5 Répartition des tests - Dans une application, quelle répartition de quantités de tests es la plus pertinente
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de tests III 15% de tests d'intégration 5% de tests unitaires



□ 0 □ 0				
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.				
Question 1 ♣ Tests unitaires vs tests d'intégration - Cochez les assertions vraies □ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet. □ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions. □ Les tests unitaires consomment en général moins de ressources que les tests d'intégration. □ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.				
Question 2 - Code coverage - Cochez les assertions vraies				
En TDD, la couverture du code par les tests est nécéssairement de 100%.				
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.				
 La couverture du code par les tests permet de mesurer objectivement la qualité des tests. Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini. 				
Question 3 ♣ TDD - Cochez les assertions vraies				
 En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test. Le refactoring consiste à remanier le code d'un programme. Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue. En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe. Le TDD améliore la conception d'un programme. 				

Question 4 la plus pertine	Répartition des tests - Dans une application, quelle répartition de quantités de tests es nte
80% de t	ests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de t	ests UI, 15% de tests d'intégration, 5% de tests unitaires.
80% de t	ests d'intégration, 15% de tests unitaires, 5% de tests UI.
Question 5 ♣	Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les TA p	permettent de montrer qu'un programme fonctionne comme attendu.
Les TA p	peuvent contenir des bugs.
Les TA p	permettent de limiter le nombre de bugs.
Les TA	permettent d'effectuer des changements dans le code en confiance.



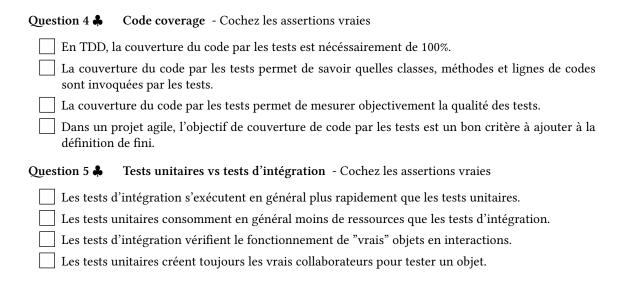
3 3 3 3 3 3 3 4 Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom				
$\boxed{}4 \ \boxed{}4 \ \boxed{}4 \ \boxed{}4 \ \boxed{}4 \ \boxed{}4 \ \boxed{}4 \ \boxed{}4$ dans la case ci-dessous.				
5 5 5 5 5 5 Nom et prénom :				
8 8 8 8 8 8 8				
Durée totale : 51 mn (1h08 mn pour les tiers temps). Partie QCM sur 5 points. Aucun document ni dispositif électronique n'est autorisé. Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte. Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prises en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, sans re-dessiner la case à cocher.				
Question 1 & Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies				
Les TA permettent de limiter le nombre de bugs.				
Les TA peuvent contenir des bugs. Les TA permettent d'effectuer des changements dans le code en confiance.				
Les TA permettent de montrer qu'un programme fonctionne comme attendu.				
Question 2 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente				
80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.				
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.				
80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.				
Question 3 - Cochez les assertions vraies				
Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.				
Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.				
Les tests unitaires consomment en général moins de ressources que les tests d'intégration.				
Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.				
Question 4 - Code coverage - Cochez les assertions vraies				
En TDD, la couverture du code par les tests est nécéssairement de 100%.				
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.				
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.				
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la				



	Ouestion 5	TDD	- Cochez le	s assertions	vraies
--	------------	-----	-------------	--------------	--------

	Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.
	Le TDD améliore la conception d'un programme.
	Le refactoring consiste à remanier le code d'un programme.
	En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Γ	En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.

	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	Nom et prénom :
Les réponses doivent être fournies uniquement prise en compte dans la notation. Les cases doiven	- · · · · · · · · · · · · · · · · · · ·
Question 1 A TDD - Cochez les assertions vra	ies
Le refactoring est une étape du TDD qu'un d	léveloppeur doit entreprendre après qu'un test échoue.
Le refactoring consiste à remanier le code d'	un programme.
En TDD, on code tous les tests d'une classe a	avant de coder toutes les méthodes de la classe.
En TDD, dès qu'un test passe, on passe à l'éc	criture d'un nouveau test.
Le TDD améliore la conception d'un prograr	nme.
Question 2 🌲 Bénéfices et limites des tests aut	tomatisés (TA) - Cochez les assertions vraies
Les TA permettent d'effectuer des changeme	ents dans le code en confiance.
Les TA permettent de limiter le nombre de b	ougs.
Les TA peuvent contenir des bugs.	
Les TA permettent de montrer qu'un progra	mme fonctionne comme attendu.
Question 3 Répartition des tests - Dans une a la plus pertinente	application, quelle répartition de quantités de tests est
80% de tests UI, 15% de tests d'intégration, 59	% de tests unitaires.
80% de tests unitaires, 15% de tests d'intégra	
80% de tests d'intégration, 15% de tests unita	ires. 5% de tests UI.

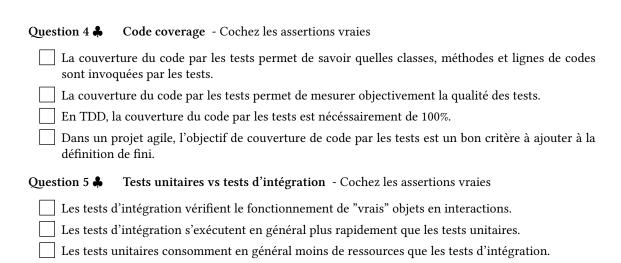


	✓ Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps). P Aucun document ni dispositif électronique n'est autor Le symbole "trèfle" sur une question indique qu'il peut Les réponses doivent être fournies uniquement sur les f prise en compte dans la notation. Les cases doivent être e en compte dans la notation. Pour modifier une réponse, re-dessiner la case à cocher.	isé. t être attendu plus d'une réponse correcte. feuilles d'énoncé. Aucune autre réponse ne sera ntièrement coloriées en noires pour être prises
Question 1 Tests unitaires vs tests d'intégration - Communication les tests unitaires consomment en général moins de Les tests d'intégration s'exécutent en général plus ra Les tests d'intégration vérifient le fonctionnement de Les tests unitaires créent toujours les vrais collaborations de la communication de la communicati	e ressources que les tests d'intégration. apidement que les tests unitaires. le "vrais" objets en interactions.
Question 2 Répartition des tests - Dans une applicat la plus pertinente 80% de tests unitaires, 15% de tests d'intégration, 5% 80% de tests d'intégration, 15% de tests unitaires, 5% 80% de tests UI, 15% de tests d'intégration, 5% de test	de tests UI.
Question 3 ♣ TDD - Cochez les assertions vraies □ En TDD, on code tous les tests d'une classe avant de □ Le refactoring consiste à remanier le code d'un prog □ En TDD, dès qu'un test passe, on passe à l'écriture d □ Le TDD améliore la conception d'un programme. □ Le refactoring est une étape du TDD qu'un développe	gramme. I'un nouveau test.



Question 4 🐥 🤍	Code coverage - Cochez les assertions vraies
En TDD, la	couverture du code par les tests est nécéssairement de 100%.
La couvertu	re du code par les tests permet de mesurer objectivement la qualité des tests.
Dans un pro définition de	ojet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la le fini.
	re du code par les tests permet de savoir quelles classes, méthodes et lignes de codes ées par les tests.
Question 5 🌲 🛚 I	Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les TA peuv	rent contenir des bugs.
Les TA perm	nettent d'effectuer des changements dans le code en confiance.
Les TA pern	nettent de montrer qu'un programme fonctionne comme attendu.
Les TA pern	nettent de limiter le nombre de bugs.

0 0	
Durée totale : 51 mn (1h08 mn pour les tiers temps Aucun document ni dispositif électronique n'est au Le symbole "trèfle" sur une question indique qu'il p Les réponses doivent être fournies uniquement sur prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une répon re-dessiner la case à cocher.	peut être attendu plus d'une réponse correcte. les feuilles d'énoncé. Aucune autre réponse ne sera re entièrement coloriées en noires pour être prises
Question 1 ♣ TDD - Cochez les assertions vraies Le refactoring est une étape du TDD qu'un dével Le refactoring consiste à remanier le code d'un p En TDD, dès qu'un test passe, on passe à l'écritu Le TDD améliore la conception d'un programme En TDD, on code tous les tests d'une classe avant	re d'un nouveau test.
Question 2 ♣ Bénéfices et limites des tests automa Les TA peuvent contenir des bugs. Les TA permettent de montrer qu'un programme Les TA permettent de limiter le nombre de bugs. Les TA permettent d'effectuer des changements	
la plus pertinente 80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaires	, 5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de	tests unitaires.



Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.

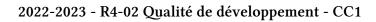


	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	dans it case of dessous.
	Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps) Aucun document ni dispositif électronique n'est aut Le symbole "trèfle" sur une question indique qu'il p Les réponses doivent être fournies uniquement sur le prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une répons re-dessiner la case à cocher.	corisé. eut être attendu plus d'une réponse correcte. es feuilles d'énoncé. Aucune autre réponse ne sera e entièrement coloriées en noires pour être prises
Question 1 . TDD - Cochez les assertions vraies	
	oppeur doit entreprendre après qu'un test échoue.
En TDD, dès qu'un test passe, on passe à l'écritur	
Le refactoring consiste à remanier le code d'un pr	
Le TDD améliore la conception d'un programme.	
En TDD, on code tous les tests d'une classe avant	de coder toutes les methodes de la classe.
Question 2 🌲 💮 Bénéfices et limites des tests automa	tisés (TA) - Cochez les assertions vraies
Les TA peuvent contenir des bugs.	
Les TA permettent de limiter le nombre de bugs.	
Les TA permettent d'effectuer des changements d	lans le code en confiance.
Les TA permettent de montrer qu'un programme	fonctionne comme attendu.
Question 3 4 Tests unitaires vs tests d'intégration	- Cochez les assertions vraies
Les tests d'intégration s'exécutent en général plu	s rapidement que les tests unitaires.
Les tests unitaires créent toujours les vrais collab	orateurs pour tester un objet.
Les tests unitaires consomment en général moins	de ressources que les tests d'intégration.
Les tests d'intégration vérifient le fonctionnemen	t de "vrais" objets en interactions.
Question 4 Répartition des tests - Dans une appli la plus pertinente	cation, quelle répartition de quantités de tests est
80% de tests d'intégration, 15% de tests unitaires,	5% de tests UI.
80% de tests unitaires, 15% de tests d'intégration,	
80% de tests UI, 15% de tests d'intégration, 5% de	

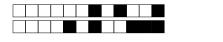


Question 5 & Code coverage - Cod	chez les assertions vraies
----------------------------------	----------------------------

La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
En TDD, la couverture du code par les tests est nécéssairement de 100%.



	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	dans la case el dessous.
	Nom et prénom :
\square_7 \square_7 \square_7 \square_7 \square_7 \square_7	
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est auto Le symbole "trèfle" sur une question indique qu'il pe Les réponses doivent être fournies uniquement sur le prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une réponse e-dessiner la case à cocher.	orisé. eut être attendu plus d'une réponse correcte. es feuilles d'énoncé. Aucune autre réponse ne sera e entièrement coloriées en noires pour être prises
Question 1 & Bénéfices et limites des tests automat Les TA permettent d'effectuer des changements de Les TA peuvent contenir des bugs.	isés (TA) - Cochez les assertions vraies ans le code en confiance.
Les TA permettent de limiter le nombre de bugs.	
Les TA permettent de montrer qu'un programme	fonctionne comme attendu.
	cation, quelle répartition de quantités de tests est
80% de tests unitaires, 15% de tests d'intégration, 5	5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% de t	
80% de tests d'intégration, 15% de tests unitaires, 5	
Question 3 A TDD - Cochez les assertions vraies	
Le TDD améliore la conception d'un programme.	
En TDD, on code tous les tests d'une classe avant	de coder toutes les méthodes de la classe.
Le refactoring est une étape du TDD qu'un dévelo	
Le refactoring consiste à remanier le code d'un pr	
En TDD, dès qu'un test passe, on passe à l'écriture	
Question 4 . Tests unitaires vs tests d'intégration	
Les tests unitaires consomment en général moins	-
Les tests unitaires créent toujours les vrais collabo	orateurs pour tester un objet.
Les tests d'integration verifient le fonctionnement	. 1 " . " 1
Les tests d'intégration s'exécutent en général plus	t de "vrais" objets en interactions.



Question 5 🦂	Code coverage	- Cochez les	assertions vraies
Quention 3	Couc coverage	COCIICZ ICS	abbertions viales

La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
En TDD, la couverture du code par les tests est nécéssairement de 100%.
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests