Loris Vignolles
Manon Malique
Groupe 4A

2022-2023

IUT BLAGNAC

Rapport SAE 2.02

Table des matières

Outils utilisés :

-Pour mesurer la qualité du code, j'ai utilisé l'outil Codacy qui permet de voir les différents problèmes d'un programme et d'y attribuer une note en fonction de sa qualité générale

-Pour le temps d'exécution, j'ai utilisé un programme disponible dans le dossier analyse. Le temps d'exécution est également le repère pour juger de l'efficacité d'un programme

-Pour la sobriété numérique d'un programme, j'ai utilisé Joular qui détermine différentes performances comme la consommation de la mémoire, la performance globale du programme ainsi qu'une bonne utilisation des structures de données adaptés

Explication notation :

Chacun des algorithmes à été évalué avec plusieurs critères et test. Les résultats sont présentés de manière succincte par catégorie en dessous. Les différentes critères sont détailles, et dans les remarques sont notés les différentes sanctions (Mauvaise nomenclature, nom de classe, erreur compilation), ainsi que les échecs aux différents passages de tests pour les programmes qui fonctionent

Algorithmes - Loris Vignolles

1)Efficacité Meilleur

Algorithme 10:

- Lisibilité du code: 8/10
- Qualité du code: C
- Efficacité: 8/10
- Sobriété numérique:
    - Complexité algorithmique : 6/10
    - Utilisation efficace des structures de données : 6/10
    - Gestion des exceptions : 7/10
    - Performance : 6/10
    - Consommation mémoire : 7/10
- Temps d'exécution: 2.049106 ms
- Note finale: 16/20
- Remarque:

Algorithme 41:

- Effet de compilation: Ne compile pas
- Note finale: N/A
- Remarque:

2)Efficacité Pire

Algorithme 20:

- Lisibilité du code: 6/10
- Qualité du code: B
- Efficacité: 3/10
- Sobriété numérique:
    - Complexité algorithmique : 7/10
    - Utilisation efficace des structures de données : 7/10
    - Gestion des exceptions : 8/10
    - Performance : 6/10
    - Consommation mémoire : 6/10
- Temps d'exécution: 44.176705 ms
- Note finale: 16/20
- Remarque: suspicion de chatGPT + Ne passe pas les tests

Algorithme 36:

- Lisibilité du code: 8/10
- Qualité du code: A
- Efficacité: 5/10
- Sobriété numérique:
    - Complexité algorithmique : 7/10

        - Utilisation efficace des structures de données : 7/10
        - Gestion des exceptions : N/A
        - Performance : 7/10
        - Consommation mémoire : 7/10
    - Temps d'exécution: 2.394731 ms
    - Note finale: 12/20
    - Remarque:

3)Simplicité Meilleur

Algorithme 49:

- Lisibilité du code: 6/10
- Qualité du code: B
- Efficacité: 9/10
- Sobriété numérique:
    - Complexité algorithmique : 7/10
    - Utilisation efficace des structures de données : 7/10
    - Gestion des exceptions : 8/10
    - Performance : 6/10
    - Consommation mémoire : 6/10
- Temps d'exécution: 1.29124 ms
- Note finale: 8/20
- Remarque: Pas une classe Exercice + chatGPT

Algorithme 64:

- Lisibilité du code: 8/10
- Qualité du code: D
- Efficacité: 4/10
- Sobriété numérique:
    - Complexité algorithmique : 7/10
    - Utilisation efficace des structures de données : 8/10

        - Gestion des exceptions : 9/10
        - Performance : 7/10
        - Consommation mémoire : 7/10
    - Temps d'exécution: 21.789928 ms
    - Note finale: 7/20
    - Remarque: Pas une classe Exercice + chatGPT

Algorithme 27:

- Lisibilité du code: 8/10
- Qualité du code: B
- Efficacité: 8/10
- Sobriété numérique:
    - Complexité algorithmique : 7/10
    - Utilisation efficace des structures de données : 7/10
    - Gestion des exceptions : N/A
    - Performance : 6/10
    - Consommation mémoire : 7/10
- Temps d'exécution: 1.902001 ms
- Note finale: 15/20
- Remarque: Mauvaise nomenclature

4)Simplicité Pire

Algorithme 28:

- Lisibilité du code: 7/10
- Qualité du code: N/A
- Efficacité: 4/10
- Sobriété numérique:
    - Complexité algorithmique : 6/10
    - Utilisation efficace des structures de données : 7/10
    - Gestion des exceptions : N/A
    - Performance : 6/10
    - Consommation mémoire : 7/10
- Temps d'exécution: 25.711398 ms

- Note finale: 15/20
- Remarque:

Algorithme 63-2:

- Lisibilité du code: 0/10
- Qualité du code: N/A
- Efficacité: 4/10
- Sobriété numérique:
    - Complexité algorithmique : 7/10
    - Utilisation efficace des structures de données : 7/10
    - Gestion des exceptions : 8/10
    - Performance : 6/10
    - Consommation mémoire : 6/10
- Temps d'exécution: 25.814743 ms
- Note finale: 18/20
- Remarque:

5)Sobriété Meilleur

Algorithme 66:

- Lisibilité du code: 7/10
- Qualité du code: E
- Efficacité: 5/10
- Sobriété numérique:
    - Complexité algorithmique : 7/10
    - Lisibilité du code : 7/10
    - Utilisation efficace des structures de données : 8/10
    - Gestion des exceptions : 10/10
    - Performance : 6/10
    - Consommation mémoire : 7/10
- Temps d'exécution: 22.702603 ms
- Note finale: 6/20
- Remarque: Pas une classe Exercice + chatGPT

Algorithme 40:

- Lisibilité du code: 8/10
- Qualité du code: B
- Efficacité: 9/10
- Sobriété numérique:
    - Complexité algorithmique : 7/10
    - Utilisation efficace des structures de données : 7/10
    - Gestion des exceptions : N/A
    - Performance : 7/10
    - Consommation mémoire : 7/10
- Temps d'exécution: 1.582789 ms
- Note finale: 14/20
- Remarque:

6)Sobriété Pire

Algorithme 22:

- Lisibilité du code: 6/10
- Qualité du code: N/A
- Efficacité: 7/10
- Sobriété numérique:
    - Complexité algorithmique : 5/10
    - Utilisation efficace des structures de données : 7/10
    - Gestion des exceptions : 7/10
    - Performance : 4/10
    - Consommation mémoire : 6/10
- Temps d'exécution: 4.616485 ms
- Note finale: 8/20
- Remarque: Pas une classe Exercice

Algorithme 63-1:

- Lisibilité du code: 4/10
- Qualité du code: B
- Efficacité: 0/10
- Sobriété numérique:
    - Complexité algorithmique : 5/10
    - Utilisation efficace des structures de données : 4/10

    - Gestion des exceptions : 7/10
    - Performance : 3/10
    - Consommation mémoire : 2/10
- Temps d'exécution: Java heap space (Erreur de débordement de mémoire)
- Note finale: 18/20
- Remarque: Tellement efficace qu'il ne fonctionne pas

Classement Algorithmes Loris:

Efficacité Meilleur: 10 – 41

Efficacité Pire : 20 – 36

Simplicité Meilleur : 27 – 49 -64

Simplicité Pire : 63-2 28

Sobriété Meilleur : 40 – 66

Efficacité Pire : 63-1 22

Algorithmes – Manon Malique

1)Efficacité Meilleur

Algorithme 9:

- Ne passe pas les trois derniers tests
- Lisibilité du code: 6/10 (+1 : bonus note qualité de Codacy ci-dessous)
    - Qualité du code: B (A:+2 ; B:+1 ; C+0,5)
- Efficacité: 4/10 (2pts par test réussi)
- Sobriété numérique: 5,2/10 (5,2 =1+1,2+0,8+1+1,2)
    - Complexité algorithmique : 1/2
    - Utilisation efficace des structures de données : 1,2/2
    - Gestion des exceptions : 0,8/2
    - Performance : 1/2
    - Consommation mémoire : 1,2/2
- Temps d'exécution: 1,758468 s

```
[Ceci, est, une, chaine, de, texte, moyennement, longue]
[moyennement, chaine, est, une, Ceci, de, texte, longue]
Temps d'exécution : 1758468 nanosecondes

Test - Chaine vide
Résultat attendu : []
Résultat obtenu  : []

[Hello, 123, World, 456]
Test - Avec chiffres
Résultat attendu : [Hello, World, 123, 456]
Résultat obtenu  : [Hello, World, 123, 456]

[Bonjour, le, monde, ]

Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: 0
    at java.base/java.lang.StringLatin1.charAt(StringLatin1.java:48)
    at java.base/java.lang.String.charAt(String.java:1512)
    at Exercice.solution(Exercice.java:37)
    at MyClass.testAvecEspaces(MyClass.java:40)
    at MyClass.main(MyClass.java:10)
```

- Note finale: 9,68/20
  (7+4+4+4+5,2)/5 =4,84/10
  (moyenne des 3 critères : le critère visé aura un coefficient de 3 contre 1 pour les autres
  exemple ici : Efficacité coefficient 3)
- suspicion chatGPT

Algorithme 41:

- Ne fonctionne pas : compile, retour erroné, ne passe pas les tests
- Note finale: 5/20

```
[Ceci, est, chaine, de, texte, moyennement, une, longue]
Temps d'exécution : 47098539 nanosecondes

Test - Chaine vide
Résultat attendu : []
Résultat obtenu  : []

Test - Avec chiffres
Résultat attendu : [Hello, World, 123, 456]
Résultat obtenu  : [123, 456, Hello, World]

Test - Avec espaces
Résultat attendu : [Bonjour, le, monde]
Résultat obtenu  : [le, monde,  , Bonjour]

Test - Classique
Résultat attendu : [This, test,  Hello, is, world, a, ]
Résultat obtenu  : [world, is, a, test., This, Hello]

Test - Mot non dans le dictionnaire
Résultat attendu : [N1c3, Hello, is, world, This, a, test]
Résultat obtenu  : [world, This, is, a, test, N1c3, Hello]
```

2)Efficacité Pire

Algorithme 40-2:

- Lisibilité du code: 6/10 (+0,5)
    - Qualité du code: C (A:+2 ; B:+1 ; C+0,5)
- Efficacité: 6/10 (+1) (+1 bonus des deux derniers tests presque réussis)
- Sobriété numérique: 4,2/10
    - Complexité algorithmique : 1,2/2
    - Utilisation efficace des structures de données : 0,8/2
    - Gestion des exceptions : 2/2
    - Performance : 1/2
    - Consommation mémoire : 0,8/2
- Temps d'exécution:(TimeOut : tellement que le programme est pas efficace donc bonus dans la note efficacité)
- Note finale: 13,88/20

```
[moyennement, chaine, est, une, Ceci, de, texte, longue]
Temps d'exécution : 327815046 nanosecondes
```

```
Test - Chaine vide
Résultat attendu : []
Résultat obtenu  : []

Test - Avec chiffres
Résultat attendu : [Hello, World, 123, 456]
Résultat obtenu  : [Hello, World, 123, 456]

Test - Avec espaces
Résultat attendu : [Bonjour, le, monde]
Résultat obtenu  : [Bonjour, le, monde]

Test - Classique
Résultat attendu : [This, test,  Hello, is, world, a, ]
Résultat obtenu  : [This, Hello, world, is, a, test]

Test - Mot non dans le dictionnaire
Résultat attendu : [N1c3, Hello, is, world, This, a, test]
Résultat obtenu  : [N1c3, Hello, world, This, is, a, test]
```

Algorithme 51:

- Lisibilité du code: 5/10 (+0,5)
    - Qualité du code: C
- Efficacité: 6/10 (+1) (+1 bonus des deux tests presque réussis)
- Sobriété numérique: 5,8/10
    - Complexité algorithmique : 1,4/2
    - Utilisation efficace des structures de données : 1/2

- Gestion des exceptions : 0,8/2
- Performance : 1,8/2
- Consommation mémoire : 0,8/2

- Temps d'exécution: (TimeOut : tellement que le programme est pas efficace
- donc bonus dans la note efficacité)
- Malus : nomenclature de la classe incorrect (pas Exercice donc -1)
- Note finale: 13,12/20

```
Le programme est en train de se lancer
15% complété
Petite pause réseaux sociaux
Le programme est en train de planifier la conquête du monde, il a besoin de votre aide et de vos données personnelles
Petite pause réseaux sociaux
Petite pause réseaux sociaux


 JDoodle - Timeout
```

```
100% complété
Test - Chaine vide
Résultat attendu : []
Résultat obtenu  : []

100% complété
Test - Avec chiffres
Résultat attendu : [Hello, World, 123, 456]
Résultat obtenu  : [Hello, World, 123, 456]

100% complété
Test - Avec espaces
Résultat attendu : [Bonjour, le, monde]
Résultat obtenu  : [Bonjour, le, monde]

100% complété
Test - Classique
Résultat attendu : [This, test,  Hello, is, world, a, ]
Résultat obtenu  : [This, Hello, world, is, a, test]

100% complété
Test - Mot non dans le dictionnaire
Résultat attendu : [N1c3, Hello, is, world, This, a, test]
Résultat obtenu  : [N1c3, Hello, world, This, is, a, test]
```

3)Simplicité Meilleur

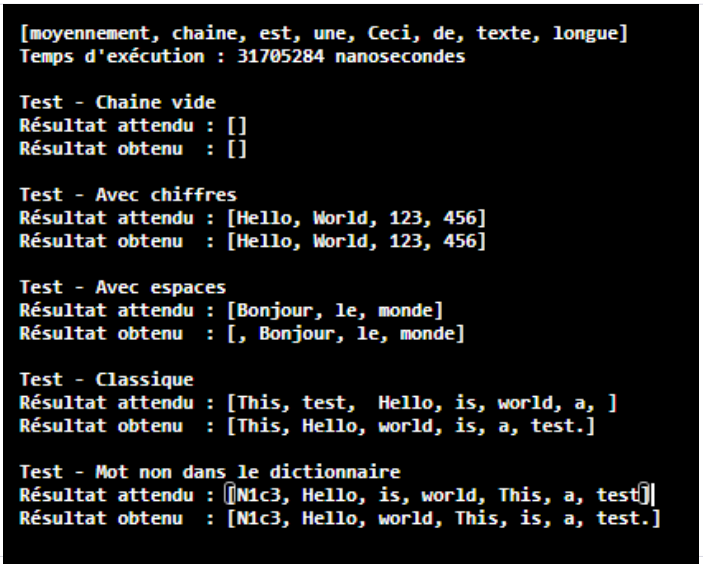
Algorithme 24:

- Ne fonctionne pas : compile, retour erroné, ne passe pas les tests
- Malus mauvaise nomenclature (-1)
- Note finale: 4/20

```
[moyennement, chaine, est, une, Ceci, de, texte, longue]
Temps d'exécution : 2210356 nanosecondes


Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: 0
    at java.base/java.lang.StringLatin1.charAt(StringLatin1.java:48)
    at java.base/java.lang.String.charAt(String.java:1512)
    at Exercice.solution(Exercice.java:16)
    at MyClass.testChaineVide(MyClass.java:18)
    at MyClass.main(MyClass.java:8)
```

Algorithme 64:

- Lisibilité du code: 8/10 (+1)
    - Qualité du code: B
- Efficacité: 4/10 (+1 bonus des trois tests presque réussis)
- Sobriété numérique: 6,8/10
    - Complexité algorithmique : 1,6/2
    - Utilisation efficace des structures de données : 1,8/2
    - Gestion des exceptions : 0/2
    - Performance : 1,8/2
    - Consommation mémoire : 1,6/2
- Temps d'exécution: 31 s
- Note finale: 15,52/20



Algorithme 30:

- Ne fonctionne pas
- Malus : mauvaise nomenclature x2 (-1) ; non respect des consignes
- Note finale : 1,5/20

4)Simplicité Pire

Algorithme 12:

- Lisibilité du code: 3/10 (-2)
    - Qualité du code: A (A:-2 ; B:-1 ; C-0,5) (le bonus dans meilleur devient un malus dans simplicite pire)
- Efficacité: 4/10
- Sobriété numérique: 4,4/10
    - Complexité algorithmique : 0,8/2
    - Utilisation efficace des structures de données : 0,4/10

- Gestion des exceptions : 2/2
- Performance : 0,8/10
- Consommation mémoire : 0,4/10

- Temps d'exécution: 26s
- Note finale: 4,56/20

```
[moyennement, chaine, est, une, Ceci, de, texte, longue]
Temps d'exécution : 26392746 nanosecondes

Test - Chaine vide
Résultat attendu : []
Résultat obtenu  : []

Test - Avec chiffres
Résultat attendu : [Hello, World, 123, 456]
Résultat obtenu  : [Hello, World, 123, 456]


Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of rar
    at java.base/java.lang.StringLatin1.charAt(StringLatin1.java:48)
    at java.base/java.lang.String.charAt(String.java:1512)
    at Exercice.lambda$solution$0(Exercice.java:24)
    at java.base/java.util.TimSort.countRunAndMakeAscending(TimSort.java:355)
    at java.base/java.util.TimSort.sort(TimSort.java:220)
    at java.base/java.util.Arrays.sort(Arrays.java:1307)
    at java.base/java.util.ArrayList.sort(ArrayList.java:1721)
    at Exercice.solution(Exercice.java:46)
    at MyClass.testAvecEspaces(MyClass.java:40)
    at MyClass.main(MyClass.java:10)
```

Algorithme 58:

- Lisibilité du code: 10/10
- Qualité du code: D
- Efficacité: 6/10 (+1 pour les deux tests presque réussis)
- Sobriété numérique: 4,4/10
    - Complexité algorithmique : 1/2
    - Utilisation efficace des structures de données : 0,4/2
    - Gestion des exceptions : 1,6/2
    - Performance : 1/2
    - Consommation mémoire : 0,4/2

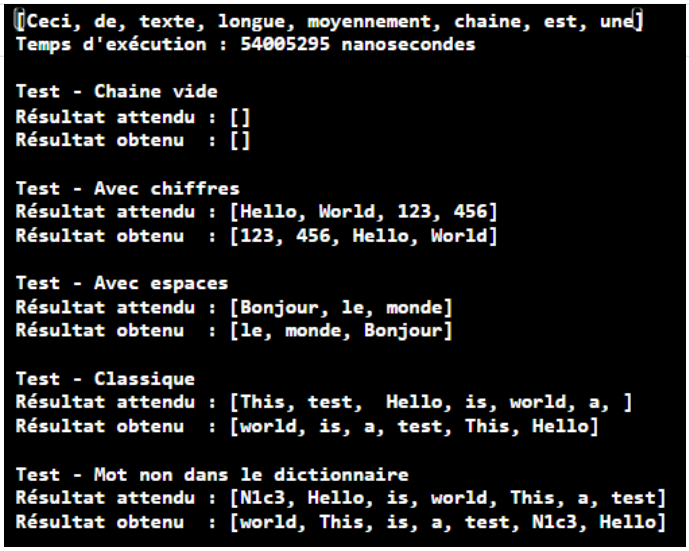
- Temps d'exécution: 22s
- Note finale: 16,56/20

```
[moyennement, chaine, est, une, Ceci, de, texte, longue]
Temps d'exécution : 22208213 nanosecondes

Test - Chaine vide
Résultat attendu : []
Résultat obtenu  : []

Test - Avec chiffres
Résultat attendu : [Hello, World, 123, 456]
Résultat obtenu  : [Hello, World, 123, 456]

Test - Avec espaces
Résultat attendu : [Bonjour, le, monde]
Résultat obtenu  : [Bonjour, le, monde]

Test - Classique
Résultat attendu : [This, test,  Hello, is, world, a, ]
Résultat obtenu  : [This, Hello, world, is, a, test]

Test - Mot non dans le dictionnaire
Résultat attendu : [N1c3, Hello, is, world, This, a, test]
Résultat obtenu  : [N1c3, Hello, world, This, is, a, test]
```

5)Sobriété Meilleur

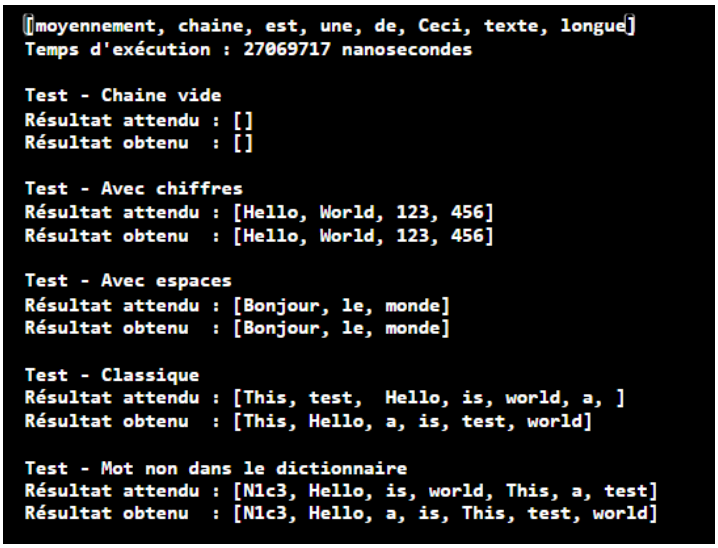
Algorithme 19:

- Lisibilité du code: 7/10 (+0,5)
    - Qualité du code: C
- Efficacité: 2/10
- Sobriété numérique: 6/10
    - Complexité algorithmique : 1,4/2
    - Utilisation efficace des structures de données : 1/2
    - Gestion des exceptions : 1,2/2
    - Performance : 1,2/2
    - Consommation mémoire : 1,2/2
- Temps d'exécution: 54s
- Malus: Pas une classe Exercice (-1)
- Note finale: 10/20
- chatGPT



Algorithme 64:

- Lisibilité du code: 7/10 (+1)
    - Qualité du code: B
- Efficacité: 6/10 (+1 pour les deux tests presque réussis)
- Sobriété numérique: 6,4/10
    - Complexité algorithmique : 1,4/2
    - Utilisation efficace des structures de données : 1/2
    - Gestion des exceptions : 1,2/2
    - Performance : 1,4/2
    - Consommation mémoire : 1,4/2
- Temps d'exécution: 27s
- Malus : pas renommée Exercice (-1)
- Note finale: 12,68/20
- chatGPT



6)Sobriété Pire

Algorithme 15:

- Lisibilité du code: 7/10 (+1)
    - Qualité du code: B
- Efficacité: 6/10 (+1 pour les deux tests presque réussis)
- Sobriété numérique: 4,4/10
    - Complexité algorithmique : 1/2
    - Utilisation efficace des structures de données : 1/2
    - Gestion des exceptions : 0,8/2
    - Performance : 1/2
    - Consommation mémoire : 0,6/2
- Temps d'exécution: 16s

- Note finale: 11,28/20

```
[moyennement, chaine, est, une, Ceci, de, texte, longue]
Temps d'exécution : 16943942 nanosecondes

Test - Chaine vide
Résultat attendu : []
Résultat obtenu  : []

Test - Avec chiffres
Résultat attendu : [Hello, World, 123, 456]
Résultat obtenu  : [Hello, World, 123, 456]

Test - Avec espaces
Résultat attendu : [Bonjour, le, monde]
Résultat obtenu  : [Bonjour, le, monde]

Test - Classique
Résultat attendu : [This, test,  Hello, is, world, a, ]
Résultat obtenu  : [This, Hello, world, is, a, test]

Test - Mot non dans le dictionnaire
Résultat attendu : [N1c3, Hello, is, world, This, a, test]
Résultat obtenu  : [N1c3, Hello, world, This, is, a, test]
```

Algorithme 5:

- Ne fonctionne pas : ne compile pas, retour erroné, ne passe pas les tests
- Note finale: 5/20
- chatGPT

```
[moyennement, chaine, est, une, Ceci, de, texte, longue]
Temps d'exécution : 25973408 nanosecondes


Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: 0
    at java.base/java.lang.StringLatin1.charAt(StringLatin1.java:48)
    at java.base/java.lang.String.charAt(String.java:1512)
    at Exercice.solution(Exercice.java:12)
    at MyClass.testChaineVide(MyClass.java:18)
    at MyClass.main(MyClass.java:8)
```

Classement Algorithme Manon:

Efficacité Meilleur:

1- 9

2- 41

Efficacité Pire :

1- 40(2)

2- 51

Simplicité Meilleur :

1- 64

2- 24

3- 30

Simplicité Pire :

1- 58

2- 12

Sobriété Meilleur :

1- 64

2- 19

Sobriété Pire :

1- 15

2- 5