#### 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). 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

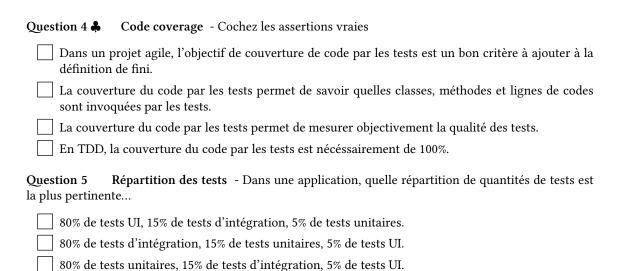
Les tests unitaires consomment en général moins de ressources que les tests d'intégration.

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.

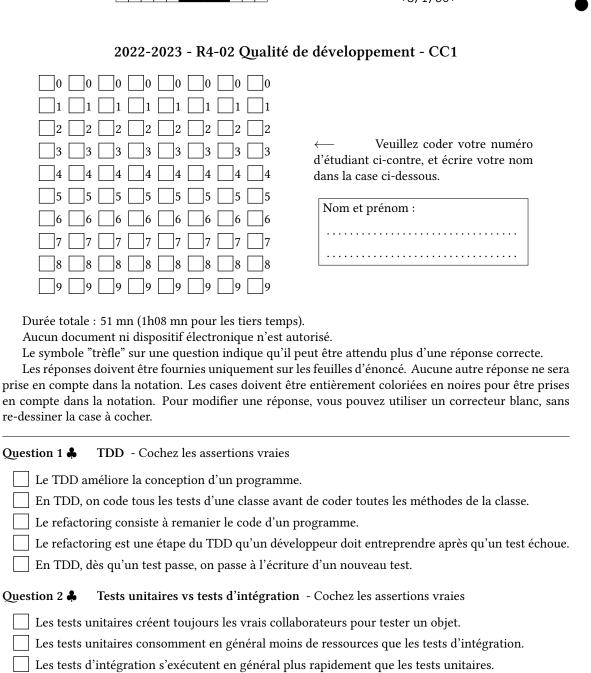
Question 3 🌲





$\square 3 \square 3$	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
555555	Nom et prénom :
6 6 6 6 6 6 6	Nom et prenom .
8 8 8 8 8 8 8 8	
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'il Les réponses doivent être fournies uniquement sur prise en compte dans la notation. Les cases doivent ê en compte dans la notation. Pour modifier une répo re-dessiner la case à cocher.	utorisé. peut être attendu plus d'une réponse correcte. r les feuilles d'énoncé. Aucune autre réponse ne sera tre entièrement coloriées en noires pour être prises
Question 1 & Code coverage - Cochez les assertie	
Dans un projet agile, l'objectif de couverture d définition de fini.	e code par les tests est un bon critère à ajouter à la
La couverture du code par les tests permet de n	nesurer objectivement la qualité des tests
En TDD, la couverture du code par les tests est	
	savoir quelles classes, méthodes et lignes de codes
Question 2 🌲 Tests unitaires vs tests d'intégration	on - Cochez les assertions vraies
Les tests d'intégration vérifient le fonctionneme	ent de "vrais" objets en interactions.
Les tests unitaires créent toujours les vrais colla	aborateurs pour tester un objet.
Les tests unitaires consomment en général moi	ns de ressources que les tests d'intégration.
Les tests d'intégration s'exécutent en général p	lus rapidement que les tests unitaires.
Question 3 Répartition des tests - Dans une app la plus pertinente	olication, quelle répartition de quantités de tests est
80% de tests d'intégration, 15% de tests unitaire	s, 5% de tests UI.
80% de tests UI, 15% de tests d'intégration, 5% d	le tests unitaires.
80% de tests unitaires, 15% de tests d'intégration	n, 5% de tests UI.
Question 4 🌲 Bénéfices et limites des tests auton	natisés (TA) - Cochez les assertions vraies
Les TA permettent d'effectuer des changements	s dans le code en confiance.
Les TA permettent de montrer qu'un programn	ne fonctionne comme attendu.
Les TA permettent de limiter le nombre de bug	s.
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



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

Les TA permettent d'effectuer des changements dans le code en confiance.

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

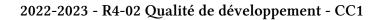
Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies

Question 3 🌲

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 <b>Répartition des tests</b> - 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.



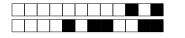
$\square 2 \ \square 2$	
	← Veuillez coder votre numéro
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps Aucun document ni dispositif électronique n'est au Le symbole "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 répon re-dessiner la case à cocher.	atorisé. 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
Question 1 4 Tests unitaires vs tests d'intégration	- Cochez les assertions vraies
Les tests unitaires consomment en général moins	s de ressources que les tests d'intégration.
Les tests d'intégration vérifient le fonctionnemen	nt de "vrais" objets en interactions.
Les tests unitaires créent toujours les vrais collab	borateurs pour tester un objet.
Les tests d'intégration s'exécutent en général plu	is rapidement que les tests unitaires.
Question 2 ♣ TDD - Cochez les assertions vraies	
En TDD, on code tous les tests d'une classe avan	it 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'écritu	
Le refactoring consiste à remanier le code d'un p	
	loppeur doit entreprendre après qu'un test échoue.
Question 3 Répartition des tests - Dans une appli la plus pertinente	ication, 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	tests unitaires.
80% de tests unitaires, 15% de tests d'intégration,	, 5% de tests UI.



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.



$\bigcirc$ Veuillez coder votre numéro
d'étudiant ci-contre, et écrire votre nom
444444 dans la case ci-dessous.
555555Nom 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 d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
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 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.
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.  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.
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.  Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
<ul> <li>Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.</li> <li>Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.</li> <li>Les tests unitaires consomment en général moins de ressources que les tests d'intégration.</li> <li>Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.</li> <li>Question 2 ♣ TDD - Cochez les assertions vraies</li> </ul>
<ul> <li>Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.</li> <li>Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.</li> <li>Les tests unitaires consomment en général moins de ressources que les tests d'intégration.</li> <li>Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.</li> <li>Question 2 ♣ TDD - Cochez les assertions vraies</li> <li>Le TDD améliore la conception d'un programme.</li> </ul>
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.  Les tests d'intégration s'exécutent en général plus 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.
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.  Les tests d'intégration s'exécutent en général plus 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 programme.
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.  Les tests d'intégration s'exécutent en général plus 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 programme.  En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
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.  Les tests d'intégration s'exécutent en général plus 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 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.  Question 3 Répartition des tests - Dans une application, quelle répartition de quantités de tests est
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.  Les tests d'intégration s'exécutent en général plus 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 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.  Question 3 Répartition des tests - Dans une application, quelle répartition de quantités de tests est la plus pertinente



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.



	$0  \boxed{0}  \boxed{0}  \boxed{0}$	$\square 0 \square 0 \square 0$	
<u> </u>	1		
	$2 \square 2 \square 2 \square 2$		
	$3 \square 3 \square 3 \square 3$	$\square_3$ $\square_3$ $\square_3$	← Veuillez coder votre numéro
	1		d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
			dans la case ci-dessous.
	) [] 5 [] 5 [] 5	555	Nom et prénom :
	6 6 6		
<u></u>	7777	777	
88	888	888	
9	9	9 9 9	
Aucun docu Le symbole Les réponses prise en compte	ment ni dispositif o "trèfle" sur une quo s doivent être fourr e dans la notation. s la notation. Pour	nies uniquement su Les cases doivent é	
Question 1 ૈ	TDD - Cochez le	es assertions vraies	
Le refacto	ring est une étape	du TDD qu'un dév	reloppeur doit entreprendre après qu'un test échoue.
En TDD,	on code tous les te	sts d'une classe ava	ant de coder toutes les méthodes de la classe.
En TDD,	dès qu'un test pass	e, on passe à l'écri	ture d'un nouveau test.
Le TDD a	méliore la concept	ion d'un programn	ne.
Le refacto	ring consiste à ren	nanier le code d'un	programme.
Question 2 ૈ	Tests unitaires v	s tests d'intégration	on - Cochez les assertions vraies
Les tests ı	unitaires créent tou	ijours les vrais coll	aborateurs pour tester un objet.
Les tests ı	unitaires consomm	ent en général moi	ins de ressources que les tests d'intégration.
Les tests o	d'intégration vérifi	ent le fonctionnem	ent de "vrais" objets en interactions.
Les tests of	l'intégration s'exé	cutent en général p	olus rapidement que les tests unitaires.
Question 3 🌲	Bénéfices et limi	ites des tests auton	natisés (TA) - Cochez les assertions vraies
Les TA pe	rmettent d'effectu	er des changement	s dans le code en confiance.
Les TA pe	rmettent de limite	r le nombre de bug	S.
Les TA pe	euvent contenir des	s bugs.	
Les TA pe	ermettent de montr	er qu'un programi	ne 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

### 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). 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.

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.

Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies

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.

Les TA permettent d'effectuer des changements dans le code en confiance.

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

Question 3 🌲

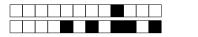
Les TA peuvent contenir des bugs.

Les TA permettent de limiter le nombre de bugs.

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.

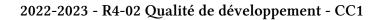


3 $3$ $3$ $3$ $3$ $3$ $3$ $3$ $3$ $3$
d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps).  Aucun document ni dispositif électronique n'est autorisé.  Le symbole "trèfle" sur une question indique qu'il peut être attendu plus d'une réponse correcte.  Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne ser prise en compte dans la notation. Les cases doivent être entièrement coloriées en noires pour être prise en compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un correcteur blanc, san re-dessiner la case à cocher.
Question 1 A 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.
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
Question 2 - Cochez les assertions vraies
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.
Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
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 limiter le nombre de bugs.
Les TA peuvent contenir des bugs.
Les TA permettent de montrer qu'un programme fonctionne comme attendu.
Question 4 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 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

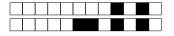
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
En TDD, la couverture du code par les tests est nécéssairement de 100%.
Dans un projet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



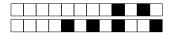
0       0	← Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.  Nom et prénom:
prise en compte dans la notation. Les cases doivent	autorisé.
Ouestion 1 ♣ Bénéfices et limites des tests auto  Les TA peuvent contenir des bugs.  Les TA permettent de limiter le nombre de bug  Les TA permettent de montrer qu'un program  Les TA permettent d'effectuer des changemen	ame fonctionne comme attendu.
Ouestion 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 d'intégration vérifient le fonctionnes de la colonne de la test d'intégration vérifient le fonctionnes de la colonne de la test d'intégration vérifient le fonctionnes de la colonne de la colo	plus rapidement que les tests unitaires. pins de ressources que les tests d'intégration. llaborateurs pour tester un objet.
Question 3  Code coverage - Cochez les assert  La couverture du code par les tests permet de  Dans un projet agile, l'objectif de couverture définition de fini.	
<ul><li>En TDD, la couverture du code par les tests es</li><li>La couverture du code par les tests permet de sont invoquées par les tests.</li></ul>	st nécéssairement de 100%. 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



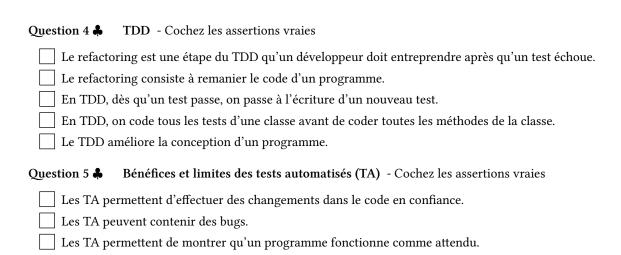
	← Veuillez coder votre numéro	
$\square_4 \square_4 \square_4 \square_4 \square_4 \square_4 \square_4 \square_4 \square_4$	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.	
	uans la case ci-uessous.	
	Nom et prénom :	
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document 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.	eut 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 & Bénéfices et limites des tests automat:  Les TA permettent de montrer qu'un programme :	isés (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 111 peavent content des bugs.		
<b>Question 2 Répartition des tests</b> - Dans une applic la plus pertinente	ation, quelle répartition de quantités de tests est	
80% de tests unitaires, 15% de tests d'intégration, 5	5% de tests UI.	
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.	
Question 3 . TDD - Cochez les assertions vraies		
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.		
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 pro	ogramme.	
Question 4 🌲 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 consomment en général moins	de ressources que les tests d'intégration.	
Les tests unitaires créent toujours les vrais collabo	orateurs pour tester un objet.	
Les tests d'intégration vérifient le fonctionnement	de "vrais" objets en interactions.	



Question 5 ૈ	Code coverage	- Cochez les assertions vraies	,

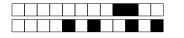
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
En TDD, la couverture du code par les tests est nécéssairement de 100%.
Dans un projet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

0       0	<ul> <li>✓ Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.</li> <li>Nom et prénom :</li> <li></li></ul>
prise en compte dans la notation. Les cases doivent	autorisé.
Question 1 ♣ Tests unitaires vs tests d'intégration  Les tests unitaires créent toujours les vrais col  Les tests d'intégration s'exécutent en général  Les tests unitaires consomment en général mod  Les tests d'intégration vérifient le fonctionner	llaborateurs pour tester un objet. plus rapidement que les tests unitaires. oins de ressources que les tests d'intégration.
Question 2 Répartition des tests - Dans une ap la plus pertinente  80% de tests unitaires, 15% de tests d'intégration, 5% de tests d'intégration, 5%	
Question 3  Code coverage - Cochez les asser	tions vraies
sont invoquées par les tests.	e savoir quelles classes, méthodes et lignes de codes
<ul><li>En TDD, la couverture du code par les tests es</li><li>Dans un projet agile, l'objectif de couverture définition de fini.</li></ul>	de code par les tests est un bon critère à ajouter à la
La couverture du code par les tests permet de	mesurer objectivement la qualité des tests.

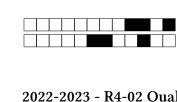


Les TA permettent de limiter le nombre de bugs.

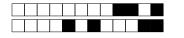
$\begin{array}{cccccccccccccccccccccccccccccccccccc$
$\bigcirc$ Veuillez coder votre numéro
d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
Nom et prénom :
Durée totale : 51 mn (1h08 mn pour les tiers temps).  Aucun document ni dispositif électronique 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 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 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 - 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.



4444444	Veuillez coder votre numéro diant ci-contre, et écrire votre nom la case ci-dessous. m 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  Les réponses doivent être fournies uniquement sur les feuill  prise en compte dans la notation. Les cases doivent être entièn  en compte dans la notation. Pour modifier une réponse, vous  re-dessiner la case à cocher.	les d'énoncé. Aucune autre réponse ne sera rement coloriées en noires pour être prises
Question 1 ♣ Code coverage - Cochez les assertions vraie  Dans un projet agile, l'objectif de couverture de code pa définition de fini.	
En TDD, la couverture du code par les tests est nécéssair  La couverture du code par les tests permet de mesurer o  La couverture du code par les tests permet de savoir qu sont invoquées par les tests.	bjectivement la qualité des tests.
Question 2 ♣ Bénéfices et limites des tests automatisés (T	'A) - Cochez les assertions vraies
Les TA permettent d'effectuer des changements dans le Les TA permettent de limiter le nombre de bugs.  Les TA peuvent contenir des bugs.  Les TA permettent de montrer qu'un programme fonction	
Question 3 4 Tests unitaires vs tests d'intégration - Coch	nez les assertions vraies
Les tests d'intégration vérifient le fonctionnement de "vi Les tests unitaires créent toujours les vrais collaborateur Les tests unitaires consomment en général moins de ress Les tests d'intégration s'exécutent en général plus rapide	rs pour tester un objet. sources que les tests d'intégration.



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.

# 2022-2023 - R4-02 Qualité de développement - CC1 $eg_1 eg_1 eg_$

 $\square_2$   $\square_2$   $\square_2$   $\square_2$   $\square_2$   $\square_2$ 

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

Durée totale : 51 mn (1h08 mn pour les tiers temps).

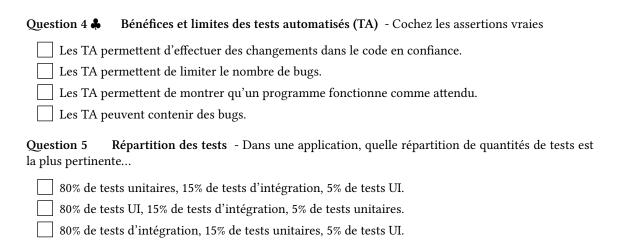
définition de fini.

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 integration - Cochez les assertions vraies
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.
Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
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 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.
Le TDD améliore la conception d'un programme.
Question 3 & 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



#### 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). 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.

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

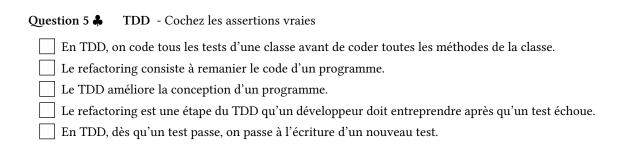
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.

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

Question 4

la plus pertinente...

Répartition des tests - Dans une application, quelle répartition de quantités de tests est



## 

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.

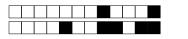
Question 1  Code coverage - Cochez les assertions vraies
La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
En TDD, la couverture du code par les tests est nécéssairement de 100%.
La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de 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 définition de fini.
Question 2 4 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 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 échou
Le TDD améliore la conception d'un programme.
Question 3 - 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 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	<b>Répartition des tests</b> - 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.



$\begin{array}{c ccccccccccccccccccccccccccccccccccc$		
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.		
$\bigcirc$ Nom et prénom :		
8 8 8 8 8 8 8		
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.		
re-dessiller la case à cocher.		
Question 1 & 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 consomment en général moins de ressources que les tests d'intégration.		
Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.		
Question 2 & 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 3 & 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 4 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 UI, 15% de tests d'intégration, 5% de tests unitaires.		

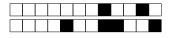


Ouestion 5 🐣	TDD -	Cochez les	assertions	vraies

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.



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.		
Question 1 ♣ TDD - Cochez les assertions vraie  Le refactoring consiste à remanier le code d'un  Le refactoring est une étape du TDD qu'un dév  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	n programme. veloppeur doit entreprendre après qu'un test échoue. rant de coder toutes les méthodes de la classe. me.		
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 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 3 Répartition des tests - Dans une ap la plus pertinente  80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitair 80% de tests UI, 15% de tests d'intégration, 5%	res, 5% de tests UI.		



Question 4 4 Code cov	erage - Cochez les assertions vraies
Dans un projet agile, définition de fini.	l'objectif de couverture de code par les tests est un bon critère à ajouter à l
La couverture du coo sont invoquées par le	de par les tests permet de savoir quelles classes, méthodes et lignes de code es tests.
La couverture du cod	e par les tests permet de mesurer objectivement la qualité des tests.
En TDD, la couvertur	re 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 d'	effectuer des changements dans le code en confiance.
Les TA permettent de	e limiter le nombre de bugs.
Les TA peuvent cont	enir des bugs.
Les TA permettent de	e montrer qu'un programme fonctionne comme attendu.

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.		
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 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 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 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.			
	cions vraies 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 la En TDD, la couverture du code par les tests es La couverture du code par les tests permet de 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	<b>Répartition des tests</b> - 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.

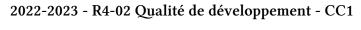


0       0	← Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.  Nom et prénom:		
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est autor Le symbole "trèfle" sur une question indique qu'il per Les réponses doivent 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. s feuilles d'énoncé. Aucune autre réponse ne sera e entièrement coloriées en noires pour être prises		
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 applic la plus pertinente  80% de tests UI, 15% de tests d'intégration, 5% de t 80% de tests d'intégration, 15% de tests unitaires, 5 80% de tests unitaires, 15% de tests d'intégration, 5	5% de tests UI.		

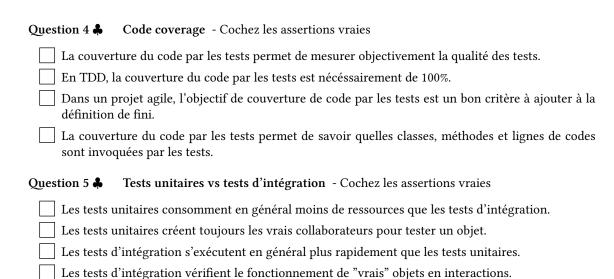


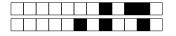
Question 5 ૈ	Code coverage	- Cochez l	es assertions	vraies

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

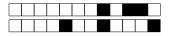


	0 0 0	
	]1	
	$]2 \square 2 \square 2$	
3 3 3 3 3	$3 \square 3 \square 3$	← Veuillez coder votre numéro
	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	]5	
		Nom et prénom :
	]	
	]8	
	]9	
Les réponses doivent être fournie prise en compte dans la notation. Le	ectronique n'est auto tion indique qu'il pe s uniquement sur les s cases doivent être	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 e, vous pouvez utiliser un correcteur blanc, sans
Question 1 A TDD - Cochez les a	assertions vraies	
Le refactoring est une étape du	ı TDD qu'un déveloj	ppeur 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	n d'un programme.	
En TDD, on code tous les tests	d'une classe avant d	de coder toutes les méthodes de la classe.
Le refactoring consiste à remai	nier le code d'un pro	ogramme.
Question 2 Répartition des tests la plus pertinente	s - Dans une applica	ation, quelle répartition de quantités de tests est
80% de tests unitaires, 15% de t	tests d'intégration, 5	% de tests UI.
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.
Question 3 🌲 Bénéfices et limites	s des tests automati	sés (TA) - Cochez les assertions vraies
Les TA permettent de montrer	qu'un programme f	onctionne comme attendu.
Les TA peuvent contenir des b	ugs.	
Les TA permettent de limiter le	e nombre de bugs.	
Les TA permettent d'effectuer	des changements da	ins le code en confiance.



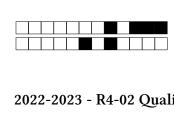


1       1	← Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.    Nom et prénom :			
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est auto Le symbole "trèfle" sur une question indique qu'il pe Les réponses 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.	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 ♣ 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 ♣ Tests unitaires vs tests d'intégration  Les tests unitaires consomment en général moins les tests unitaires créent toujours les vrais collaboration les tests d'intégration s'exécutent en général plus Les tests d'intégration vérifient le fonctionnement	de ressources que les tests d'intégration. orateurs pour tester un objet. rapidement que les tests unitaires. 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	s vraies			
<ul> <li>En TDD, la couverture du code par les tests est né</li> <li>La couverture du code par les tests permet de sav sont invoquées par les tests.</li> <li>La couverture du code par les tests permet de mes</li> <li>Dans un projet agile, l'objectif de couverture de c définition de fini.</li> </ul>	voir quelles classes, méthodes et lignes de codes urer objectivement la qualité des tests.			



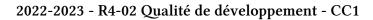
Ouestion 5 🌲	TDD	- Cochez les	assertions	vraies

_	
	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.



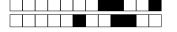
$\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$				
	← Veuillez coder votre numéro			
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.			
	dalis la case ci-dessous.			
	Nom et prénom :			
Durée totale : 51 mn (1h08 mn pour les tiers temps).  Aucun 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 & 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 n	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 me	esurer objectivement la qualité des tests.			
Question 2 . TDD - Cochez les assertions vraies				
Le TDD améliore la conception d'un programme	<u>.</u>			
Le refactoring est une étape du TDD qu'un dével	loppeur 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 p	orogramme.			
Question 3 4 Tests unitaires vs tests d'intégration	a - Cochez les assertions vraies			
Les tests unitaires créent toujours les vrais collal	borateurs pour tester un objet.			
Les tests unitaires consomment en général moin	s de ressources que les tests d'intégration.			
Les tests d'intégration s'exécutent en général plu	us rapidement que les tests unitaires.			
Les tests d'intégration vérifient le fonctionneme	nt de "vrais" objets en interactions.			

Question 4 la plus pertine	<b>Répartition des tests</b> - 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 t	ests 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.

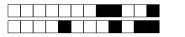


0       0				
Durée totale : 51 mn (1h08 mn pour les tiers temps Aucun document ni dispositif électronique n'est au Le symbole "trèfle" sur une question indique qu'il p Les réponses doivent être fournies uniquement sur l prise en compte dans la notation. Les cases doivent être en compte dans la notation. Pour modifier une répon re-dessiner la case à cocher.	ntorisé. 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 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 . TDD - Cochez les assertions vraies				
<ul> <li>□ Le refactoring consiste à remanier le code d'un programme.</li> <li>□ Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.</li> <li>□ En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.</li> <li>□ Le TDD améliore la conception d'un programme.</li> <li>□ En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.</li> </ul>				
Question 3 & Code coverage - Cochez les assertion	ns vraies			
En TDD, la couverture du code par les tests est n	nécéssairement 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			
La couverture du code par les tests permet de sa sont invoquées par les tests.	avoir quelles classes, méthodes et lignes de codes			
La couverture du code par les tests permet de me	esurer 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	initaires créent toujours les vrais collaborateurs pour tester un objet.
Les tests o	



	← Veuillez coder votre numéro			
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.			
	Nom et prénom :			
Durás totals : 51 mm (1h00 mm nour les tiers tomas)				
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	eut être attendu plus d'une réponse correcte.			
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.	,			
Question 1 & Bénéfices et limites des tests automat	isés (TA) - Cochez les assertions vraies			
Les TA permettent d'effectuer des changements d	ans 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	fonctionne comme attendu.			
Question 2 & Code coverage - Cochez les assertion	s vraies			
La couverture du code par les tests permet de sa sont invoquées par les tests.	voir quelles classes, méthodes et lignes de codes			
	Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la			
Fn TDD la converture du code par les tests est né	céssairement de 100%			
En TDD, la couverture du code par les tests est né				
La couverture du code par les tests permet de mes	urer objectivement la qualité des tests.			
	urer objectivement la qualité des tests.			
<ul> <li>☐ La couverture du code par les tests permet de mes</li> <li>Question 3 ♣ Tests unitaires vs tests d'intégration</li> <li>☐ Les tests unitaires créent toujours les vrais collaboration</li> </ul>	urer objectivement la qualité des tests Cochez les assertions vraies orateurs pour tester un objet.			
<ul> <li>☐ La couverture du code par les tests permet de mes</li> <li>Question 3 ♣ Tests unitaires vs tests d'intégration</li> <li>☐ Les tests unitaires créent toujours les vrais collaboration</li> <li>☐ Les tests d'intégration vérifient le fonctionnement</li> </ul>	urer objectivement la qualité des tests.  - Cochez les assertions vraies  orateurs pour tester un objet.  t de "vrais" objets en interactions.			
<ul> <li>☐ La couverture du code par les tests permet de mes</li> <li>Question 3 ♣ Tests unitaires vs tests d'intégration</li> <li>☐ Les tests unitaires créent toujours les vrais collaboration</li> </ul>	urer objectivement la qualité des tests.  - Cochez les assertions vraies  orateurs pour tester un objet.  t de "vrais" objets en interactions.			
<ul> <li>☐ La couverture du code par les tests permet de mes</li> <li>Question 3 ♣ Tests unitaires vs tests d'intégration</li> <li>☐ Les tests unitaires créent toujours les vrais collaboration</li> <li>☐ Les tests d'intégration vérifient le fonctionnement</li> </ul>	rurer objectivement la qualité des tests.  - Cochez les assertions vraies prateurs pour tester un objet.  t de "vrais" objets en interactions. rapidement que les tests unitaires.			
Les tests d'intégration vérifient le fonctionnement  Les tests d'intégration vérifient le fonctionnement  Les tests d'intégration vérifient le général plus  Les tests d'intégration s'exécutent en général moins	rurer objectivement la qualité des tests.  - Cochez les assertions vraies prateurs pour tester un objet.  t de "vrais" objets en interactions. rapidement que les tests unitaires.			
Les tests d'intégration vérifient le fonctionnement Les tests d'intégration vérifient le fonctionnement Les tests d'intégration s'exécutent en général plus Les tests unitaires consomment en général moins Question 4 Répartition des tests - Dans une applie la plus pertinente □	urer objectivement la qualité des tests.  - Cochez les assertions vraies  orateurs pour tester un objet.  de "vrais" objets en interactions.  rapidement que les tests unitaires.  de ressources que les tests d'intégration.  eation, quelle répartition de quantités de tests est			
La couverture du code par les tests permet de mes  Question 3 ♣ Tests unitaires vs tests d'intégration  Les tests unitaires créent toujours les vrais collabe  Les tests d'intégration vérifient le fonctionnement  Les tests d'intégration s'exécutent en général plus  Les tests unitaires consomment en général moins  Question 4 Répartition des tests - Dans une applic la plus pertinente  80% de tests d'intégration, 15% de tests unitaires, s'exécutent en général moins	urer objectivement la qualité des tests.  - Cochez les assertions vraies prateurs pour tester un objet.  de "vrais" objets en interactions. rapidement que les tests unitaires. de ressources que les tests d'intégration. ration, quelle répartition de quantités de tests est			
Les tests d'intégration vérifient le fonctionnement Les tests d'intégration vérifient le fonctionnement Les tests d'intégration s'exécutent en général plus Les tests unitaires consomment en général moins Question 4 Répartition des tests - Dans une applie la plus pertinente □	urer objectivement la qualité des tests.  - Cochez les assertions vraies  orateurs pour tester un objet.  de "vrais" objets en interactions.  rapidement que les tests unitaires.  de ressources que les tests d'intégration.  ation, quelle répartition de quantités de tests est  6% de tests UI.  ests unitaires.			

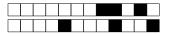


Ouestion 5 🌲	TDD	- Cochez les	assertions	vraies

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).  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.			

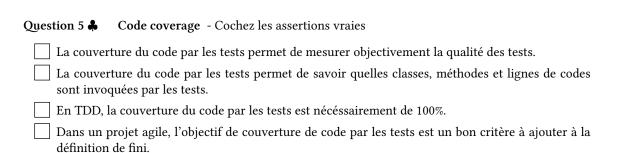


Question 5 🌲	Code coverage	- Cochez les	assertions	vraies

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%.

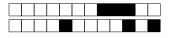


0       0	<ul> <li>✓ Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.</li> <li>Nom et prénom :</li> <li></li></ul>			
Le symbole "trèfle" sur une question indique qu'il pe Les réponses doivent être fournies uniquement sur les prise en compte dans la notation. Les cases doivent être	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			
Question 1 ♣ TDD - Cochez les assertions vraies				
<ul> <li>En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.</li> <li>Le refactoring consiste à remanier le code d'un programme.</li> <li>Le TDD améliore la conception d'un programme.</li> <li>En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.</li> <li>Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.</li> </ul>				
Question 2 Répartition des tests - Dans une application 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 🌲 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  Bénéfices et limites des tests automati	sés (TA) - Cochez les assertions vraies			
<ul> <li>Les TA peuvent contenir des bugs.</li> <li>Les TA permettent d'effectuer des changements dans le code en confiance.</li> <li>Les TA permettent de limiter le nombre de bugs.</li> </ul>				
Les TA permettent de montrer qu'un programme f	Conctionne comme attendu.			



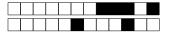


$\square$ 2				
	← Veuillez coder votre numéro			
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.			
	dans in case of dessous.			
	Nom et prénom :			
Durée totale : 51 mn (1h08 mn pour les tiers temps). Aucun document ni dispositif électronique n'est au Le symbole "trèfle" sur une question indique qu'il p Les réponses doivent être fournies uniquement sur l prise en compte dans la notation. Les cases doivent être no compte dans la notation. Pour modifier une répons re-dessiner la case à cocher.	torisé. Deut être attendu plus d'une réponse correcte. Les feuilles d'énoncé. Aucune autre réponse ne sera de entièrement coloriées en noires pour être prises			
Question 1 & Bénéfices et limites des tests automa	atisés (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	e fonctionne comme attendu.			
Les TA permettent d'effectuer des changements d	dans le code en confiance.			
Question 2 4 Tests unitaires vs tests d'intégration	- Cochez les assertions vraies			
Les tests unitaires créent toujours les vrais collab	porateurs pour tester un objet.			
Les tests unitaires consomment en général moins				
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.				
la plus pertinente	ication, quelle répartition de quantités de tests est			
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.				
	3% de tests Of.			
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 p	Le refactoring consiste à remanier le code d'un programme.			

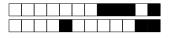


Question 5 ૈ	Code coverage	- Cochez les	assertions v	raies

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.

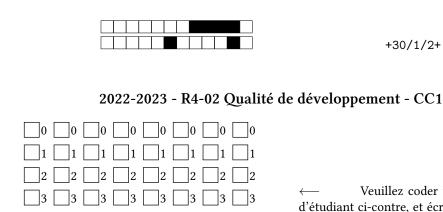


	✓ Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom			
	dans la case ci-dessous.			
	Nom et prénom :			
	Trom or pronom			
8 8 8 8 8 8 8				
Aucun document ni dispositif électronique n'est auto Le symbole "trèfle" sur une question indique qu'il per Les réponses doivent être fournies 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.	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 ♣ 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 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				
80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.				
Question 3 4 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 & Code coverage - Cochez les assertions	vraies			
En TDD, la couverture du code par les tests est néc	céssairement 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 mess	urer objectivement la qualité des tests.			



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.



<del></del>	Veuillez coder votre numéro
d'étudiant	ci-contre, et écrire votre non
dans la ca	se ci-dessous.

N	O	ı	n	l	e	t	r	r	é	n	10	)1	n	ì	:											
							•			•			•			•		•			•		•	•		
٠							•																			

Durée totale : 51 mn (1h08 mn pour les tiers temps).

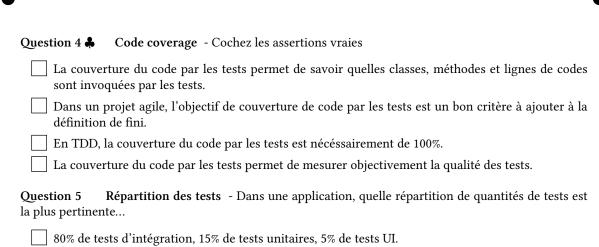
Aucun document ni dispositif électronique n'est autorisé.

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.

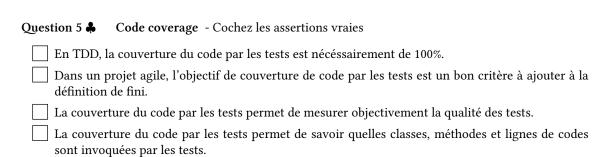
Question 1 & Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les TA permettent de limiter le nombre de bugs.
Les TA permettent de montrer qu'un programme fonctionne comme attendu.
Les TA permettent d'effectuer des changements dans le code en confiance.
Les TA peuvent contenir des bugs.
Question 2 - 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
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.

Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test
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.
Question 3 . Tests unitaires vs tests d'intégration - Cochez les assertions vraies
$\sim$ .
Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.



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.

$\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$	
	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	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 au 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 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 UI, 15% de tests d'intégration, 5% de	tests 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 collab	orateurs pour tester un objet.
Les tests d'intégration vérifient le fonctionnemer	nt de "vrais" objets en interactions.
Les tests d'intégration s'exécutent en général plu	s 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 automa	tisés (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	lans le code en confiance.
Les TA permettent de limiter le nombre de bugs.	
Question 4 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.
En TDD, dès qu'un test passe, on passe à l'écritur	e d'un nouveau test.
Le refactoring est une étape du TDD qu'un dével	oppeur doit entreprendre après qu'un test échoue.
Le TDD améliore la conception d'un programme.	
Le refactoring consiste à remanier le code d'un p	rogramme.



# 2022-2023 - R4-02 Qualité de développement - CC1 $\boxed{\phantom{a}}$ $\boxed{\phantom{a}}$

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

ı	V	C	1	n	L	e	t	p	r	é	n	ıc	)1	n	1	:									

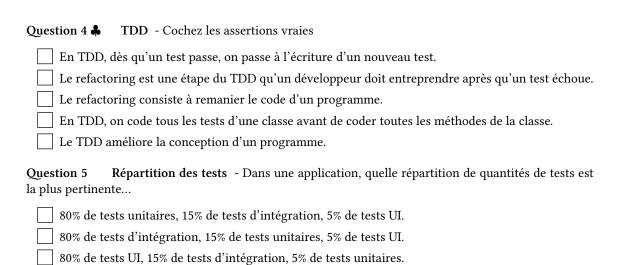
Durée totale : 51 mn (1h08 mn pour les tiers temps).

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 🌲 🛚 Bé	néfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les TA peuve	nt contenir des bugs.
Les TA perme	ttent de montrer qu'un programme fonctionne comme attendu.
Les TA perme	ttent de limiter le nombre de bugs.
Les TA perme	ttent d'effectuer des changements dans le code en confiance.
Question 2 4 Te	sts unitaires vs tests d'intégration - Cochez les assertions vraies
Les tests unita	ires consomment en général moins de ressources que les tests d'intégration.
Les tests unita	ires créent toujours les vrais collaborateurs pour tester un objet.
Les tests d'int	égration s'exécutent en général plus rapidement que les tests unitaires.
Les tests d'int	égration vérifient le fonctionnement de "vrais" objets en interactions.
Question 3 A Co	de coverage - Cochez les assertions vraies
En TDD, la co	uverture du code par les tests est nécéssairement de 100%.
	du code par les tests permet de savoir quelles classes, méthodes et lignes de codes s par les tests.
Dans un proje définition de f	et agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la îni.
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). 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. Les TA permettent de limiter le nombre de bugs. Les TA permettent de montrer qu'un programme fonctionne comme attendu.

Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la

La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes

La couverture du code par les tests permet de mesurer objectivement la qualité des tests.

Code coverage - Cochez les assertions vraies

En TDD, la couverture du code par les tests est nécéssairement de 100%.

Question 3 🐥

définition de fini.

sont invoquées par les tests.

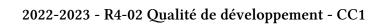
Question 4 la plus pertine	<b>Répartition des tests</b> - Dans une application, quelle répartition de quantités de tests es ente
80% de t	rests UI, 15% de tests d'intégration, 5% de tests unitaires.
80% de t	ests unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de 1	ests d'intégration, 15% 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.



	0
	1
	2
33333333	Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	Nom et prénom :
	8
	9
Les réponses doivent être fournies uniquer prise en compte dans la notation. Les cases de	
Question 1 & Code coverage - Cochez les	
Dans un projet agile, l'objectif de couve définition de fini.	erture de code par les tests est un bon critère à ajouter à la
En TDD, la couverture du code par les t	ests est nécéssairement de 100%.
La couverture du code par les tests per sont invoquées par les tests.	rmet de savoir quelles classes, méthodes et lignes de codes
La couverture du code par les tests perm	net de mesurer objectivement la qualité des tests.
Question 2 4 Tests unitaires vs tests d'in	tégration - Cochez les assertions vraies
Les tests d'intégration vérifient le fonct	ionnement de "vrais" objets en interactions.
Les tests unitaires consomment en géné	éral moins de ressources que les tests d'intégration.
Les tests unitaires créent toujours les vi	rais collaborateurs pour tester un objet.
Les tests d'intégration s'exécutent en gé	énéral plus rapidement que les tests unitaires.
Question 3 🌲 Bénéfices et limites des test	es 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 pr	ogramme fonctionne comme attendu.
Les TA permettent d'effectuer des chan	gements 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	← 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  Les tests unitaires créent toujours les vrais col  Les tests d'intégration vérifient le fonctionner  Les tests unitaires consomment en général mo  Les tests d'intégration s'exécutent en général	llaborateurs pour tester un objet. nent de "vrais" objets en interactions. oins de ressources que les tests d'intégration.	
Question 2 & Code coverage - Cochez les assert	tions vraies	
En TDD, la couverture du code par les tests es	et nécéssairement de 100%.	
La couverture du code par les tests permet de sont invoquées par les tests.	e savoir quelles classes, méthodes et lignes de codes	
<ul><li>La couverture du code par les tests permet de</li><li>Dans un projet agile, l'objectif de couverture définition de fini.</li></ul>	mesurer objectivement la qualité des tests. de code par les tests est un bon critère à ajouter à la	
Question 3 . TDD - Cochez les assertions vraie	s	
En TDD, dès qu'un test passe, on passe à l'écr	iture d'un nouveau test.	
Le refactoring consiste à remanier le code d'u	n programme.	
Le refactoring est une étape du TDD qu'un dé	veloppeur doit entreprendre après qu'un test échoue.	
En TDD, on code tous les tests d'une classe av	rant de coder toutes les méthodes de la classe.	
Le TDD améliore la conception d'un program	me.	

Question 4 la plus pertine	<b>Répartition des tests</b> - 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.



$\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$	
	← Veuillez coder votre numéro
	d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous.
	Nom et prénom :
	Trom et prenom :
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 auto Le symbole "trèfle" sur une question indique qu'il pe 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.	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 ♣ Bénéfices et limites des tests automaticular la	
Question 2 Répartition des tests - Dans une applic 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	
80% de tests UI, 15% de tests d'intégration, 5% de t	
Question 3 . Tests unitaires vs tests d'intégration	
Les tests unitaires créent toujours les vrais collabo	
Les tests d'intégration vérifient le fonctionnement	
Les tests unitaires consomment en général moins	
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 né	céssairement de 100%.
La couverture du code par les tests permet de sav sont invoquées par les tests.	voir quelles classes, méthodes et lignes de codes
La couverture du code par les tests permet de mes	urer objectivement la qualité des tests.
Dans un projet agile, l'objectif de couverture de co définition de fini.	ode par les tests est un bon critère à ajouter à la



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

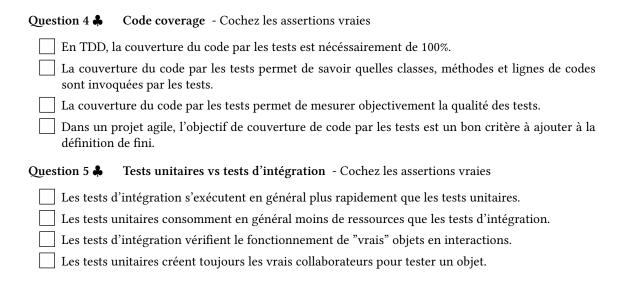
	Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.
	Le TDD améliore la conception d'un programme.
	Le refactoring consiste à remanier le code d'un programme.
	En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
Γ	En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

# 2022-2023 - R4-02 Qualité de développement - CC1 $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ $\square_2$ Veuillez coder votre numéro d'étudiant ci-contre, et écrire votre nom dans la case ci-dessous. 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 - 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, 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 d'un nouveau test.
Le TDD améliore la conception d'un programme.
Question 2 - 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.
Question 3 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 unitaires, 15% de tests d'intégration, 5% de tests UI.
80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.

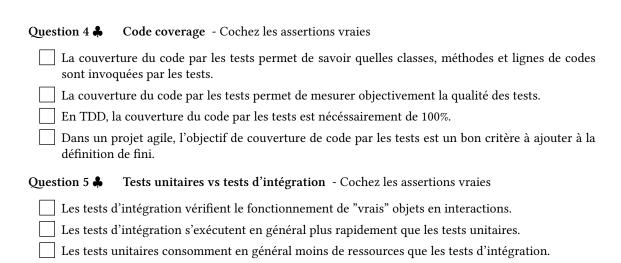


$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	
7       7	
Durée totale : 51 mn (1h08 mn pour les tiers temp Aucun document ni dispositif électronique n'est a Le symbole "trèfle" sur une question indique qu'i Les réponses doivent être fournies uniquement su prise en compte dans la notation. Les cases doivent é en compte dans la notation. Pour modifier une répo re-dessiner la case à cocher.	autorisé. l 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
Ouestion 1 ♣ Tests unitaires vs tests d'intégration  Les tests unitaires consomment en général mon  Les tests d'intégration s'exécutent en général proposition de la test d'intégration vérifient le fonctionnem  Les tests unitaires créent toujours les vrais colle	ins de ressources que les tests d'intégration. olus rapidement que les tests unitaires. nent de "vrais" objets en interactions.
la plus pertinente  80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaire	es, 5% de tests UI.
Question 3 . TDD - Cochez les assertions vraies	
☐ En TDD, on code tous les tests d'une classe ava ☐ Le refactoring consiste à remanier le code d'un ☐ En TDD, dès qu'un test passe, on passe à l'écri ☐ Le TDD améliore la conception d'un programm	n programme. ture d'un nouveau test.
Le refactoring est une étape du TDD qu'un dév	reloppeur doit entreprendre après qu'un test échoue.



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 pr définition d	ojet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la le fini.
	ure du code par les tests permet de savoir quelles classes, méthodes et lignes de codes nées par les tests.
Question 5 🌲	Bénéfices et limites des tests automatisés (TA) - Cochez les assertions vraies
Les TA peu	vent contenir des bugs.
Les TA per	mettent d'effectuer des changements dans le code en confiance.
Les TA per	mettent de montrer qu'un programme fonctionne comme attendu.
Les TA per	mettent de limiter le nombre de bugs.

	autorisé. peut être attendu plus d'une réponse correcte. r les feuilles d'énoncé. Aucune autre réponse ne sera etre entièrement coloriées en noires pour être prises
Question 1 ♣       TDD - Cochez les assertions vraies         □       Le refactoring est une étape du TDD qu'un dévont         □       Le refactoring consiste à remanier le code d'un         □       En TDD, dès qu'un test passe, on passe à l'écrit         □       Le TDD améliore la conception d'un programm         □       En TDD, on code tous les tests d'une classe avantes	eloppeur doit entreprendre après qu'un test échoue. programme. cure d'un nouveau test. ne.
Question 2 ♣ Bénéfices et limites des tests auton         ☐ Les TA peuvent contenir des bugs.         ☐ Les TA permettent de montrer qu'un programm         ☐ Les TA permettent de limiter le nombre de bug         ☐ Les TA permettent d'effectuer des changements	s.
Question 3 Répartition des tests - Dans une appara plus pertinente  80% de tests unitaires, 15% de tests d'intégration 80% de tests d'intégration, 15% de tests unitaire 80% de tests UI, 15% de tests d'intégration, 5% de	es, 5% de tests UI.
00% de tests 01, 13% de tests d'integration, 3% d	ic tests unitanes.



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



□ 0       □ 0	votre numéro rire votre nom
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 re Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune prise en compte dans la notation. Les cases doivent être entièrement coloriées en nen compte dans la notation. Pour modifier une réponse, vous pouvez utiliser un cre-dessiner la case à cocher.	autre réponse ne sera oires pour être prises
Question 1 ♣ TDD - Cochez les assertions vraies         Le refactoring est une étape du TDD qu'un développeur doit entreprendre aprendre aprendre aprendre 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	
Question 2 ♣ Bénéfices et limites des tests automatisés (TA) - Cochez les asser         □ 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 3 ♣ Tests unitaires vs tests d'intégration - Cochez les assertions vrai  Les tests d'intégration s'exécutent en général plus rapidement que les tests un  Les tests unitaires créent toujours les vrais collaborateurs pour tester un obje  Les tests unitaires consomment en général moins de ressources que les tests d'intégration vérifient le fonctionnement de "vrais" objets en interac	nitaires. t. l'intégration.
Question 4 Répartition des tests - Dans une application, quelle répartition de la plus pertinente  80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.	quantités de tests est



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.		
	Nom et prénom :		
8 8 8 8 8 8 8			
Durée totale : 51 mn (1h08 mn pour les tiers temp	s).		
Aucun document ni dispositif électronique n'est a			
Le symbole "trèfle" sur une question indique qu'il			
Les reponses doivent être fournies uniquement sur prise en compte dans la notation. Les cases doivent êt	les feuilles d'énoncé. Aucune autre réponse ne sera		
en compte dans la notation. Pour modifier une répoi	<u>.</u> .		
re-dessiner la case à cocher.	•		
Question 1 & Bénéfices et limites des tests autom	natisés (TA) - Cochez les assertions vraies		
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	S.		
Les TA permettent de montrer qu'un programm	ne fonctionne comme attendu.		
Question 2 Répartition des tests - Dans une app la plus pertinente	olication, quelle répartition de quantités de tests est		
80% de tests unitaires, 15% de tests d'intégration	ı, 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 . TDD - Cochez les assertions vraies			
Le TDD améliore la conception d'un programm			
En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.			
	eloppeur 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'écrit	ure d'un nouveau test.		
Question 4 🌲 Tests unitaires vs tests d'intégration	n - 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 fonctionneme	ent de "vrais" objets en interactions.		
Les tests d'intégration s'exécutent en général pl	lus rapidement que les tests unitaires.		



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