OS161 DVZ Progress report 2

I. SYSTEM CALL PROGRESS OUT OF 5:

`__getcwd` : 4-5 | `__chdir` : 5 | `execv` : 3-4 | `_exit` : 5 | `read` : 5 | `write` : 5 | `open` : 5 | `close` : 5 | `fork` : 5 | `getpid`: 5 | `waitpid`: 5 | `lseek`: 5 | `dup2`: 5

II. EXPLANATION OF SYSCALL STATUS:

- Fully complete, no other notes:
  `__chdir, exit, read, write, open, close, fork, getpid, waitpid, lseek, dup2`

- Not yet complete but partially working:
  - `Execv` is implemented in logic but fails to work in testing, strange stack issues, likely caused by changing stack slightly incorrectly. `Execv` works for only non argument functions/commands. Arguments give it issues. `Execv` calls an edited version of runprogram w/ args and the number of args is implemented in logic but doesn't work in testing, also due to argument copying issues. Adjustments in the stack have small flaws in logic when it comes to shifting the stack and rounding towards a multiple of 4.
  - `__getcwd` currently has issues but there is an issue with a dependent file, `emu.c`.
  - Scheduler got the main idea but not fully tested (2 out of 5), in branch `vud_scheduling` not merged to `main` yet. Set up extra parameters in thread structure to bookkeeping CPU's stat time like burst_time, wait_time, arrival_time, exit_time and implemented the `schedule()` function from the pseudocode from the paper with the Equitable Shortest Job First scheduling algorithm. Still need to implement helper functions from threadlist to sort the run queue by the shortest remaining time and test the scheduler.

III. TESTS THAT PASS:

- `Badcall`: all pass except `execv`
- `Bigseek, Closetest, Fileonlytest, Filetest, Forkbomb, Forktest, F_test, Palin, Readwritetest, Shelltest, Shll, Tictac, shell`

IV. RESPONSIBILITIES:

- John: `open, close, read, write, fork`, ⅓ of backend
- Emma: `lseek, dup2, waitpid, getpid,` scheduler (in `vud_scheduling` branch), ⅓ of backend
- Eric: `execv, getcwd, chdir, exit`, ⅓ of backend

We worked independently and then merge in `GeneralJD` branch so John can test and review to make sure everything works instead of merging directly to `main` and causing the `main` to break like the last time. By doing this method we don't have to resolve merge conflicts and duplicated code from merging like the last time