Chauvière Romy 3B

Évaluation des algorithmes

Algorithmes évalués :

- Efficacité :
  - 03efficacite.c
  - 06efficacite.java
  - 12efficacite.c
  - 23efficacite.java
  - 68efficacite.java

- Simplicité :
  - 11simplicite.java
  - 12simplicite.java
  - 18simplicite.java
  - 30simplicite.py
  - 31simplicite.java

- Sobriété :
  - 11sobriete.c
  - 27sobriete.py
  - 41sobriete.java
  - 61sobriete.java

Sommaire

Présentation des outils d’évaluation

Pour évaluer ces fonctions, j’ai utilisé comme IDE VSCode, IntelliJ et PyCharm.

J’ai effectué les tests avec JUnit et PyUnit et évalué la qualité du code avec Codacy.

Pour calculer le temps d’exécution j’ai utilisé les fonctionnalités de System comme currentTimeMillis().

Évaluation des algorithmes

Données utilisées pour les tests de temps :

- «WWWWWWWWWBWWWWWWWWBBBWWWBWWWWWWW»
- 50 itérations (ou 30 mais dans ce cas là ce sera précisé)

Efficacité

Efficacite03 :

Critère	Résultat
Lisibilité	Lisible, quelques commentaires, Javadoc sur la classe, non de variables pas toujours clairs
Qualité du code	Qualité B d’après Codacy mais possibilité d’avoir une boucle infinie
Efficacité	$O(n^2)$ complexité quadratique dans le pire des cas car boucles for et while imbriquée sinon O(n)
Temps d’exécution	court
Tests	Compile (avec ajout d’un main) mais ne passe pas les tests

Note attribuée : 10/20

Efficacite06 :

Critère	Résultat
Lisibilité	Lisible, quelques commentaires, pas de Javadoc
Qualité du code	Qualité A d'après Codacy
Efficacité	$O(n^2)$ complexité quadratique dans le pire des cas car boucles for et while imbriquée sinon O(n)
Temps d'exécution	(352ms,335ms,341ms) soit en moyenne 342ms
Tests	4/4

Note attribuée : 18/20

12efficacite.c :

Critère	Résultat
Lisibilité	Lisible, pas de commentaires ou documentation, nom de variables clairs
Qualité du code	Qualité D d'après Codacy mais mauvaise sécurité (utilisation de printf)
Efficacité	$O(n^2)$ complexité quadratique dans le pire des cas car boucles for et while imbriquée sinon O(n)
Temps d'exécution	court
Tests	4/4

Note attribuée : 18/20

Efficacite23 :

Critère	Résultat
Lisibilité	Lisible, quelques commentaires, pas de Javadoc, erreur de syntaxe sur une variable (nommée Compressed)
Qualité du code	Qualité A d'après Codacy
Efficacité	$O(n^2)$ complexité quadratique dans le pire des cas car boucles for et while imbriquée sinon O(n)
Temps d'exécution	Pour 30 itérations : 842ms au-delà → trop long
Tests	0/4 fait l'inverse de ce qui est demandé (au lieu de retourner 1a1b1c pour la chaîne abc, retourne a1b1c1)

Note attribuée : 4/20 et hors concours

Efficace68 :

Critère	Résultat
Lisibilité	Lisible, quelques commentaires, Javadoc sur la classe, non de variables pas toujours clairs
Qualité du code	Qualité A d'après Codacy
Efficacité	$O(n^2)$ complexité quadratique dans le pire des cas car boucles for et while imbriquée sinon O(n)
Temps d'exécution	(276ms,253ms,293ms) soit une moyenne de 274ms
Tests	4/4

Note attribuée : 18/20

Sobriété

27sobriete.py :

Critère	Résultat
Lisibilité	Lisible, quelques commentaires, pas de documentation, nom de variables clairs
Qualité du code	Qualité C d'après Codacy mais la variable résutlat ne peut pas être utilisée
Efficacité	$O(n^2)$ complexité quadratique dans le pire des cas car boucles for et while imbriquée sinon O(n)
Temps d'exécution	court
Tests	2/4

Note attribuée 10/20

41sobriete.java :

Critère	Résultat
Lisibilité	Lisible, quelques commentaires, pas de Javadoc, nom de variables explicites
Qualité du code	Qualité C d'après Codacy mais mauvaise vérification pour la condition null et code peu propre
Efficacité	O(n)
Temps d'exécution	347ms (pour 30 itérations)
Tests	4/4

Note attribuée : 18/20

61sobriete.java :

Critère	Résultat
Lisibilité	Lisible, quelques commentaires, Javadoc sur la classe, non de variables pas toujours clairs
Qualité du code	Qualité B d’après Codacy mais manquements aux normes de codage par exemple
Efficacité	O(n) mais mauvaise gestion des boucles donc 2 boucles infinies sont possibles
Temps d’exécution	384ms (pour 30 itérations)
Tests	4/4

Note attribuée : 18/20

Simplicité

Simplicite11 :

Critère	Résultat
Lisibilité	Lisible, aucun commentaires, aucune Javadoc, nom de variables clairs
Qualité du code	Qualité A d’après Codacy
Efficacité	O(n)
Tests	4/4

Note attribuée : 18/20

Simplicite12 :

Critère	Résultat
Lisibilité	Lisible, aucun commentaires, aucune Javadoc, nom de variables clairs
Qualité du code	Qualité A d’après Codacy
Efficacité	O(n)
Tests	4/4

Note attribuée : 18/20

Simplicite18 :

Critère	Résultat
Lisibilité	Lisible, aucun commentaires, aucune Javadoc, nom de variables clairs
Qualité du code	Qualité A d'après Codacy
Efficacité	O(n)
Tests	3/4

Note attribuée : 10/20

Simplicite30 :

Critère	Résultat
Lisibilité	Lisible, code commenté, documentation, nom de variables clairs
Qualité du code	Qualité A d'après Codacy
Efficacité	O(n)
Tests	4/4 mais ne prévoit pas les cas où le message est vide !

Note attribuée : 10/20

Simplicite31 :

Critère	Résultat
Lisibilité	Lisible, code commenté, pas de documentation, nom de variables clairs
Qualité du code	Qualité B d'après Codacy
Efficacité	O(n)
Tests	4/4

Note attribuée : 18/20

Classement des algorithmes par critères de comparaison

Efficacité	Simplicité	Sobriété
Efficacite68.java Efficacite06.java Efficacite23.java	Simplicité11.java Simplicite12.java Simplicite31.java Simplicite18.java Simplicite30.java	Sobriete61.java Sobriete41.java Sobriete27.py