Details of Implementation

I tokenize and parse the input string in main function and then call command handler function with args array, background status ,redirection type and redirection file name. Command handler has two different types of commands: built-ins and externals. Myshell just has two built-in commands. They are bello and exit. If the given command is not built-in then it iterates over the path environment variable and looks for the executable. If it exists then it will be executed via fork exec mechanism. Myshell uses execv() from exec function family.

Shell also has aliasing mechanism. Aliases are replaced after parsing and before calling command handller in main function. Aliases are kept in a linked list of struct and also stored in a file in order to be able to utilize them even after the termination of the shell.

To achieve redirections, I change file descriptors with dup2() function. For reverse redirection operator, I use grandchild process. The grandchild handles the command and writes into a temp file and then terminates. The child waits its termination and then read its outputs, reverse them and writes into the actual file.

I have a function which prints the content “bello” command must print. For the name of the current shell, it looks at the folder with the pid of the parent process (the shell we call myshell) under the *proc directory. After that, the name of the process is accessed from its status file. For the number of processes being executed, it look at proc/$pid/task/$pid/children file and repeat recursively to find grandchildren. Other information “bello” prints was straightforward.*

Difficulties Encountered

Handling dynamic memory allocation errors (segmentation fault) was challenging.