SAE : Dossier de tests unitaires

SAE : Calcul distribué sur un cluster de Raspberry	Version : 1.0
Document : Dossier de tests	Date : 03/10/2025 (debut)
Responsable de la rédaction : Valérian LAMBERT Clément BERARD	

Dossier de tests unitaires

1. Introduction

*Ce document décrit les **tests unitaires boîte blanche**, conçus selon la **méthode des chemins indépendants**, pour l'algorithme de recherche de nombres premiers, l'algorithme de Monte Carlo et l'intégrale de la loi normale (Simpson).*

2. Description des informations à enregistrer pour les tests

1. Campagne de test

Composant testé : Algorithme Prime Improved
Configuration logicielle : Pycharm/IntelliJ
Configuration matérielle : PC 8 Coeurs, 16 processeurs logiques
Conditions spécifiques : N/A
Responsable(s) de la campagne de test : Clément BERARD

Composant testé : Algorithme Monte Carlo Master Worker (threads)
Configuration logicielle : Pycharm/IntelliJ
Configuration matérielle : PC 8 Coeurs, 16 processeurs logiques
Conditions spécifiques : N/A
Responsable(s) de la campagne de test : Clément BERARD

Composant testé : Algorithme Monte Carlo Master Worker MPI
Configuration logicielle : Pycharm/IntelliJ
Configuration matérielle : PC 8 Coeurs, 16 processeurs logiques
Conditions spécifiques : N/A
Responsable(s) de la campagne de test : Clément BERARD

Composant testé : Algorithme Monte Carlo Master.py
Configuration logicielle : Pycharm/IntelliJ
Configuration matérielle : PC 8 Coeurs, 16 processeurs logiques
Conditions spécifiques : N/A
Responsable(s) de la campagne de test : Clément BERARD

Composant testé : Worker Socket Simpson (simpson(a, b, n))

Configuration logicielle : Pycharm/IntelliJ
Configuration matérielle : PC 8 Coeurs, 16 processeurs logiques
Conditions spécifiques : N/A
Responsable(s) de la campagne de test : Clément BERARD

2. Tests

Prime Improved.py

Identification du test : C1 : boucle interne non exécutée			Version : 1.0	
Description du test : Vérifier le comportement lorsque la boucle de division n'est jamais exécutée				
Ressources requises : Script Prime Improved				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Nombre inférieur à 2	Acteur: Fonction find_prime	Actions: Appeler la fonction avec start_number = 1, end_number = 2, step = 2	Résultats attendus: Boucle interne non exécutée, aucun test de division, primes = [1], test_count = 0

Identification du test : C2 : Nombre premier aucun diviseur trouvé	Version : 1.0
Description du test : Vérifier qu'un nombre premier est correctement identifié lorsque la boucle de division s'exécute sans trouver de diviseur.	
Ressources requises : Script Prime Improved	

Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Nombre premier impair	Acteur: Fonction find_prime	Actions: Appeler la fonction avec start_number = 7, end_number = 9, step = 2	Résultats attendus: Aucun diviseur trouvé, found_prime = True, primes = [7], test_count = 1

Identification du test : C3 : Nombre composé - diviseur trouvé			Version : 1.0	
Description du test : Vérifier l'arrêt anticipé de la boucle interne lorsqu'un diviseur est trouvé.				
Ressources requises : Script Prime Improved				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Nombre composé impair	Acteur: Fonction find_prime	Actions: Appeler la fonction avec start_number = 9, end_number = 11, step = 2	Résultats attendus: Diviseur détecté (9 % 3 = 0), sortie anticipée de la boucle, primes = [], test_count = 2

Identification du test : C4 : Plusieurs itérations de la boucle externe			Version : 1.0	
Description du test : Vérifier le bon fonctionnement de la boucle externe sur plusieurs candidats consécutifs.				
Ressources requises : Script Prime Improved				
Responsable : BERARD Clément				

Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Suite de nombres impairs	Acteur: Fonction find_prime	Actions: Appeler la fonction avec start_number = 3, end_number = 15, step = 2	Résultats attendus: Calcul répété correct, primes = [3, 5, 7, 11, 13]

Identification du test : C5 : Aucun candidat à tester			Version : 1.0	
Description du test : Vérifier le comportement de l'algorithme lorsque la boucle externe ne s'exécute pas.				
Ressources requises : Script Prime Improved				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Intervalle vide	Acteur: Fonction find_prime	Actions: Appeler la fonction avec start_number = 3, end_number = 15, step = 2	Résultats attendus: Calcul répété correct, primes = [3, 5, 7, 11, 13]

Monte Carlo, Master Worker (threads).java

Identification du test : C1 : Boucle non exécutée (Worker.call)			Version : 1.0	
Description du test : Vérifier le comportement du Worker lorsque le nombre d'itérations est nul.				
Ressources requises : JDK, classe Worker				
Responsable : BERARD Clément				
Etat initial: Script prêt à être	Cas testé: Nombre d'itérations égal à 0	Acteur: Classe Worker	Actions: Instancier un Worker avec	Résultats attendus: Boucle non

exécuté, aucun calcul en cours			numIterations = 0 et appeler call()	exécutée, résultat retourné = 0

Identification du test : C2 : Condition toujours vraie (Worker.call)			Version : 1.0	
Description du test : Vérifier le comptage lorsque les points générés sont dans le cercle.				
Ressources requises : JDK, classe Worker				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Itérations positives	Acteur: Classe Worker	Actions: Exécuter call() avec numIterations > 0	Résultats attendus: Résultat compris entre 0 et numIterations

Identification du test : C3 : Chemins mixtes vrai/faux (Worker.call)			Version : 1.0	
Description du test : Vérifier le fonctionnement normal avec plusieurs itérations.				
Ressources requises : JDK, classe Worker				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Grand nombre d'itérations	Acteur: Classe Worker	Actions: Appeler call() avec numIterations = 100000	Résultats attendus: Résultat =< numIterations, aucune exception

Identification du test : C4 : Liste impaire (Master.calculateMedian)			Version : 1.0	

Description du test : Vérifier le calcul de la médiane pour une liste de taille impaire.				
Ressources requises : JDK, classe Worker				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Liste impaire	Acteur: Méthode calculateMedian	Actions: Appeler calculateMedian([1,3,5])	Résultats attendus: Médiane = 3

Identification du test : C5 : Liste paire (Master.calculateMedian)			Version : 1.0	
Description du test : Vérifier le calcul de la médiane pour une liste de taille paire.				
Ressources requises : JDK, classe Worker				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Liste impaire	Acteur: Méthode calculateMedian	Actions: Appeler calculateMedian([2,4])	Résultats attendus: Médiane = 3

Identification du test : C6 : Exécution simple (Master.doRun)			Version : 1.0	
Description du test : Vérifier l'exécution avec un seul Worker.				
Ressources requises : JDK, classe Worker				
Responsable : BERARD Clément				
Etat initial: Script prêt à être	Cas testé: 1 Worker	Acteur: Méthode doRun	Actions: Appeler doRun(1000,1)	Résultats attendus: Valeur de π

exécuté, aucun calcul en cours				comprise entre 0 et 4

Identification du test : C7 : Exécution parallèle (Master.doRun)			Version : 1.0	
Description du test : Vérifier l'exécution avec plusieurs Workers.				
Ressources requises : JDK, classe Worker				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Plusieur Workers	Acteur: Méthode doRun	Actions: Appeler doRun(100000,4)	Résultats attendus: Valeur de π cohérente, aucune erreur

Monte Carlo, Master Worker MPI

Identification du test : C1 : Boucle non exécutée			Version : 1.0	
Description du test : Vérifier le comportement de la fonction Monte-Carlo lorsque le nombre d'itérations est nul.				
Ressources requises : Python 3, fonction monter_carlo				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: N = 0	Acteur: Fonction monte_carlo	Actions: Appeler monte_carlo(0)	Résultats attendus: Boucle non exécutée, résultat = 0

Identification du test : C2 : Condition toujours vraie (Monte-Carlo)			Version : 1.0	
Description du test : Vérifier le comportement de l'algorithme lorsque la condition $x^2 + y^2 \leq 1$ est satisfaite lors des itérations, entraînant une incrémentation du compteur.				
Ressources requises : Python 3, fonction monter_carlo				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Nombre d'itérations strictement positif	Acteur: Fonction monte_carlo	Actions: Appeler la fonction avec total_count = 1	Résultats attendus: Le résultat retourné est compris entre 0 et 1

Identification du test : C3 : Condition toujours fausse			Version : 1.0	
Description du test : Vérifier le comportement de l'algorithme lorsque la condition $x^2 + y^2 \leq 1$ n'est jamais satisfaite durant les itérations.				
Ressources requises : Python 3, fonction monter_carlo				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Points générés hors du quart de cercle	Acteur: Fonction monte_carlo	Actions: Appeler la fonction avec total_count > 0	Résultats attendus: Le compteur inside reste faible ou nul, résultat ≥ 0

Identification du test : C4 : Cas nominal (condition vraie et fausse)			Version : 1.0	
Description du test : Vérifier le fonctionnement normal de l'algorithme lorsque la condition est vraie pour certaines itérations et fausse pour d'autres.				
Ressources requises : Python 3, fonction monter_carlo				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Nombre d'itérations élevé	Acteur: Fonction monte_carlo	Actions: Appeler la fonction avec total_count = 100000	Résultats attendus: Résultat strictement compris entre 0 et total_count, aucune exception

Monte Carlo, Master.py

Identification du test : C1 : Boucle non exécutée			Version : 1.0	
Description du test : Vérifier le comportement de la fonction Monte-Carlo lorsque le nombre d'itérations est nul.				
Ressources requises : Python 3, fonction monter_carlo				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: N = 0	Acteur: Fonction monte_carlo	Actions: Appeler monte_carlo(0)	Résultats attendus: Boucle non exécutée, résultat = 0

Identification du test : C2 : Condition toujours vraie (Monte-Carlo)			Version : 1.0	
Description du test : Vérifier le comportement de l'algorithme lorsque la condition $x^2 + y^2 \leq 1$ est satisfaite lors des itérations, entraînant une incrémentation du compteur.				
Ressources requises : Python 3, fonction monter_carlo				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Nombre d'itérations strictement positif	Acteur: Fonction monte_carlo	Actions: Appeler la fonction avec total_count = 1	Résultats attendus: Le résultat retourné est compris entre 0 et 1

Identification du test : C3 : Condition toujours fausse			Version : 1.0	
Description du test : Vérifier le comportement de l'algorithme lorsque la condition $x^2 + y^2 \leq 1$ n'est jamais satisfaite durant les itérations.				
Ressources requises : Python 3, fonction monter_carlo				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Points générés hors du quart de cercle	Acteur: Fonction monte_carlo	Actions: Appeler la fonction avec total_count > 0	Résultats attendus: Le compteur inside reste faible ou nul, résultat ≥ 0

Identification du test : C4 : Cas nominal (condition vraie et fausse)			Version : 1.0	
Description du test : Vérifier le fonctionnement normal de l'algorithme lorsque la condition est vraie pour certaines itérations et fausse pour d'autres.				
Ressources requises : Python 3, fonction monter_carlo				
Responsable : BERARD Clément				
Etat initial: Script prêt à être exécuté, aucun calcul en cours	Cas testé: Nombre d'itérations élevé	Acteur: Fonction monte_carlo	Actions: Appeler la fonction avec total_count = 100000	Résultats attendus: Résultat strictement compris entre 0 et total_count, aucune exception

Loi Normale (Simpson), Worker Sockets.java

Identification du test : C1 : Valeur de n invalide			Version : 1.0	
Description du test : Vérifier le comportement de l'algorithme de Simpson lorsque le nombre de subdivisions est nul ou négatif.				
Ressources requises : PC, JDK / Python, fonction Simpson				
Responsable : BERARD Clément				
Etat initial: Fonction prête	Cas testé: n = 0	Acteur: Fonction Simpson	Actions: Appeler la fonction avec a=0, b=1, n=0	Résultats attendus: Erreur ou arrêt du calcul

Identification du test : C2 : n impair			Version : 1.0	
Description du test : Vérifier que l'algorithme refuse un nombre de subdivisions impair.				
Ressources requises : PC, JDK / Python, fonction Simpson				
Responsable : BERARD Clément				
Etat initial: Fonction prête	Cas testé: n = 3	Acteur: Fonction Simpson	Actions: Appeler la fonction avec a=0, b=1, n=3	Résultats attendus: Erreur ou résultat non calculé

Identification du test : C3 : Cas minimal valide			Version : 1.0	
Description du test : Vérifier le fonctionnement de Simpson avec le minimum de subdivisions valides.				
Ressources requises : PC, JDK / Python, fonction Simpson				
Responsable : BERARD Clément				
Etat initial: Fonction prête	Cas testé: n = 2	Acteur: Fonction Simpson	Actions: Appeler la fonction avec a=0, b=1, n=2	Résultats attendus: Résultat numérique valide

Identification du test : C4 : Plusieurs itérations			Version : 1.0	
Description du test : Vérifier le calcul Simpson avec plusieurs subdivisions.				
Ressources requises : PC, JDK / Python, fonction Simpson				
Responsable : BERARD Clément				
Etat initial: Fonction prête	Cas testé: n = 100	Acteur: Fonction Simpson	Actions: Appeler la fonction avec a=0, b=1, n=100	Résultats attendus: Valeur proche de l'intégrale réelle

Identification du test : C5 : Cas nominal			Version : 1.0	
Description du test : Vérifier la précision de la méthode de Simpson sur une fonction connue.				
Ressources requises : PC, JDK / Python, fonction Simpson				
Responsable : BERARD Clément				
Etat initial: Fonction prête	Cas testé: Intégrale de $f(x)=x^2$	Acteur: Fonction Simpson	Actions: Appeler la fonction sur [0,1]	Résultats attendus: Résultat ≈ 1/3

3. Résultats des tests

Référence du test appliqué	Prime Improved – C1 : boucle interne non exécutée 1.0
Responsable	BERARD Clément
Date de l'application du test	19/10/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Prime Improved – C2 : nombre premier sans diviseur 1.0
Responsable	BERARD Clément
Date de l'application du test	19/10/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Prime Improved – C3 : nombre composé avec diviseur 1.0
Responsable	BERARD Clément
Date de l'application du test	19/10/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Prime Improved – C4 : plusieurs itérations 1.0
Responsable	BERARD Clément
Date de l'application du test	19/10/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Prime Improved – C5 : aucun candidat à tester 1.0
Responsable	BERARD Clément
Date de l'application du test	19/10/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Threads – C1 : boucle non exécutée 1.0
Responsable	BERARD Clément
Date de l'application du test	12/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Threads – C2 : condition toujours vraie 1.0
Responsable	BERARD Clément
Date de l'application du test	12/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Threads – C3 : chemins mixtes 1.0
Responsable	BERARD Clément
Date de l'application du test	12/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Threads – C4 : médiane liste impaire 1.0
Responsable	BERARD Clément
Date de l'application du test	12/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Threads – C5 : médiane liste paire 1.0
Responsable	BERARD Clément
Date de l'application du test	12/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Threads – C6 : exécution simple 1.0
Responsable	BERARD Clément
Date de l'application du test	12/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Threads – C7 : exécution parallèle 1.0
Responsable	BERARD Clément
Date de l'application du test	12/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo MPI – C1 : boucle non exécutée 1.0
Responsable	BERARD Clément
Date de l'application du test	26/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo MPI – C2 : condition toujours vraie 1.0
Responsable	BERARD Clément
Date de l'application du test	26/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo MPI – C3 : condition toujours fausse 1.0
Responsable	BERARD Clément
Date de l'application du test	26/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo MPI – C4 : cas nominal 1.0
Responsable	BERARD Clément
Date de l'application du test	26/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Master.py – C1 : boucle non exécutée 1.0
Responsable	BERARD Clément
Date de l'application du test	16/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Master.py – C2 : condition toujours vraie 1.0
Responsable	BERARD Clément
Date de l'application du test	16/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Master.py – C3 : condition toujours fausse 1.0
Responsable	BERARD Clément
Date de l'application du test	16/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Monte Carlo Master.py – C4 : cas nominal 1.0
Responsable	BERARD Clément
Date de l'application du test	16/12/2025
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Simpson – C1 : n invalide 1.0
Responsable	BERARD Clément
Date de l'application du test	12/01/2026
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Simpson – C2 : n impair 1.0
Responsable	BERARD Clément
Date de l'application du test	12/01/2026
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Simpson – C3 : cas minimal valide 1.0
Responsable	BERARD Clément
Date de l'application du test	12/01/2026
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Simpson – C4 : plusieurs itérations 1.0
Responsable	BERARD Clément
Date de l'application du test	12/01/2026
Résultat du test	OK
Occurrences des résultats	Systématique

Référence du test appliqué	Simpson – C5 : cas nominal 1.0
Responsable	BERARD Clément
Date de l'application du test	12/01/2026
Résultat du test	OK
Occurrences des résultats	Systématique