rapport projet mini-shell

khadija akkar

June 2023

Contents

1 Question 1, 2, 3, 4, 5

L'objectif de ce projet était de développer un mini-shell performant et robuste. J'ai conçu une boucle pour lire les commandes de l'utilisateur à l'aide de la fonction readcmd(), puis j'ai utilisé fork() pour créer un processus fils qui interprète et exécute ces commandes via execvp(). Pour résoudre un problème d'affichage, j'ai ajouté une fonction d'attente dans le processus père avec wait() afin de s'assurer de la fin de l'exécution du processus fils.

Ensuite, j'ai implémenté des commandes internes telles que cd pour changer de répertoire avec chdir(), ainsi que exit pour quitter le shell en utilisant exit(). De plus, j'ai essayé de traiter l'exécution de commandes en arrière-plan en détectant le caractère "" grâce à une structure cmdline*, permettant ainsi au processus père de ne pas attendre la fin de l'exécution pour ces commandes.

2 Question 6, 7

j'ai introduit dans ce qui suit la fonctionnalité de la commande list en utilisant une structure de "listProcess" pour gérer les processus.

La commande "bg" utilise la fonction "kill" pour basculer un processus en arrière-plan, tandis que la commande "fg" utilise la fonction "wait" pour attendre la fin de l'exécution d'un processus.

Il est également possible de suspendre un processus spécifique en utilisant la commande "stop" ou en appuyant sur "Ctrl+Z" au clavier. J'ai pris en charge cette fonctionnalité en ajoutant deux gestionnaires de signaux (handlerSIGTSTP et handlerSIGCHLD) qui traitent les signaux SIGCHLD et SIGSTOP. Ainsi, lorsque le signal SIGSTOP est reçu, le processus est suspendu et lorsque le signal SIGCHLD est reçu, le traitement approprié est effectué.

3 Question 8

La terminaison des processus par l'appui sur "Ctrl+C" au clavier est gérée en ajoutant un gestionnaire de signal (handlerSIGINT) qui traite le signal SIGINT. Ainsi, lorsque le signal SIGINT est reçu, le traitement approprié est effectué pour assurer la terminaison adéquate des processus. (Je n'ai pas réussi à l'implémenter correctement.)

4 Question 9, 10, 11

La redirection est gérée en vérifiant les conditions (cmd-¿in == NULL) et (cmd-¿out == NULL), ce qui permet de modifier les flux d'entrée et de sortie standard.

Les commandes composées sont traitées en reliant les tubes, en dupliquant les descripteurs de fichier et en fermant les extrémités non utilisées des tubes. Cela permet aux différentes commandes de communiquer entre elles via les tubes.

Pour enchaîner plusieurs séquences de filtres liées par des tubes, le processus précédent est généralisé en ajoutant un deuxième tube. Cela facilite la communication entre les séquences de filtres via les tubes.

5 Conclusion

En résumé, j'ai consacré mes meilleurs efforts à appliquer ce que j'ai appris lors des travaux dirigés et pratiques pour aborder le maximum de points dans ce projet. J'ai rencontré plusieurs problèmes et il reste encore des aspects qui ne fonctionnent pas parfaitement. Cependant, j'ai réussi à développer plusieurs points importants dans le projet.