Projet Mini-Shell
Rapport

Cazes Noa

Département Sciences du Numérique - Première année
2019-2020

Question 1 et 2 On remarque que, comme le processus père n'attend pas la fin du processus fils pour continuer son exécution après la fin d'une boucle, l'affichage du prompt se fera avant l'affichage du résultat d'exécution de la commande entrée par l'utilisateur.

Question 3 Pour éviter cela, j'ai décidé de mettre en place un traitant du signal SIGCHLD avec la fonction waitpid. Ainsi tant que le fils n'a pas terminé son exécution, le père attend.

Question 4 Pour ne pas que ces commandes internes soient lancées par le processus fils, on crée une condition, de telle sorte que si les commandes sont cd ou exit, le processus père s'en charge, sinon on crée un processus fils avec la primitive fork.

Question 5 Afin de lancer une commande en tâche de fond, on fait de sorte que le processus père n'attende pas la fin de l'exécution du processus fils pour accepter de prendre en charge une nouvelle commande.

Question 6 Pour pouvoir gérer la commande list, on crée une liste chaînée dont chaque élément comporte l'identifiant du processus, son pid, son état (actif ou suspendu), la ligne de commande qu'il traite et un pointeur vers l'élément suivant de cette liste chaînée. Pour la commande stop, on la traite à l'aide du signal SIGINT appliqué à un processus particulier avec la primitive kill. Pour la commande bg, on la traite à l'aide d'un signal SIGCONT, et on fait de sorte que le processu père n'attende pas la fin de son traitement. Pour la commande fg, on fait en sorte que le processus père attende la fin de du processus fils, à l'aide de SIGSTOP. Et on va mettre tout cela en place à l'aide du traitant de SIGCHLD (qui utilise waitpid).

Question 7 Afin de pouvoir gérer la signal SIGINT, on crée un traitant qu'on lui assigne à l'aide de la primitive sigaction, en ayant crée au préalable une structure *struct sigaction sasigint*. Il doit provoquer la terminaison du processus en avant-plan, mais pas de ceux en arrière-plan. De plus, il ne doit pas provoquer la terminaison du shell.

Ainsi, dans le traitant on inscrit alors kill(pid, SIGKILL) et on supprime le processus en question de la liste chaînée. Ensuite on le bloque lorsque le processus fils est concerné mais on l'active lors de la création du processus fils par *fork()*.

Question 8 Pour gérer les redirections, on se sert des attributs in et out de la ligne de commande.

Tout d'abord, on associe aux descripteurs d'entrée et de sortie, les descripteurs standards.

Si *cmd->in != NULL*, cela signifie que le descripteur d'entrée est *cmd->in* et non plus le descripteur 0. Donc on affecte le descripteur d'entrée à *cmd->in*. De même pour le descripteur de sortie.

Question 9 On considère deux tubes p et q, p est le premier tube et q le second tube. L'implantation de cette partie se trouve dans la méthode lancerCmdAvecPipe.

Question 10 Afin de généraliser le code de la question 10, on raisonne comme par récurrence, en traitant la première commande et la dernière commande à part, et en répétant le cas général dans une boucle qui s'exécute tant qu'il reste des commandes (sans considérer la dernière).

Architecture générale et Tests L'ensemble des commandes est rassemblé dans une méthode appelée lancerCmdGenerale, qui est appelée dans la méthode lancerCmdAvecPipe qui permet de réaliser la gestion des pipes, et qui est appelée dans le main.

Ensuite pour la commande list, les différentes informations d'un processus sont dans un enregistrement et une structure enumération a été créée pour l'état d'un processus. On crée une liste chaînée de processus afin de pouvoir avoir accès à toutes les informations nécessaires de chaque processus et de pouvoir ajouter un nombre non limité processus à cet ensemble.

Les signaux ont été traités avec la primitive sigaction et non avec la primitive signal.

Les tests réalisés ont été faits sur des commandes internes, externes, en arrière-plan, en avant-plan, les ramener en avant-plan ou les continuer/stopper en arrière-plan, sur le fait de faire des commandes avec des pipes ou des commandes composées...

Les réponses aux questions 1 à 7 sont dans le fichier question7.c et l'implantation des questions suivantes dans le fichier question10.c.