1 Introduction

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

2 Fonctionnalité

La fonctionnalité est le critère d'évaluation fondamentalle. Un programme doit respecté les spécifications du problème donné et fonctionné correctement. Cela inclus de ce comporté comme voulu, de produire les bon résultat et sorties pour une variétés d'entrées possibles. Cela inclus aussi d'écrire un programme d'une certaines manières ou sous certaines restrictions si cela est demandé dans l'énoncé du travail.

Si la spécification du problème est ambigue, vous avez deux choix : faire une supposition à propos de ce qui est requis ou de demandé à l'enseignant. Si vous faites une supposition, vous devriez mentionné dans vos commentaire la supposition utilisé 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 un tierce partie. Cela inclus :

- Bonne utilisation de l'indentation.
- L'ajout d'espacement (espace et saut de ligne) où approprié pour distingué différents bloc 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 nom significatifs de variables.
- Ne pas écrire du code sur une ligne trop longues (environ 80 collones maximum)

4 Documentation

5 Efficacité

Il y a toujours plusieurs manières d'écrire une programme qui suit une spécification donnée et une multitudes d'implémentations possibles peuvents être considérés comme mauvaises. L'efficacité peux être diminué par une implémentation qui prends beaucoup plus de lignes de code que ce qui est nécéssaire ou qui prends un grand temps d'exécution. Par exemple, de déroullé une boucle

(de 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 sousmissions précise (par moodle ou par courriel) ou d'autres spécification à l'intérieur du programme (noms de fichiers, validation demandé). Faites bien attention de lire l'énoncé pour bien voir les différentes spécifications demandés.

7 Grille d'évaluation

Chaque critères représente une partie de la note finale; cela est présenté dans la colonne "Pourcentage de la note". Les points seront évaluers par rapport au paliers suivants: "Excellent", "Adéquat", "Faible", "Inadéquat". La fonctionnalité est l'élément primordiale et sera la note initalles maximum pouvant être atteinte. Les autres critères seront évalué 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 erreurs, le programmes fonctionne toujours correctement et réponds aux spécifications et requis.	Détails mineurs du programmes sont erronées, le programme fonctionne correctement avec certaines entrées.	Portions significatives de la spécification ne sont pas suivies, plusieurs entrées donnes de mauvais résultat.	Le programme fonctionne seulement pour un cas limité d'entrée ou pas du tout.
Lisibilité	-20%	Aucune erreurs, 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 défaut majeurs dans les sous-catégorie de lisibilité.
Documentation	-20%	Aucune erreurs, le code est bien documenté.	Un ou deux endroits qui bénifieraient de meilleurs commentaire (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 erreurs, 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 erreurs.	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 envoyé dans le bon médium.

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