Cahier des charges N°1

Création d'un Shell

Description

Ce document détaille le premier projet AISE. Vous devrez développer un Shell. Il devra se lancer dans un autre Shell et permettre entre autres de lancer des commandes avec des arguments. L'évaluation se fera sur deux parties, une première portera sur les composants basiques étudiés en cours, la seconde sur des aspects avancés. La date de rendu est fixée au :

Vendredi 1er Mars 2018, 23:59:59

Le barème de cette première partie sera très proche de la répartition suivante et vous permettra d'atteindre facilement la moyenne :

Code	Composant	Description	Notation approximative
A1	**Gestion des arguments**	Traiter la chaîne entrée dans la ligne de commande (générer le tableau argv et argc)	1,5
A2	**Support du chemin absolu**	Changer le répertoire courant avec `cd` prenant comme argument un chemin absolu	1,5
A3	**Lancement d'un programme**	Lorsque l'utilisateur appuie sur « entrée », un programme du `$PATH` se lance avec une fonction de la famille **exec.** Le parent ne doit pas terminer avant le fils (comme un lancement de programme traditionnel)	2
A4	**Support de CTRL+D**	Si l'utilisateur fait CTRL+D, le shell doit se fermer	0,5
A5	**Support de CTRL+C**	Interception du signal d'interruption pour interrompre le programme en cours, **mais pas le Shell**	1
A6	**Afficher le prompt**	La ligne de commande doit afficher (hors PS1) l'utilisateur courant, l'hote et le répertoire courant par exemple …) ex : « `mike@ordi1:/home/mike $` »	2
A7	**Système de build**	Le code se construit simplement avec un Makefile (ou mieux (Autotools, Cmake…)	1,5
—	**TOTAL**	—	**10**

Pour aller plus loin, et tenter d'améliorer votre score, voici un certain nombre de fonctionnalités que vous êtes encouragés à explorer. Vous êtes notés sur 25 **pour une note sur 20**, il devrait être facile d'avoir une très bonne note ! De plus, tout bonus vient s'ajouter à ce total de points. Un exemple à 8 points est donné dans le tableau ci-dessous, mais libre à vous d'ajouter les fonctionnalités qui vous intéressent :

Code	Composant	Description	Notation approximative
B8	**Commandes « Built-in »**	Implémenter par exemple : logout, exit, echo, export...	1
B9	**Support du pipe à deux commandes**	Exécuter deux commandes en connectant la sortie de l'une à l'entrée de l'autre	2

jbbesnard@paratools.com adamj@paratools.com

B10	**Chainage de commandes**	Exécuter N commandes en connectant la sortie de l'une à l'entrée de l'autre (dépend du point précédent)	1
B11	**Gestion des chemins relatifs**	Permettre l'utilisation de chemin relatifs	1
B12	**Support de PS1**	Afficher un prompt customizable	0,5
B13	**Exécution en arrière plan**	Si la commande se termine par « & » le shell rend la main immédiatemment	1
B14	**Support PS1 avancé**	et supporter certaines séquence d'échappement communes (http://tldp.org/HOWTO/Bash-Prompt-HOWTO/bash-prompt-escape-sequences.html)	1
B15	**Redirection de sorties**	Redirection de flux selon les arguments (`>>`,`>`, `>&`)	2
B16	**Historique des commandes**	Placez les commandes dans une liste chaînée, parcourez la avec les flèches	2
B17	**Gestion des Jobs**	Support de la commande « jobs » qui permet de lister les processus en arrière plan et leurs PIDs	1
B18	**Support de CTRL+R**	Recherche dans l'historique des commandes quand l'utilisateur fait CTRL+R + expression régulière.	2
B19	**Style de programmation**	Les codes de retour doivent êtres vérifiés (gestion des erreurs). Le code doit être structuré indenté.	0,5
—	**TOTAL**	—	**15**
C20	**Shell Scriptable (BONUS)**	Implémentation votre propre langage de script, le plus complet possible (ex: support de variables, des `test`, `for`, `while`, `read`… (exemple & documentation souhaités).	8
C21	**Autres idées (BONUS)**	Soyez créatifs, points bonus à discretion…	X

Rendu

Vous devrez rendre une archive compressée (lisible sous Linux) nommée selon votre binôme, qui devra contenir au minimum :

- Les **sources** du Shell, en langage C ;
- Un **README** au format texte (« Markdown » si possible) à la racine du projet, expliquant dans les grandes lignes votre projet et la manière de le compiler ;
- La **liste des fonctionnalités** que vous pensez avoir implémentées, en listant l'ensemble des codes des grilles ci-dessus (dans le cas de C21, merci de préciser quelle est votre feature) ;

Chaque archive devra nous parvenir avant la date indiquée en première page, de la manière qui vous convient (email, serveur de téléchargement type WeTransfer, Torrent…). Une évaluation des projets sera faite le dernier jour (après le devoir sur table), directement par une démo de votre Shell. Vous pourrez utiliser la machine que vous souhaitez, tant que celle-ci est sous Linux. Vous pourrez avoir à nous expliquer chaque ligne de votre programme, afin que nous puissions vérifier que vous avez bien compris leur fonctionnement. Chaque archive sera sommée via `sha256sum` à réception pour vérifier qu'il n'y a pas d'altération entre la date de rendu et la démo. Nous utilisons des systèmes de détection du plagiat comme celui fourni par http://theory.stanford.edu/~aiken/moss/, il ne sert donc à rien de renommer les variables en copiant le code d'un autre… N'oubliez pas que vous travaillez pour vous et que la connaissance du Shell et des éléments de base de programmation (Makefile, I/O en C, fork(), signaux, …) est un élément essentiel pour tout informaticien, surtout s'orientant vers le HPC.

Bon Courage !

jbbesnard@paratools.com adamj@paratools.com