Assignment 1

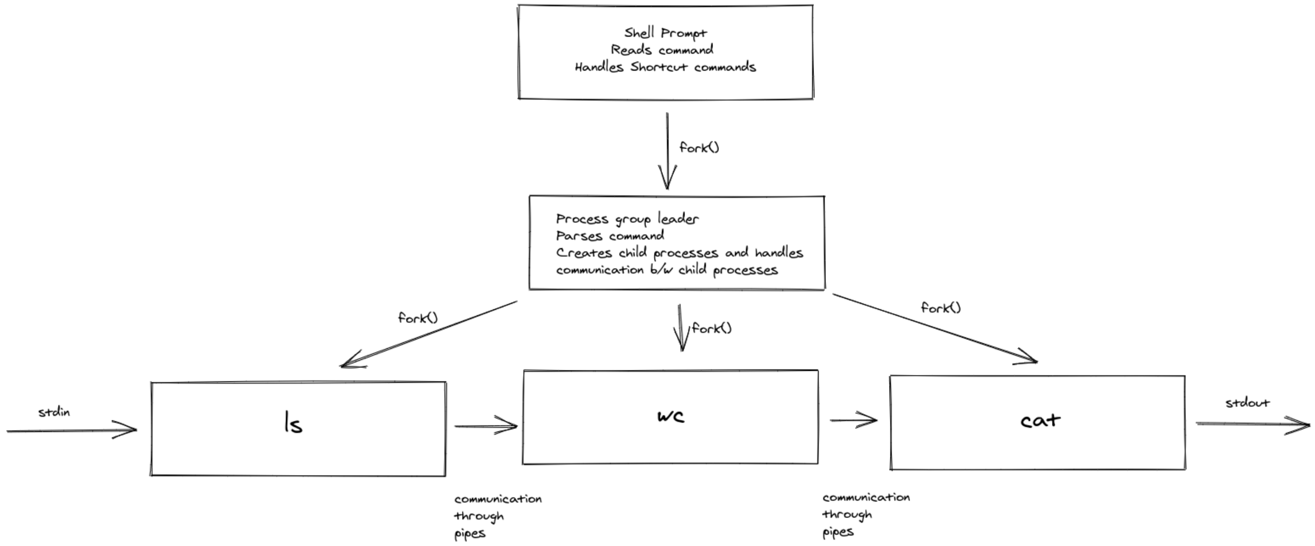
Bash like Shell

Submitted To - Dr. Hari Babu K.
ID NOs - 2018A7PS0219P, 2018A7PS0166P

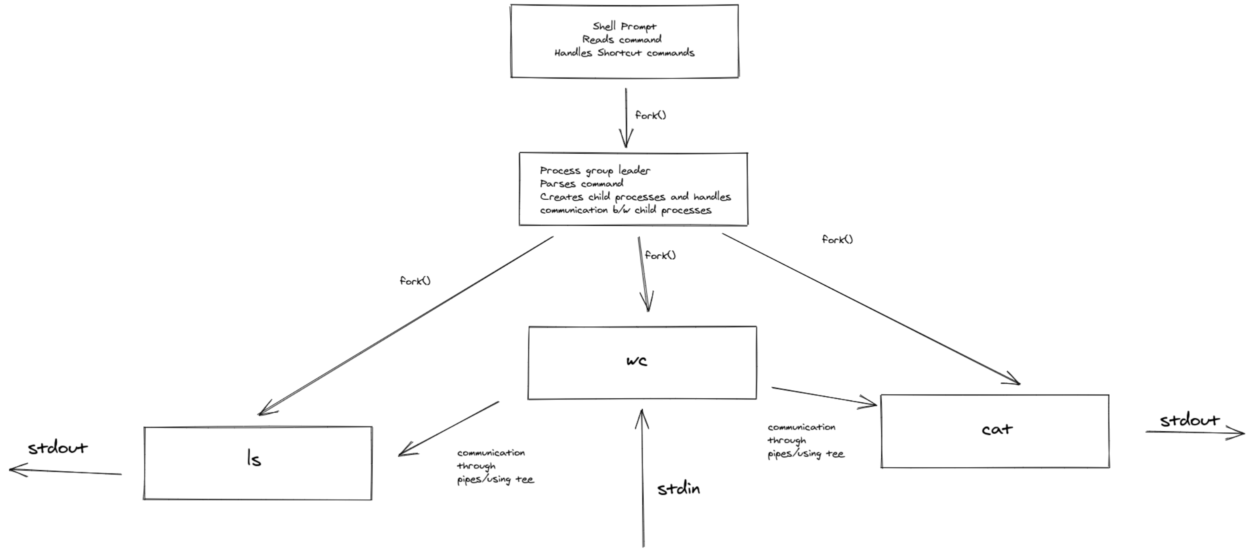
Design

This exercise develops a bash-like shell which includes features like chaining process via pipes, input-output redirection, some new features like double piping, triple piping and shortcut mode.

The design for chaining processes via pipes is shown in the figure below :-



The design for double piping commands is shown in the figure below :-



Shell:

The main shell process is a prompting process, which shows the prompt and waits for the user to enter the command. It also looks for SIGINT signal and goes to shortcut mode and waits for the user to then enter a digit to run the corresponding saved shortcut. Shell also recognises shortcut insertion and deletion commands and updates the lookup table for the commands accordingly.

On receiving the command, it passes the command to a new child process and then makes this process leader of a new process group. The shell also checks if the command is foreground or background and sets the foreground terminal control accordingly. Main shell then waits for the process if it is a foreground process and regains control over the terminal.

The leader first parses the command. The main leader process creates child processes for the sub commands and executes them using fork() and execv() and communicate between each other using pipes and tee in case of double piping and triple piping.

Shortcut Command:

When a shell receives sc -i <index> <cmd> or sc -d <index> cmd, it adds or deletes the entry corresponding in the lookup table implemented using singly linked lists. Insertion is done at the beginning of the table. When a SIGINT signal is received by the shell it goes to the shortcut mode and accepts the user to enter an integer which corresponds to an already added command and if there is no command shell gives an error. After getting the integer shell just basically searches the command in the lookup table and executes that command as described above.

Double Piping and Triple Piping:

Double piping and Triple piping is achieved by using the tee() system call which takes in two file descriptors and copies the content of one to another.

How to Run:

```
gcc shell.c -o shell.o
./shell.o
```

To exit the shell you can press Ctrl + Z and then kill the process from the background using the kill command from bash.

Assumptions:

- At most, only one ‘||’ or ‘|||’, not both.
- There can be only input redirection in a single command (remaining input redirections ‘<’ are ignored), and only one output redirection (either ‘>’ or ‘>>’, not both).
- If there is any I/O redirection is any of the piped commands, the pipe is broken and the redirection is given higher priority.