Patrick Felschen, Matr. Nr. 932056

Julian Voß, Matr. Nr. 934505

Aufgabenblatt 5

Aufgabe 1

Die *execute* Funktion wurde durch eine Abfrage erweitert. Hier wird in der Nutzereingabe nach dem Pipe-Symbol „|“ gesucht. Ist das Zeichen vorhanden, wird die Nutzereingabe in zwei Arrays abgespeichert. *Args1* beinhaltet somit den Befehl und die Argumente links vom Pipe-Symbol und *args2* den Befehl und Argumente rechts vom Symbol.

```
if (strstr(line, "|") != nullptr) {

    char **pipes = splitPipe(line);
    char **args1 = splitArgs(pipes[0]);
    char **args2 = splitArgs(pipes[1]);

    launchPipe(args1, args2);

}
```

Die *launchPipe* Funktion erstellt zwei geschachtelte Kindprozesse. Die Ausgabe vom ersten Kindprozess dient hier als Eingabe für den zweiten Kindprozess. Die Kommunikation der beiden Prozesse wird über eine *pipe* hergestellt. Der erste Kindprozess greift auf den schreibenden Filedeskriptor (1) zu um produzierte Ausgaben an den zweiten Kindprozess zu leiten. Dieser verwendet den lesenden Filedeskriptor (0) um die von Prozess 1 erstellte Ausgabe weiterzuverarbeiten. Währenddessen wartet der Elternprozess auf die Beendigung der beiden Prozesse. Die Terminierung wird über das Schließen der Filedeskriptoren gekennzeichnet.

```
void MiniShell::launchPipe(char **args1, char **args2) {
    int pipefd[2];
    pid_t p1, p2;

    if (pipe(pipefd) < 0) {
        cerr << "Pipe konnte nicht erstellt werden!" << endl;
    return;
    }

    p1 = fork();
    if (p1 == 0) {
        // Kindprozess 1
        close(pipefd[0]);
        dup2(pipefd[1], STDOUT_FILENO);
        close(pipefd[1]);
    } else {
    // Elternprozess
    p2 = fork();
    if (p2 == 0) {
        // Kindprozess 2
```

```
        close(pipefd[1]);
        dup2(pipefd[0], STDIN_FILENO);
        close(pipefd[0]);
    } else {
        // Warten auf Beendigung der Child-Prozesse
        wait(nullptr);
        return;
    }
}
```

Aufgabe 2

a)

Befehl: yes | (sleep 10; cat -n)

Bash: Unendliche Ausgabe von „y“ auf der Kommandozeile

Minishell: „Befehl 2 konnte nicht ausgefuhert werden“

b)

Befehl: tail | cat -n

Bash: Keine Ausgabe auf der Kommadozeile

Minishell: Keine Ausgabe auf der Kommadozeile

c)

Befehl: kill <PID Kindprozess> (getestet mit yes | cat -n)

Minishell: Sobald ein Kindprozess beendet wird, wird der andere Kindprozess ebenfall beendet und der Hauptprozess läuft mit dem nächsten Schleifendurchlauf weiter.

d)

Befehl: kill <PID Elternprozess> (getestet mit yes | cat -n)

Bash: Prozess wird nicht beendet

Minishell: Prozess inkl. Kindprozess wird beendet

e)

Reihenfolge des Pipe-Datenverkehrs ändern.