Groupe 3B

BABELA Guychel

MONESTIER Lilian

SAE 2.02 Exploration algorithmique d'un problème

Introduction

Dans cette situation d'apprentissage et d'évaluation (SAE), nous allons étudier l'approche algorithmique dans le domaine du développement informatique. L'objectif est de résoudre un problème simple mais qui offre plusieurs façons de le résoudre. Le problème consiste à classer les mots d'un texte dans un ordre spécifique. Les élèves devront écrire un programme en Java ou en C pour effectuer ce classement, en respectant les règles précisées. Dans notre compte-rendu, nous examinerons les différentes solutions proposées par les élèves et les classeront dans différentes catégories de leçons. Pour cela, nous utiliserons des outils d'évaluation adaptés dont nous expliquerons les détails dans la première partie de la leçon.

Outils d'évaluation

Joular

Joular, également appelé Joularjx, est un outil spécial qui permet d'évaluer la consommation d'énergie d'un programme informatique. Il surveille combien d'énergie est utilisée par chaque partie du programme pendant son exécution. Cela se fait grâce à une sorte de "super espion" intégré dans le langage de programmation Java. Grâce à cet outil, on peut connaître en temps réel la quantité d'énergie utilisée par chaque partie du programme, ainsi que la quantité totale d'énergie consommée par le programme entier. C'est super utile pour les développeurs car cela leur permet d'optimiser leur code afin de réduire la quantité d'énergie utilisée. Ici, nous allons utiliser Joular pour mesurer combien d'énergie consomment nos algorithmes.

JArchitect

JArchitect est un outil informatique très utile pour évaluer la qualité du code des programmes. Il est utilisé par de nombreux développeurs dans l'industrie du logiciel. Avec JArchitect, on peut analyser, mesurer et voir comment est écrit le code source d'un programme. Il permet de comprendre la structure du code, de repérer les parties qui posent problème et de détecter les erreurs. Ainsi, les développeurs peuvent prendre de bonnes décisions pour améliorer la qualité et la facilité de maintenance de leur code. En utilisant JArchitect, ils peuvent économiser du temps en analysant leur code, trouver les problèmes potentiels et prendre des mesures pour maintenir un code de haute qualité. Nous allons utiliser cet outil pour évaluer la qualité du code des algorithmes que nous étudions.

Critères d’évaluation

Critère de comparaison : Clarté du code

La Clarté du code est un critère qui évalue à quel point le code est facile à lire et à comprendre. Cela signifie que le code doit être bien organisé, avec des noms de variables et de fonctions clairs, afin qu'une personne puisse comprendre rapidement ce que fait le code sans confusion. Un code bien lisible facilite la maintenance et les modifications ultérieures, car il est plus facile de suivre la logique du code.

Critère de comparaison : Qualité du code

La qualité du code se réfère à la manière dont le code est écrit et structuré. Un code de qualité est exempt d'erreurs et suit les bonnes pratiques de programmation. Cela signifie utiliser des conventions de codage cohérentes, éviter les redondances et les duplications de code, et écrire des fonctions et des classes réutilisables. L'utilisation d'outils automatisés pour évaluer la qualité du code, comme Codacy, peut aider à identifier les erreurs et les problèmes potentiels.

Critère de comparaison : Efficacité

L'efficacité d'un code concerne sa performance et la manière dont il utilise les ressources disponibles. Un code efficace s'exécute rapidement et utilise une quantité raisonnable de mémoire. Lorsqu'on parle d'efficacité, il est important de s'assurer que le code ne prend pas trop de temps pour s'exécuter et qu'il ne consomme pas trop de ressources de l'ordinateur. Cela peut être mesuré en évaluant la complexité de l'algorithme utilisé dans le code, c'est-à-dire comment le temps d'exécution augmente lorsque la taille des données augmente.

Critère de comparaison : Sobriété numérique

La sobriété numérique est un concept qui concerne la consommation d'énergie d'un code. Cela signifie évaluer combien d'énergie est nécessaire pour exécuter un programme. Un code sobre numériquement est conçu pour minimiser la consommation d'énergie et est donc plus écologique. Cela peut être important car cela peut réduire les coûts énergétiques et contribuer à la durabilité environnementale.

Critère de comparaison : Temps d'exécution

Le temps d'exécution fait référence à la durée nécessaire pour exécuter un code ou une fonction. Il est important de mesurer le temps d'exécution d'un code car cela peut influencer les performances globales d'un programme. En optimisant le temps d'exécution, on peut rendre un programme plus rapide et plus réactif. Par exemple, si un programme met beaucoup de temps à s'exécuter, cela peut entraîner une expérience utilisateur plus lente. Mesurer le temps d'exécution permet de comparer différentes approches et de choisir celle qui offre les meilleures performances.

Classement des algorithmes de Guychel

Simplicité meilleur:

Numéro	Clarté du code/10	Qualité du code/10	Note/20	Rang
44	9	5	14	2
52	7	9,5	16,5	1
43	6	4	10	3

Simplicité pire:

Numéro	Clarté du code/10	Qualité du code/10	Note/20	Rang
49	6,5	7,5	14	1
32	7,5	6	13,5	Hors concours

Efficacité meilleur:

Numéro	Qualité du code/10	Temps d'exécution	Note/20	Rang
5	7,5	200ns Temps d'exécution : 200 nanosecondes	15	2
15	8	100ns Temps d'exécution : 100 nanosecondes	17	1

Efficacité pire:

Numéro	Qualité du code/10	Temps d'exécution	Note/20	Rang
10	8	100ns Temps d'exécution : 100 nanosecondes	17	1
29	6	100ns Temps d'exécution : 100 nanosecondes	15	2

Sobriété meilleur:

Numéro	Qualité du code/10	Consommation d'énergie électrique	Note/20	Rang
43	5,5	7,96 joules Program consumed 7,96 joules	14	2
7	8	4,79 joules Program consumed 4,79 joules	17	1

Sobriété pire:

Numéro	Qualité du code/10	Consommation d'énergie électrique	Note/20	Rang
29	10	7,66 joules Program consumed 7,66 joules	19,5	1
65	-	6,31 Program consumed 6,31 joules	-	Hors concours

Notation des algorithmes de Guychel

Numéro	Types	Justification	Note
44	Simplicité meilleur	Il passe tous les tests fournis initialement et ne passe aucun de mes tests supplémentaires	17
52	Simplicité meilleur	Il passe que 2 tests fournis initialement au lieu de 4 et ne passe aucun de mes tests supplémentaires	15
43	Simplicité meilleur	Il passe que 3 tests fournis initialement au lieu de 4 et ne	16

		passe aucun de mes tests supplémentaires	
49	Simplicité Pire	Il passe que 1 test fournis initialement au lieu de 4 et ne passe aucun de mes tests supplémentaires	12
32	Simplicité Pire	Il ne fonctionne pas, il renvoi un retour erroné	5
5	Efficacité-meilleur	Il fonctionne mais ne passe pas les tests fournis initialement	10
15	Efficacité-meilleur	Il passe tous les tests fournis initialement et ne passe aucun de mes tests supplémentaires	18
10	Efficacité-pire	Il passe que 2 tests fournis initialement au lieu de 4 et ne passe aucun de mes tests supplémentaires	15
29	Efficacité-pire	Il passe que 3 tests fournis initialement au lieu de 4 et ne passe aucun de mes tests supplémentaires	16
43	Sobriété-meilleur	Il passe que 3 tests fournis initialement au lieu de 4 et ne passe aucun de mes tests supplémentaires	16
7	Sobriété-meilleur	Il fonctionne mais ne passe pas les tests fournis initialement	10
29	Sobriété-pire	Il passe que 3 tests fournis initialement au lieu de 4 et ne passe aucun de	16

		mes tests supplémentaires	
65	Sobriété-pire	Il ne fonctionne pas, il renvoi un retour erroné et je constate que l'algo a été fait avec chatgpt	5

Classement des algorithmes de Lilian

Simplicité meilleur:

Numéro	Clarté du code/10	Qualité du code/10	Note/20	Rang
34	5	6	11	2
28	9	7	16	1
55	NA	NA	NA	NA

Simplicité pire:

Numéro	Clarté du code/10	Qualité du code/10	Note/20	Rang
63-3	8	9	17	1
22	4	5	9	2

Efficacité meilleur:

Numéro	Qualité du code/10	Temps d'exécution	Note/20	Rang
19	7	16ms	10	2
56	9	1ms	12	1

Efficacité pire:

Numéro	Qualité du code/10	Temps d'exécution	Note/20	Rang
51	-	>100ms	8.5	2

63-2	**-**	**>100ms**	**9**	**1**

Sobriété meilleur:

Numéro	**Qualité du code/10**	**Note/20**	**Rang**
34	**5**	**10**	**1**
3	**5**	**9**	**2**

Sobriété pire:

Numéro	**Qualité du code/10**	**Note/20**	**Rang**
64	**4**	**8**	**2**
29	**7**	**16**	**1**

Notation des algorithmes de Lilian

Numéro	**Types**	**Justification**	**Note**
34	Simplicité meilleur	Il passe tous les tests fournis initialement mais ne correspond pas à une classe Exercice	11
28	Simplicité meilleur	Il passe sur 3 tests sur les 4 fournis initialement mais le reste du code est bien	15
55	Simplicité meilleur	PAS FAIT	NA

63-3	Simplicité Pire	Il passe sur 3 tests sur les 4 fournis initialement mais l'idée est très originale et le programme fonctionne	17
22	Simplicité Pire	Il passe sur 3 tests sur les 4 fournis initialement et ne correspond pas à une classe Exercice	9
19	Efficacité-meilleur	Il fonctionne mais ne correspond pas à une classe Exercice	10
56	Efficacité-meilleur	Il passe sur 3 tests sur les 4 fournis initialement mais le programme à un temps d'exécution excellent	12
51	Efficacité-pire	Le programme ne respecte pas le nom de la classe et possède des méthodes interdites dans Efficacité. Hors concours	8.5
63-2	Efficacité-pire	Le programme possède des méthodes interdites dans Efficacité. Hors concours	9
34	Sobriété-meilleur	Le programme fonctionne, qualité code correct. Le nom de la classe ne respecte pas les consignes. Temps d'exécution excellent. Suspicion ChatGPT.	10
3	Sobriété-meilleur	Le programme fonctionne, qualité et lisibilité du code correct. Le nom de la classe ne	9

		respecte pas les consignes et le programme passe seulement 3 tests sur les 4 initiales. Temps d'exécution excellent.	
64	Sobriété-pire	Le programme fonctionne avec un temps d'exécution excellent, le programme devrait être dans les pires conditions possibles cependant, il n'y a rien qui change d'un programme "Meilleur" classique. Je pense que le programme consomme beaucoup de ressources (mémoire,calcul..) mais je ne peut pas vérifier. Nom de classe non valide.	8
29	Sobriété-pire	Le programme fonctionne avec un temps d'exécution excellent, le programme devrait être dans les pires conditions possibles alors que là, il n'y a rien qui change d'un programme "Meilleur" classique. Je pense que le programme consomme beaucoup de ressources (mémoire,calcul..) mais je ne peux pas vérifier.	16