Rapport : MiniShell

HUANG Julien

Département Sciences du Numérique - Première année
2022-2023

Table des matières

Table des figures

1 Introduction

Ce projet vise à vous permettre de mettre en pratique les notions de base vues en TD et TP, autour de la gestion des processus, des signaux et des E/S, dans un contexte suffisamment réaliste. Il s'agit plus précisément de développer un interpréteur de commandes simplifié, offrant les fonctionnalités de base des shells Unix, comme le bash...

2 Question (AVANCEMENT)

— QUESTION 1 : Traité
— QUESTION 2 : Voir Q2.pdf
— QUESTION 3 : Traité
— QUESTION 4 : Traité
— QUESTION 5 : Traité
— QUESTION 6 : Traité
— QUESTION 7 : Traité
— QUESTION 8 : Traité
— QUESTION 9 : Traité
— QUESTION 10 : Traité
— QUESTION 11 : Traité

Il y a parfois quelques bug : exit de fonctionne pas (l'invite de commande ne s'affiche pas, je ne sais pas d'où peut venir le problème).

3 Architecture de l'application

Associe d'abord les signaux à des handler qui vont les gérer (SIGCHLD, SIGINT, SIGTSTP).
Affiche l'invite de commande.

Dès une commande correcte : c'est soit une commande interne dans le processus principal (cd, exit, lj, sj, bg, fg), soit une commande simple dans un processus fils, soit une commande avec un traitement en pipeline dans un processus fils.

Commande simple : 1. fork, 2. redirection, 3. execvp.

Execution avec des pipes : 1. Créer les tubes, 2. fork, 3. redirection, 4. execvp

4 Choix et spécificités de conception

J'ai décidé d'utiliser qu'un seul waitpid, pour eviter la redondance de code, le risque d'incohérence, ou difficulté à la maintenir. Par exemple, un waitpid en dehors du traitant consomme le signal SIGCHLD ce qui peut empêcher d'entrer dans le traitant. Un choix pertinent proposé par Monsieur Hamrouni.

Les commandes internes sont des opérations ou des fonctions (code interne) exécutées par le minishell même.

À la création de chaque fils, ignore les signaux envoyé par ctrl+z et ctrl+c. (SIG_IGN)

Utilisation d'un tableau pour stocker les jobs : Choisi pour sa simplicité de mise en oeuvre.

Ouverture d'un tube et d'un processus fils pour chaque argument.

5 TESTS

Test des commandes simples : les commandes de base telles que "ls", "pwd" ou "echo" sont correctement exécutées et affichent les résultats attendus.

Test d'arret/ suspension : les commande tel que "sleep" avec l'option "&" permet de mettre en background un processus, On peut bien confirmer le bon fonctionnement de CTRL+C et de CTRL+Z avec la commande listjob (lj).

Test des commandes avec arguments : les commandes avec des arguments fonctionnent correctement. Par exemple, testez la commande "ls -l" pour vous assurer que les fichiers sont listés en détail.

Test des redirections : Les redirections d'entrée et de sortie fonctionnent comme prévu. Par exemple, utilisez la commande "cat < fichier.txt" pour vérifier si le contenu du fichier est correctement affiché.

Test des pipelines : Les pipelines, c'est-à-dire l'exécution de plusieurs commandes en séquence, fonctionnent correctement. Par exemple, utilisez la commande "ls | grep .txt" pour filtrer les fichiers avec l'extension ".txt" dans le répertoire courant.

Test des signaux : La gestion des signaux est efficace en envoyant des signaux tels que SIGINT (Ctrl+C) ou SIGTSTP (Ctrl+Z) pendant l'exécution d'une commande.

Test des commandes en arrière-plan : Les commandes en arrière-plan fonctionnent correctement en exécutant une commande avec le symbole "&" à la fin.