COMPTE RENDU
PROJET 1 – PROGSYS

Christopher Marie-Angélique – Groupe 2B

Table des matières

Introduction

Cette première partie du projet de Programmation Système consiste en la mise en place d'un minishell. Notre programme Python qui lit des lignes de commandes sera capable de les lancées en exécution.

Partie 1 : Minishell

Code source :

```python
import os

cmd = input('$ ')

while cmd != 'exit':

    if cmd.strip() != '':

        args = cmd.split(' ')

        if args[-1] == '&':

            args = args[:-1]
            background = True
        else:

            background = False


        pid = os.fork()

        if pid:

            if not background:

                codes = os.wait()

                while codes[0] != pid:

                    codes = os.wait()

                exit_code = codes[1] // 256

                print(f'Processus {codes[0]} terminé avec le code de retour {exit_code}')

        else:

            try:

                os.execvp(args[0], args)

            except OSError as e:

                print('Erreur lors de l\'appel :', e.strerror)

                os._exit(e.errno)


    cmd = input('$ ')
```

Explication :

En premier lieu, ce programme fonctionnera tant que la commande « exit » n'est pas entrée par l'utilisateur, Si celle-ci est entrée par l'utilisateur, le programme se ferme.

Par la suite, nous vérifions que la commande saisie par l'utilisateur contient bien des arguments, soit n'est pas vide. Si c'est le cas, nous récupérons la liste de ces arguments en les séparant lorsqu'il y'a un espace.

Si le dernier argument de la commande est « & », on le supprime de la liste pour éviter les éventuels problèmes que cela peut causer et nous initialisons une variable booléenne « background » à True qui indique que la commande sera exécutée en background ou en foreground en fonction de sa valeur.

Par la suite, nous créons un sous-programme qui s'occupera de l'exécution.

Une vérification est effectuée pour s'assurer que nous sommes le processus appelant. Si c'est la cas le programme attend que le processus fils se termine. Par la suite, dans un « try catch » nous essayons de lancer la commande demandée par l'utilisateur et si une exception se présente, nous l'indiquons à l'utilisateur et renvoyons au processus père le code d'erreur.