Objectives:
The project objectives and value allows us to understand the inner workings of a shell. It helped us understand the commands needed to change path or directory, run programs, terminate programs and much more. I also learned we can make complex commands through multiple in-built shell commands such as the check function in this assignment. I used a while loop to keep the program running and cues for the command of exit and terminate to break the loop. Inside of the while loop consist of function calls according to the command given by the user. The functions such as run program, terminate, exit, stop, continue and others are called outside of the while loop.

Design Overview:
cdir pathname - Change directory by specifying pathname.

pdir - Print the current working directory.

lstasks - Lists all the task currently running where index, pid and the command is displayed.

run pgm - The command allows the execution of a task through forking and the maximum number of tasks are 32. The use of execvp, execlp and execvp is shown.

stop taskNo - Allows a certain task to be stopped using the signal SIGSTOP and the process id.

continue taskNo - Allows a certain task to be continued after being stopped, the signal SIGCONT and the process id are used to continue the stopped task.

terminate taskNo - Allows a certain running task to be terminated using the signal SIGKILL and the process id.

check target_pid - Displays user, pid, ppid, state, start, command of a certain process. popen was used to get the information of all the processes.

exit - Use the terminate command to terminate existing and running processes and exit of the main loop.

quit - Exit the main loop without terminating any processes.

Project status:
I had to add sh to every myclock process in the mMyclock script to make the run successful. Stop and continue for mMyclock does not work smoothly and termination of mMyclock only results in the termination of mMyclock process but not the child processes. I also had difficulty in parsing the processes from the popen file stream, but it is working smoothly. I also had to use a couple for loops to print the child of mMyclock and also the child of myclock so it is quite messy but works smoothly.

Testing and Results:
For every function that is created, I made tested edge cases and tried cases that is invalid to see if the function would still run. I also made sure to pass all the test cases given in eclass.

Acknowledgement:
https://stackoverflow.com/questions/437802/how-to-portably-convert-a-string-into-an-uncommon-integer-type

https://stackoverflow.com/questions/347949/how-to-convert-a-stdstring-to-const-char-or-char

https://stackoverflow.com/questions/45202379/how-does-popen-work-and-how-to-implement-it-into-c-code-on-linux/45308042