Jarod KOHLER S3-C2
Simon NGUYEN S3-C2

Rapport du Mini-projet en C

1. Commande xxd

Point fort :

- Le programme fonctionne correctement
- Une documentation
- Gestion des erreurs
- Écrire n'importe quel message/caractère dans le fichier
- L'affichage est identique à celle demandée
- Le programme xxd est identique à la commande système xxd (vérifié dans l'exercice 2)

Point faible :

- Il manque la gestion de quelques erreurs
- La commande doit prendre seulement 1 fichier à la fois
- La taille du contenu du fichier est limitée dans notre programme (taille maximum de 4096 caractères)

2. Comparaison de programmes

Point fort :

- Le programme fonctionne correctement
- Une documentation
- Gestion des erreurs
- Utilisation de différentes commandes (les commandes basiques ont été testées : ls, cat, echo, xxd, etc...)
- Les commandes fonctionnent peu importe le nombre d'arguments
- Question bonus → Les fichiers créés sont correctement supprimés

Point faible :

- Il manque la gestion de quelques erreurs
- Il doit forcément avoir 2 commandes
- Il manque la création des fichiers temporaires à l'aide de mkstemp (fichiers « temporaires » créés puis supprimés)
- Lorsque l'on veut utiliser une commande avec des options qui commencent par « -- » dans la ligne de commande, ça fonctionne uniquement si c'est la 2ème commande.

2) L’interface de bas-niveau est utilisée dans l’exercice 2 parce que les fonctions étaient plus simples à gérer qu’en haut-niveau et que nous voulions utiliser. (dup2, mkstemp, etc...)

3) Tout d’abord, nous avons récupéré en paramètre la commande 1 et la commande 2 grâce à une boucle qui récupère les chaînes de caractères « argv ». Ensuite, on ajoute les chaînes de caractères de la commande 1 dans un tableau cmd1 et de même pour la commande 2 dans le tableau cmd2.

Puis, nous avons utilisé execvp() et execlp() pour exécuter les différentes commandes. Cependant, tout ce qui vient après execvp() ou execlp() ne s’exécute pas donc il faut utiliser un fork() afin de créer un processus fils et d’exécuter une commande.

Dans notre programme nous, avons utilisé 3 fork() pour exécuter 3 commandes, c’est-à-dire, la commande 1, ensuite la commande 2 et enfin la commande diff -u qui permet de comparer les fichiers A et B et afficher les différences s’il y en a.

4) Le programme du deuxième exercice peut être utilisé pour vérifier le programme du premier exercice en utilisant la commande :

$ ./compare ./xxd texte.txt -- xxd texte.txt

La commande ci-dessus permet de vérifier si le résultat du programme du premier exercice est identique au résultat de la commande xxd du système. En effet, les fichiers sont identiques.