Rapport-final-SEC

EL ALOUT ISMAIL

Département Sciences du Numérique - Première année
2021-2022

1 Question 1

J'ai créé une boucle infinie dans laquelle on lit la commande saisie et ensuite un processus fils est créé.

2 Question 2

On peut voir sur le fichier Q2.pdf le problème rencontré lors de l'éxécution du programme. En effet, le processus père n'attend pas la fin de l'éxécution du processus fils.

3 Question 3

Pour résoudre le problème de la question 2, j'ai utilisé la primitive wait() pour que le père attende la terminaison du processus fils.

4 Question 4

Pour ajouter les commandes internes demandées dans cette question, j'ai utilisé la primitive chdir() pour ajouter cd et la primitive exit() pour ajouter la commande interne exit.

5 Question 5

Dans cette question, j'ai vérifié s'il s'agit d'une commande en tâche de fond grace au parseur fourni readcmd ; si c'est le cas le processu est ajouté dans la liste des processus (nommée liste sur le programme). Pour mieux gérer la liste des processus, j'ai créé un module nommé listeProc. J'ai défini toutes les fonctions qui gère la liste des processus (ajout d'un processus à la liste, suppression d'un processus de la liste, obtenir le pid d'un processus...) sur le fichier listeProc.c (interface : listeProc.h).

6 Question 6

Pour l'implantation de la question 6, j'ai utilisé la fonction afficher que j'ai défini dans le module listeProc pour donner la liste des processus lancées depuis le terminal. Ensuite, j'ai créé 3 sous-programmes dans le fichier minishell.c pour implanter les 3 autres commandes interne (exec_cmd_sj, exec_cmd_bg et exec_cmd_fg).

7 Question 7

Pour cette question, j'ai défini le traitant gérant le signal SIGTSTP (handler_SIGTSTP). La conception a été réalisée de la manière suivante : une frappe ctrl+z provoque l'envoi du signal SIGTSTP au processus courant grace à la primitive kill(). Ce signal est redirigé vers les processus en cours en avant-plan et est masquée pour les processus en background.

8 Question 8

Comme pour la question précédente, j'ai défini le traitant gérant le signal SIGINT (handler_SIGINT). Pour chaque entrée dans la boucle, on traite ce signal et on le redirige vers le processus en avant-plan, puis on le masque pour les processus en tâche de fond.

9 Question 9

Pour associer l'entrée standard ou la sortie standard d'une commande à un fichier, j'ai utilisé dup2 pour dupliquer le descripteur du fichier à l'entrée ou à la sortie ; la différence entre les 2 cas est que pour l'association de l'entrée standard à un fichier, le fichier est ouvert en mode lecture, alors que pour l'association de la sortie standard le fichier est ouvert en mode écriture.

10 Questions 10 et 11

Pour ces 2 questions, j'ai créé une pipe et j'ai créé un processus fils et un sous-fils. Le sous-fils va exécuter la première commande et le fils va exécuter le reste de la commande.

11 Quelques tests réalisés

On lance des commandes simples.

```
ielalout@n7-ens-lnx037:~/1A/S6/SEC/Projet$ ./minishell
ismail@elalout /home/ielalout/1A/S6/SEC/Projet ls
f1                  ielalout_etapes1_5.tar  listeProc.c  Q1          readcmd.h
f2                  ielalout_etapes6        listeProc.h  q6.c        try
fournitures         ielalout_etapes6.tar    minishell    q9
f.tar               ielalout.tar            minishell5   q9.c
ielalout            LisezMoi.html           minishell6   Question1.c
ielalout_etapes1_5  LisezMoi.md             minishell.c  readcmd.c
ismail@elalout /home/ielalout/1A/S6/SEC/Projet echo test
test
```

On lance des commandes avec cd et exit.

```
ismail@elalout /home/ielalout/1A/S6/SEC/Projet cd ..
ismail@elalout /home/ielalout/1A/S6/SEC cd Projet
ismail@elalout /home/ielalout/1A/S6/SEC/Projet exit
ielalout@n7-ens-lnx037:~/1A/S6/SEC/Projet$
```

On lance une commande en tâche de fond.

```
ismail@elalout /home/ielalout/1A/S6/SEC/Projet sleep 40&
ismail@elalout /home/ielalout/1A/S6/SEC/Projet [1] 0
ls
f1                  ielalout_etapes1_5.tar  listeProc.c  Q1          readcmd.h
f2                  ielalout_etapes6        listeProc.h  q6.c        try
fournitures         ielalout_etapes6.tar    minishell    q9
f.tar               ielalout.tar            minishell5   q9.c
ielalout            LisezMoi.html           minishell6   Question1.c
ielalout_etapes1_5  LisezMoi.md             minishell.c  readcmd.c
ismail@elalout /home/ielalout/1A/S6/SEC/Projet
```

On liste les processus courants avec la commande lj.

```
ismail@elalout /home/ielalout/1A/S6/SEC/Projet sleep 45&
ismail@elalout /home/ielalout/1A/S6/SEC/Projet [1] 0
lj
 id         pid         état            commande
0      0      En cours      ♦♦♦♦♦♦♦♦♦p/d@Vsleep 45
ismail@elalout /home/ielalout/1A/S6/SEC/Projet
```

On peut suspendre un processus en avant-plan avec ctrl+z ou ctrl+c.

```
ielalout@n7-ens-lnx037:~/1A/S6/SEC/Projet$ ./minishell
ismail@elalout /home/ielalout/1A/S6/SEC/Projet sleep 50
^Z[27670] suspendu
ismail@elalout /home/ielalout/1A/S6/SEC/Projet
```

```
ielalout@n7-ens-lnx037:~/1A/S6/SEC/Projet$ ./minishell
ismail@elalout /home/ielalout/1A/S6/SEC/Projet sleep 50
^C[28695] suspendu
```

On ne peut suspendre un processus en background avec ctrl+c ou ctrl+z.

```
ismail@elalout /home/ielalout/1A/S6/SEC/Projet sleep 50&
ismail@elalout /home/ielalout/1A/S6/SEC/Projet [1] 0
^C^C^C^C^Z^Z^Z
```

Gestion des redirections

```
ismail@elalout /home/ielalout/1A/S6/SEC/Projet ls | wc -l
26
ismail@elalout /home/ielalout/1A/S6/SEC/Projet
```