+1/1/60+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.

Question 2 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.

Question 3 ♣ **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 plus 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 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

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

Question 5 ♣ **Code coverage** - Cochez les assertions vraies :

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

Question 6 ♣ **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.
- [ ] Le TDD améliore la conception d'un programme.
- [ ] 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.

Question 7 ♣ **Suite de tests** - Cochez les assertions vraies :

- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- [ ] Un test en erreur indique en général un bug dans le test.
- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- [ ] Un test en échec indique en général un bug dans le programme principal.

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 :

- ☐ 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.
- ☐ 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 n'est pas nécéssairement de 100%.

Question 2 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

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

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.

Question 4 ♣ **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 permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA peuvent contenir des bugs.

Question 5 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en échec indique en général un bug dans le programme principal.

Question 6 ♣ **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.
- ☐ 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.

Question 7 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.

+3/1/56+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- ☐ Les TA permettent de limiter le nombre de bugs.
- ☐ Les TA permettent d'effectuer des changements dans le code en confiance.
- ☐ Les TA peuvent contenir des bugs.
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.

Question 2 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.

Question 3 ♣ **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 n'est pas nécéssairement de 100%.

Question 4 ♣ **TDD** - Cochez les assertions vraies :

- [ ] Le TDD améliore la conception d'un programme.
- [ ] Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.
- [ ] 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.

Question 5 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- [ ] Un collaborateur simulé est communément appelé en anglais "mock object".
- [ ] Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- [ ] Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- [ ] On utilise des collaborateurs simulés en général dans les tests d'intégration.
- [ ] Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- [ ] Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.

Question 6 **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.

Question 7 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- [ ] Les tests unitaires consomment en général plus 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.

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

.....

.....

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.

Question 2 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.

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

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 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.

Question 5 ♣ Simuler des collaborateurs - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

Question 6 ♣ 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 s'assurer qu'un programme fonctionne comme attendu.

Question 7 ♣ Code coverage - Cochez les assertions vraies :

- ☐ En TDD, la couverture du code par les tests n'est pas 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.

+5/1/52+

2023-2024 - R4-02 Qualité de développement - CC1

☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0
☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1
☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2
☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3
☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4
☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5
☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6
☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7
☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8
☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.

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 peuvent contenir des bugs.
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA permettent de limiter le nombre de bugs.

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

Question 4 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- [ ] Les tests unitaires consomment en général plus 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 5 ♣ **TDD** - Cochez les assertions vraies :

- [ ] En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
- [ ] Le TDD améliore la conception d'un programme.
- [ ] En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- [ ] Le refactoring 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 6 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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 7 ♣ **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 n'est pas 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.

+6/1/50+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.

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

Question 4 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.

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.
- ☐ 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 n'est pas nécéssairement de 100%.
- ☐ La couverture du code par les tests permet de mesurer objectivement la qualité des tests.

Question 6 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA permettent d'effectuer des changements dans le code en confiance.

Question 7 ♣ **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 s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

2023-2024 - R4-02 Qualité de développement - CC1

☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0
☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1
☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2
☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3
☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4
☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5
☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6
☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7
☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8
☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.

Question 2 ♣ **Code coverage** - Cochez les assertions vraies :

- ☐ En TDD, la couverture du code par les tests n'est pas nécéssairement de 100%.
- ☐ 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.
- ☐ Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.

Question 3 ♣ **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 s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

Question 4 ♣ **TDD** - Cochez les assertions vraies :

- [ ] En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- [ ] Le TDD améliore la conception d'un programme.
- [ ] 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 refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.

Question 5 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

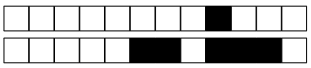
- [ ] 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.
- [ ] 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
- [ ] 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.

Question 6 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.
- [ ] Les TA permettent d'effectuer des changements dans le code en confiance.

Question 7 ♣ **Suite de tests** - Cochez les assertions vraies :

- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- [ ] Un test en erreur indique en général un bug dans le test.
- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- [ ] Un test en échec indique en général un bug dans le programme principal.



+8/1/46+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en échec indique en général un bug dans le programme principal.

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 s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA permettent de limiter le nombre de bugs.

Question 3 ♣ **Code coverage** - Cochez les assertions vraies :

- ☐ En TDD, la couverture du code par les tests n'est pas 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 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 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 consomment en général plus de ressources que les tests d'intégration.

Question 5 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

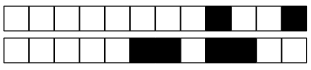
- [ ] Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- [ ] On utilise des collaborateurs simulés en général dans les tests d'intégration.
- [ ] Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- [ ] Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- [ ] Un collaborateur simulé est communément appelé en anglais "mock object".
- [ ] Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.

Question 6 ♣ **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.
- [ ] 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.

Question 7 **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.



+9/1/44+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 :

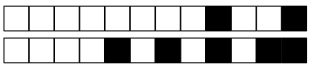
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA peuvent contenir des bugs.
- ☐ Les TA permettent de limiter le nombre de bugs.
- ☐ Les TA permettent d'effectuer des changements dans le code en confiance.

Question 2 ♣ **TDD** - Cochez les assertions vraies :

- ☐ En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- ☐ En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
- ☐ 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 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

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



Question 4 ♣ **Code coverage** - Cochez les 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 n'est pas nécéssairement de 100%.
- ☐ La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.

Question 5 ♣ **Suite de tests** - Cochez les assertions vraies :

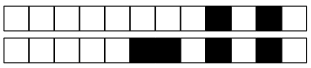
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en échec indique en général un bug dans le programme principal.

Question 6 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.

Question 7 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests unitaires consomment en général plus 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.



+10/1/42+

2023-2024 - R4-02 Qualité de développement - CC1

☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0
☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1
☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2
☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3
☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4
☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5
☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6
☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7
☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8
☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

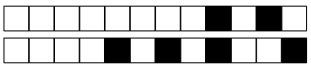
- ☐ 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.
- ☐ 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
- ☐ 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.

Question 2 ♣ **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 plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.



Question 4 ♣ **Suite de tests** - Cochez les assertions vraies :

- [ ] Un test en échec indique en général un bug dans le programme principal.
- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- [ ] Un test en erreur indique en général un bug dans le test.
- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.

Question 5 ♣ **Code coverage** - Cochez les assertions vraies :

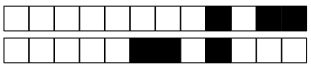
- [ ] Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
- [ ] La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
- [ ] En TDD, la couverture du code par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

Question 6 ♣ **TDD** - Cochez les assertions vraies :

- [ ] En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
- [ ] Le TDD améliore la conception d'un programme.
- [ ] En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- [ ] Le refactoring 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 7 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- [ ] Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- [ ] 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.



+11/1/40+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.

Question 2 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires consomment en général plus 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.

Question 3 ♣ **TDD** - Cochez les assertions vraies :

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

Question 4 ♣ **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.
- [ ] 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 n'est pas nécéssairement de 100%.

Question 5 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

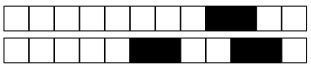
- [ ] Les TA permettent de limiter le nombre de bugs.
- [ ] Les TA permettent d'effectuer des changements dans le code en confiance.
- [ ] Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- [ ] Les TA peuvent contenir des bugs.

Question 6 ♣ **Suite de tests** - Cochez les assertions vraies :

- [ ] Un test en échec indique en général un bug dans le programme principal.
- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- [ ] Un test en erreur indique en général un bug dans le test.
- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.

Question 7 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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 UI, 15% de tests d'intégration, 5% de tests unitaires.



+12/1/38+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 :

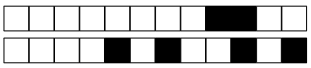
- ☐ Les TA permettent d'effectuer des changements dans le code en confiance.
- ☐ Les TA permettent de limiter le nombre de bugs.
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA peuvent contenir des bugs.

Question 2 ♣ **Code coverage** - Cochez les assertions vraies :

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

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".



Question 4 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.

Question 5 ♣ **TDD** - Cochez les assertions vraies :

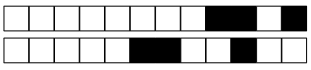
- ☐ En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- ☐ Le refactoring consiste à remanier le code d'un programme.
- ☐ Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.
- ☐ En TDD, dès qu'un test passe, on passe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

Question 6 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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 UI, 15% de tests d'intégration, 5% de tests unitaires.

Question 7 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests unitaires consomment en général plus 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.



+13/1/36+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 d'effectuer des changements dans le code en confiance.
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA peuvent contenir des bugs.
- ☐ Les TA permettent de limiter le nombre de bugs.

Question 2 ♣ **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.
- ☐ 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 n'est pas nécéssairement de 100%.

Question 3 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.

Question 4 ♣ **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.
- [ ] 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 est la plus pertinente :

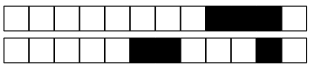
- [ ] 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
- [ ] 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.
- [ ] 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.

Question 6 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- [ ] Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- [ ] Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- [ ] Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- [ ] Un collaborateur simulé est communément appelé en anglais "mock object".
- [ ] Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- [ ] On utilise des collaborateurs simulés en général dans les tests d'intégration.

Question 7 ♣ **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 d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- [ ] Les tests unitaires consomment en général plus de ressources que les tests d'intégration.



+14/1/34+

2023-2024 - R4-02 Qualité de développement - CC1

☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0
☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1
☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2
☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3
☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4
☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5
☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6
☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7
☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8
☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 plus de ressources que les tests d'intégration.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

Question 2 ♣ **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 TDD améliore la conception d'un programme.
- ☐ Le refactoring consiste à remanier le code d'un programme.

Question 3 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.

Question 4 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- [ ] Les TA permettent de limiter le nombre de bugs.
- [ ] Les TA peuvent contenir des bugs.
- [ ] Les TA permettent d'effectuer des changements dans le code en confiance.
- [ ] Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.

Question 5 ♣ **Code coverage** - Cochez les assertions vraies :

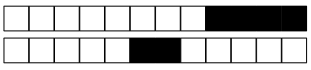
- [ ] En TDD, la couverture du code par les tests n'est pas nécéssairement de 100%.
- [ ] Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
- [ ] La couverture du code par les tests permet de savoir quelles classes, méthodes et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

Question 6 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- [ ] Un collaborateur simulé est communément appelé en anglais "mock object".
- [ ] Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- [ ] Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- [ ] On utilise des collaborateurs simulés en général dans les tests d'intégration.
- [ ] Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- [ ] Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.

Question 7 **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.



+15/1/32+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 d'effectuer des changements dans le code en confiance.
- ☐ Les TA permettent de limiter le nombre de bugs.
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA peuvent contenir des bugs.

Question 2 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

Question 3 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.

Question 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 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 5 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

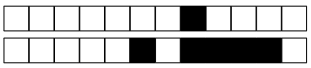
- [ ] Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- [ ] Un collaborateur simulé est communément appelé en anglais "mock object".
- [ ] Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- [ ] Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- [ ] On utilise des collaborateurs simulés en général dans les tests d'intégration.
- [ ] Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

Question 6 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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 7 ♣ **Code coverage** - Cochez les assertions vraies :

- [ ] En TDD, la couverture du code par les tests n'est pas 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.



+16/1/30+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

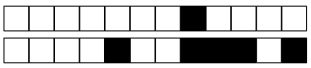
- ☐ 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
- ☐ 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
- ☐ 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.

Question 2 ♣ **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 TDD améliore la conception d'un programme.
- ☐ Le refactoring consiste à remanier le code d'un programme.

Question 3 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.



Question 4 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.

Question 5 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

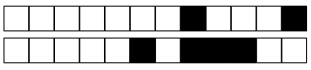
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

Question 6 ♣ **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.
- ☐ En TDD, la couverture du code par les tests n'est pas 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.

Question 7 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ 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.



+17/1/28+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 s'assurer 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.

Question 2 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

Question 3 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.

Question 4 ♣ **TDD** - Cochez les assertions vraies :

- [ ] En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- [ ] Le refactoring consiste à remanier le code d'un programme.
- [ ] 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 est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.

Question 5 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

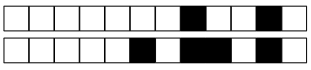
- [ ] Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- [ ] Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- [ ] Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
- [ ] Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.

Question 6 **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.

Question 7 ♣ **Code coverage** - Cochez les assertions vraies :

- [ ] En TDD, la couverture du code par les tests n'est pas 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.



+18/1/26+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 :

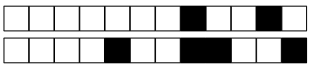
- ☐ Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.
- ☐ Le refactoring consiste à remanier le code d'un programme.
- ☐ En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
- ☐ En TDD, on code tous les tests d'une classe avant de coder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

Question 2 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.

Question 3 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- ☐ Les TA peuvent contenir des bugs.
- ☐ Les TA permettent de s'assurer 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.



Question 4 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.

Question 5 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

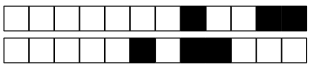
- ☐ 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
- ☐ 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
- ☐ 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.

Question 6 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.

Question 7 ♣ **Code coverage** - Cochez les 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.
- ☐ 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 n'est pas nécéssairement de 100%.



+19/1/24+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

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

Question 2 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".

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 consomment en général plus 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.

Question 4 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- [ ] Les TA peuvent contenir des bugs.
- [ ] Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- [ ] Les TA permettent d'effectuer des changements dans le code en confiance.
- [ ] Les TA permettent de limiter le nombre de bugs.

Question 5 ♣ **Code coverage** - Cochez les assertions vraies :

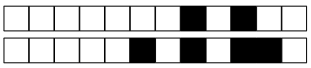
- [ ] Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
- [ ] En TDD, la couverture du code par les tests n'est pas nécéssairement de 100%.
- [ ] La couverture du code par les tests permet de savoir quelles classes, méthodes et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

Question 6 ♣ **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.
- [ ] 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.

Question 7 ♣ **Suite de tests** - Cochez les assertions vraies :

- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- [ ] Un test en échec indique en général un bug dans le programme principal.
- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- [ ] Un test en erreur indique en général un bug dans le test.



+20/1/22+

2023-2024 - R4-02 Qualité de développement - CC1

☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0
☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1
☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2
☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3
☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4
☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5
☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6
☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7
☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8
☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 :

- ☐ 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 n'est pas nécéssairement de 100%.
- ☐ 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.

Question 2 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Un test en échec indique en général un bug dans le programme principal.

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.

Question 4 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.

Question 5 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

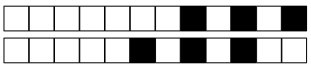
- ☐ 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.
- ☐ 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
- ☐ 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.

Question 6 ♣ **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 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.

Question 7 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA permettent d'effectuer des changements dans le code en confiance.



+21/1/20+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

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

Question 2 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

Question 4 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.

Question 5 ♣ **Suite de tests** - Cochez les assertions vraies :

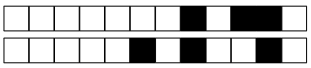
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.

Question 6 ♣ **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.
- ☐ 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.
- ☐ En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.

Question 7 ♣ **Code coverage** - Cochez les assertions vraies :

- ☐ Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
- ☐ En TDD, la couverture du code par les tests n'est pas 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.



+22/1/18+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

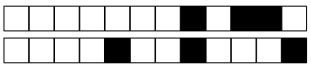
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en échec indique en général un bug dans le programme principal.

Question 2 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

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

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.



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.
- [ ] 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 refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.

Question 5 ♣ **Code coverage** - Cochez les assertions vraies :

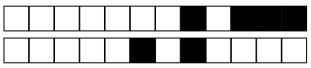
- [ ] La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.
- [ ] Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
- [ ] La couverture du code par les tests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

Question 6 ♣ **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 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 plus de ressources que les tests d'intégration.

Question 7 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.
- [ ] Les TA permettent de limiter le nombre de bugs.



+23/1/16+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Un test en échec indique en général un bug dans le programme principal.

Question 2 ♣ **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.
- ☐ 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 ♣ **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 peuvent contenir des bugs.
- ☐ Les TA permettent de limiter le nombre de bugs.
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.

Question 4 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires consomment en général plus 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.

Question 5 ♣ **Code coverage** - Cochez les assertions vraies :

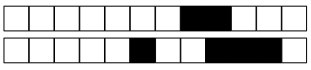
- ☐ La couverture du code par les tests permet de mesurer objectivement la qualité des tests.
- ☐ En TDD, la couverture du code par les tests n'est pas nécéssairement de 100%.
- ☐ La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les 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.

Question 6 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".

Question 7 **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.



+24/1/14+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

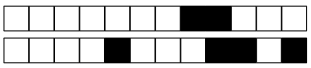
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en erreur indique en général un bug dans le test.

Question 2 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".

Question 3 ♣ **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.
- ☐ 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.
- ☐ Le refactoring consiste à remanier le code d'un programme.



Question 4 ♣ **Code coverage** - Cochez les assertions vraies :

- ☐ En TDD, la couverture du code par les tests n'est pas nécéssairement de 100%.
- ☐ 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.
- ☐ Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.

Question 5 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

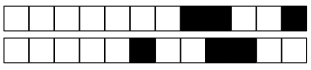
- ☐ Les TA permettent d'effectuer des changements dans le code en confiance.
- ☐ Les TA permettent de limiter le nombre de bugs.
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA peuvent contenir des bugs.

Question 6 ♣ **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 plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

Question 7 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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 UI, 15% de tests d'intégration, 5% de tests unitaires.



+25/1/12+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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, 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 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 consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.

Question 4 ♣ **Code coverage** - Cochez les assertions vraies :

- [ ] En TDD, la couverture du code par les tests n'est pas nécéssairement de 100%.
- [ ] 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.
- [ ] Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.

Question 5 ♣ **Suite de tests** - Cochez les assertions vraies :

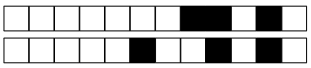
- [ ] Un test en échec indique en général un bug dans le programme principal.
- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- [ ] Un test en erreur indique en général un bug dans le test.
- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.

Question 6 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

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

Question 7 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- [ ] Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- [ ] 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.



+26/1/10+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

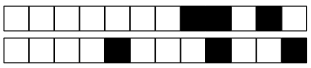
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.

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 s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA permettent de limiter le nombre de bugs.

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.



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

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

Question 5 ♣ **TDD** - Cochez les assertions vraies :

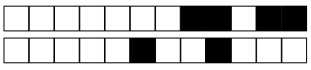
- ☐ En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.
- ☐ Le refactoring consiste à remanier le code d'un programme.
- ☐ En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- ☐ Le TDD améliore la conception d'un programme.
- ☐ Le refactoring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

Question 6 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.

Question 7 ♣ **Code coverage** - Cochez les assertions vraies :

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



2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- ☐ Les TA permettent de limiter le nombre de bugs.
- ☐ Les TA peuvent contenir des bugs.
- ☐ Les TA permettent d'effectuer des changements dans le code en confiance.
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.

Question 2 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".

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

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.
- [ ] 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.

Question 5 ♣ **Suite de tests** - Cochez les assertions vraies :

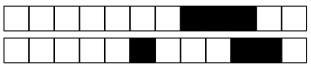
- [ ] Un test en erreur indique en général un bug dans le test.
- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- [ ] Un test en échec indique en général un bug dans le programme principal.
- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.

Question 6 ♣ **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 n'est pas nécéssairement de 100%.

Question 7 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- [ ] Les tests unitaires consomment en général plus 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.



+28/1/6+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

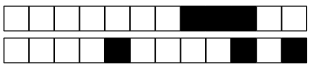
- ☐ 80% de tests d'intégration, 15% de tests unitaires, 5% de tests UI.
- ☐ 80% de tests UI, 15% de tests d'intégration, 5% de tests unitaires.
- ☐ 80% de tests unitaires, 15% de tests d'intégration, 5% de tests UI.

Question 2 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.

Question 3 ♣ **Code coverage** - Cochez les assertions vraies :

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



Question 4 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.

Question 5 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

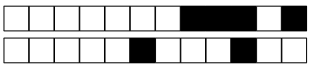
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.

Question 6 ♣ **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.
- ☐ En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- ☐ Le TDD améliore la conception d'un programme.
- ☐ Le refactoring consiste à remanier le code d'un programme.

Question 7 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.



2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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.
- ☐ 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 n'est pas nécéssairement de 100%.
- ☐ La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.

Question 2 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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 UI, 15% de tests d'intégration, 5% de tests unitaires.

Question 3 ♣ **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 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 plus de ressources que les tests d'intégration.

Question 4 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Un test en échec indique en général un bug dans le programme principal.

Question 5 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

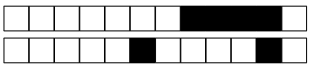
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.

Question 6 ♣ **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.
- ☐ 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.
- ☐ Le refactoring consiste à remanier le code d'un programme.

Question 7 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ 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.



+30/1/2+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.

Question 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.
- ☐ 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.
- ☐ Le refactoring consiste à remanier le code d'un programme.

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.

Question 4 ♣ **Suite de tests** - Cochez les assertions vraies :

- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- [ ] Un test en erreur indique en général un bug dans le test.
- [ ] Un test en échec indique en général un bug dans le programme principal.
- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.

Question 5 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

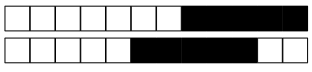
- [ ] Les TA permettent de limiter le nombre de bugs.
- [ ] Les TA peuvent contenir des bugs.
- [ ] Les TA permettent d'effectuer des changements dans le code en confiance.
- [ ] Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.

Question 6 **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 7 ♣ **Code coverage** - Cochez les assertions vraies :

- [ ] En TDD, la couverture du code par les tests n'est pas 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.



+31/1/60+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.

Question 2 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests unitaires consomment en général plus 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 application, quelle répartition de quantités de tests est la plus pertinente :

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

Question 4 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA permettent de limiter le nombre de bugs.
- ☐ Les TA peuvent contenir des bugs.

Question 5 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- [ ] Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- [ ] Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- [ ] On utilise des collaborateurs simulés en général dans les tests d'intégration.
- [ ] Un collaborateur simulé est communément appelé en anglais "mock object".
- [ ] Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- [ ] Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.

Question 6 ♣ **TDD** - Cochez les assertions vraies :

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

Question 7 ♣ **Code coverage** - Cochez les assertions vraies :

- [ ] Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
- [ ] En TDD, la couverture du code par les tests n'est pas nécéssairement de 100%.
- [ ] 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.

+32/1/58+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

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

Question 1 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.

Question 2 ♣ **TDD** - Cochez les assertions vraies :

- ☐ Le TDD améliore la conception d'un programme.
- ☐ Le refactoring est une étape du TDD qu'un développeur doit entreprendre après qu'un test échoue.
- ☐ 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.

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 de s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA permettent d'effectuer des changements dans le code en confiance.

Question 4 ♣ **Suite de tests** - Cochez les assertions vraies :

- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- [ ] Un test en échec indique en général un bug dans le programme principal.
- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- [ ] Un test en erreur indique en général un bug dans le test.

Question 5 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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 6 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- [ ] Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- [ ] Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- [ ] Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- [ ] Un collaborateur simulé est communément appelé en anglais "mock object".
- [ ] On utilise des collaborateurs simulés en général dans les tests d'intégration.
- [ ] Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

Question 7 ♣ **Code coverage** - Cochez les assertions vraies :

- [ ] Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
- [ ] En TDD, la couverture du code par les tests n'est pas nécéssairement de 100%.
- [ ] 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.

+33/1/56+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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, 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 2 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.

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 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 plus de ressources que les tests d'intégration.

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.
- ☐ En TDD, la couverture du code par les tests n'est pas 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.

Question 6 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

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

Question 7 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- ☐ 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.

+34/1/54+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

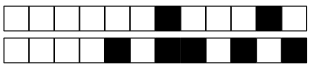
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.

Question 2 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est la plus pertinente :

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

Question 3 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests unitaires consomment en général plus 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 4 ♣ **Code coverage** - Cochez les assertions vraies :

- ☐ Dans un projet agile, l'objectif de couverture de code par les tests est un bon critère à ajouter à la définition de fini.
- ☐ En TDD, la couverture du code par les tests n'est pas nécéssairement de 100%.
- ☐ 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.

Question 5 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.

Question 6 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.

Question 7 ♣ **TDD** - Cochez les assertions vraies :

- ☐ 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.
- ☐ 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.

+35/1/52+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.

Question 2 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.

Question 3 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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 UI, 15% de tests d'intégration, 5% de tests unitaires.

Question 4 ♣ **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 n'est pas 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.

Question 5 ♣ **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 plus de ressources que les tests d'intégration.
- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.

Question 6 ♣ **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 7 ♣ **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 peuvent contenir des bugs.
- ☐ Les TA permettent de limiter le nombre de bugs.
- ☐ Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.

+36/1/50+

2023-2024 - R4-02 Qualité de développement - CC1

☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0
☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1
☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2
☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3
☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4
☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5
☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6
☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7
☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8
☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 n'est pas nécéssairement de 100%.
- ☐ 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.
- ☐ La couverture du code par les tests permet de savoir quelles classes, méthodes et lignes de codes sont invoquées par les tests.

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

Question 4 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

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

Question 5 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

Question 6 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Un test en échec indique en général un bug dans le programme principal.

Question 7 ♣ **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 s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA permettent d'effectuer des changements dans le code en confiance.

+37/1/48+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 s'assurer qu'un programme fonctionne comme attendu.
- ☐ 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.

Question 2 ♣ **Code coverage** - Cochez les assertions vraies :

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

Question 3 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

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 ♣ **Suite de tests** - Cochez les assertions vraies :

- [ ] Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- [ ] Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- [ ] Un test en erreur indique en général un bug dans le test.
- [ ] Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- [ ] Un test en échec indique en général un bug dans le programme principal.

Question 6 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- [ ] Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- [ ] Les tests unitaires consomment en général plus 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.

Question 7 **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.

+38/1/46+

2023-2024 - R4-02 Qualité de développement - CC1

☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0 ☐0
☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1
☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2 ☐2
☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3 ☐3
☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4 ☐4
☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5 ☐5
☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6 ☐6
☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7 ☐7
☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8 ☐8
☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9 ☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 n'est pas 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.

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 d'intégration s'exécutent en général plus rapidement que les tests unitaires.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.

Question 3 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en erreur indique en général un bug dans le test.

Question 4 ♣ **TDD** - Cochez les assertions vraies :

- [ ] En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- [ ] 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 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 5 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- [ ] Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- [ ] On utilise des collaborateurs simulés en général dans les tests d'intégration.
- [ ] Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- [ ] Un collaborateur simulé est communément appelé en anglais "mock object".
- [ ] Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- [ ] Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

Question 6 **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 7 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- [ ] Les TA peuvent contenir des bugs.
- [ ] Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- [ ] Les TA permettent d'effectuer des changements dans le code en confiance.
- [ ] Les TA permettent de limiter le nombre de bugs.

+39/1/44+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.

Question 2 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- ☐ Les tests d'intégration vérifient le fonctionnement de "vrais" objets en interactions.
- ☐ Les tests unitaires consomment en général plus de ressources que les tests d'intégration.
- ☐ Les tests unitaires créent toujours les vrais collaborateurs pour tester un objet.
- ☐ Les tests d'intégration s'exécutent en général plus rapidement que les tests unitaires.

Question 3 ♣ **Code coverage** - Cochez les assertions vraies :

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

Question 4 ♣ **TDD** - Cochez les assertions vraies :

- [ ] En TDD, on code tous les tests d'une classe avant de coder toutes les méthodes de la classe.
- [ ] 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.
- [ ] En TDD, dès qu'un test passe, on passe à l'écriture d'un nouveau test.

Question 5 ♣ **Bénéfices et limites des tests automatisés (TA)** - Cochez les assertions vraies :

- [ ] Les TA permettent de s'assurer qu'un programme fonctionne comme attendu.
- [ ] 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.

Question 6 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- [ ] Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.
- [ ] Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- [ ] Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- [ ] On utilise des collaborateurs simulés en général dans les tests d'intégration.
- [ ] Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- [ ] Un collaborateur simulé est communément appelé en anglais "mock object".

Question 7 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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.

+40/1/42+

2023-2024 - R4-02 Qualité de développement - CC1

☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8
☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9	☐9

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

Nom et prénom :

. . . . . .

. . . . . .

Durée totale : 42 mn (56 mn pour les tiers temps). Partie QCM sur 7 points.

Aucun document ni dispositif électronique n'est autorisé.

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

Les réponses doivent être fournies uniquement sur les feuilles d'énoncé. Aucune autre réponse ne sera prise en compte dans la notation. Les cases doivent être entièrement coloriées en noir 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 ♣ **Simuler des collaborateurs** - Cochez les assertions vraies :

- ☐ Simuler des collaborateurs dans un test permet en général de limiter le coût d'exécution d'un test.
- ☐ Il n'est pas possible de créer des collaborateurs simulés dans des tests en PHP.
- ☐ On utilise des collaborateurs simulés en général dans les tests d'intégration.
- ☐ Il existe des librairies pour créer des collaborateurs simulés dans à peu près tous les langages de programmation courants.
- ☐ Un collaborateur simulé est communément appelé en anglais "mock object".
- ☐ Il n'est pas toujours possible de créer un collaborateur simulé "à la main" pour une classe donnée.

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 s'assurer qu'un programme fonctionne comme attendu.
- ☐ Les TA permettent de limiter le nombre de bugs.

Question 3 ♣ **Suite de tests** - Cochez les assertions vraies :

- ☐ Les tests des cas d'erreur suffisent pour tester la robustesse d'un programme.
- ☐ Un test en échec indique en général un bug dans le programme principal.
- ☐ Un test en erreur indique en général un bug dans le test.
- ☐ Une bonne suite de tests est une suite de tests où les tests dépendent les uns des autres.
- ☐ Une bonne suite de tests teste explicitement uniquement l'interface publique de l'objet testé.

Question 4 ♣ **TDD** - Cochez les assertions vraies :

- [ ] 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 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.

Question 5 **Répartition des tests** - Dans une application, quelle répartition de quantités de tests est 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 UI, 15% de tests d'intégration, 5% de tests unitaires.

Question 6 ♣ **Tests unitaires vs tests d'intégration** - Cochez les assertions vraies :

- [ ] Les tests unitaires consomment en général plus 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 7 ♣ **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 n'est pas 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.