1 Introduction

Ce document décrit les critères d'évaluations pour la correction des travaux de programmations. Chaque critère contient des paliers de réussite, une description de ces paliers et la manière de les atteindre.

2 Fonctionnalité

La fonctionnalité est le critère d'évaluation fondamentale. Un programme doit respecter les spécifications du problème donné et fonctionner correctement. Cela inclut de se comporter comme voulu, de produire les bons résultats et sorties pour une variété d'entrées possibles. Cela inclut aussi d'écrire un programme d'une certaine manière ou sous certaines restrictions si cela est demandé dans l'énoncé du travail.

Si la spécification du problème est ambiguë, vous avez deux choix : faire une supposition à propos de ce qui est requis ou demander à l'enseignant. Si vous faites une supposition, vous devriez mentionner dans vos commentaires la supposition utilisée pour que le correcteur soit en connaissance de cause. Une mauvaise supposition apportera une perte de points.

3 Lisibilité

Votre code doit être lisible pour vous et une tierce partie. Cela inclus :

- Bonne utilisation de l'indentation.
- L'ajout d'espacement (espace et saut de ligne) où approprié pour distinguer différents blocs de code. Par exemple, un saut de ligne après une action concrète de plusieurs instructions ou un espace après l'utilisation de virgule dans la liste de paramètres.
- Avoir des noms significatifs de variables.
- Ne pas écrire du code sur une ligne trop longue (environ 80 colonnes maximum)

4 Documentation

5 Efficacité

Il y a toujours plusieurs manières d'écrire un programme qui suit une spécification donnée et une multitude d'implémentations possibles peuvent être considérées comme mauvaises. L'efficacité peut être diminuée par une implémentation qui prend beaucoup plus de lignes de code que ce qui est nécessaire ou qui prend un grand temps d'exécution. Par exemple, de dérouler une boucle (c'est-à-dire

copier-coller plusieurs fois la même partie de code plutôt que d'utiliser une structure itérative) n'est pas efficace, car ça complexifie le code pour rien.

6 Spécification

Les énoncés vont habituellement contenir d'autres spécification ou requis en dehors de la fonctionnalité directe du programme. Par exemple, on peut demander l'ajout de tests avec la remise, des règles de soumissions précises (par moodle ou par courriel) ou d'autres spécifications à l'intérieur du programme (noms de fichiers, validation demandée). Faites bien attention de lire l'énoncé pour bien voir les différentes spécifications demandées.

7 Grille d'évaluation

Chaque critère représente une partie de la note finale; cela est présenté dans la colonne "Pourcentage de la note". Les points seront évalués par rapport aux paliers suivants: "Excellent", "Adéquat", "Faible", "Inadéquat". La fonctionnalité est l'élément primordial et sera la note initiale maximum pouvant être atteinte. Les autres critères seront évalués en corrections négatives.

Par exemple, un programme avec une fonctionnalité "Adéquat", une lisibilité "Faible", une documentation "Adéquat" et "Excellent" dans les autres critères serait évalué comme suit:

$$8/10 - (0.5 - (6/10 * 0.2 + 8/10 * 0.2 + 10/10 * 0.05 + 10/10 * 0.05)) = 68\%$$

Critère	Pourcentage	Excellent(100%)	Adéquat(80%)	Faible(60%)	Inadéquat(0%)
Fonctionnalité	100%	Aucune erreur, le programme fonctionne toujours correctement et réponds aux spécifications et requis.	Détails mineurs du programme sont erronées, le programme fonctionne correctement avec certaines entrées.	Portions significatives de la spécification ne sont pas suivies, plusieurs entrées donnent de mauvais résultats.	Le programme fonctionne seulement pour un cas limité d'entrée ou pas du tout.
Lisibilité	-20%	Aucune erreur, le code est propre, compréhensible et bien organisé.	Détails mineurs dans la consistance d'indentation, utilisation d'espacements, nom de variable ou organisation du code.	Au moins un défaut majeur d'indentation, espacements, noms de variable ou organisation.	Plusieurs défauts majeurs dans les sous-catégories de lisibilité.
Documentation	-20%	Aucune erreur, le code est bien documenté.	Un ou deux endroits qui bénéficieraient de meilleurs commentaires (soit plus ou moins)	Manque au moins un en-tête de fichier, partie de code non commenté ou commentaire impertinent.	En-tête manquant ou commentaire manquant.
Efficacité	-5%	Aucune erreur, utilise la bonne approche partout.	N/A	Au moins, une mauvaise approche dans le code.	Plusieurs mauvaises approches dans le code qui pourraient être optimisées.
Spécifications	-5%	Aucune erreur.	N/A	Détails mineurs de la spécification incorrecte. Par exemple, mauvais nom de fichier ou une dérogation mineure de l'énoncé.	Des erreurs graves des spécifications brisées. Par exemple, ne pas remettre en équipe correctement, ne pas envoyer dans le bon médium.

Il est à noter qu'un programme avec une note de 0 en fonctionnalité ne sera pas évalué sur les autres critères.