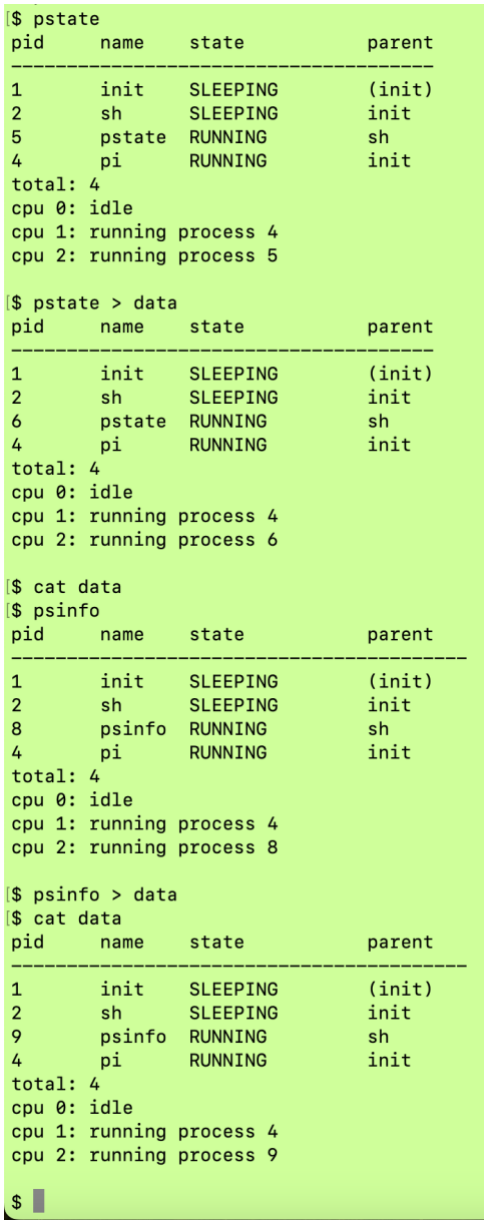
Getting Process Info from Kernel

Implement system call psinfo() and the corresponding user program psinfo.c so that users can use shell command psinfo to retrieve the same process information that shell command pstate provides. psinfo() however cannot use kernel/printf.c to directly print the process status. Instead, it should pass the information to the user program.

Note that the I/O redirection in shell commands works in psinfo but not in pstate.

Make your own decision on the arguments of this system call, if needed.

Sample run:



```
$ pstate
pid     name    state           parent
----------------------------------------
1       init    SLEEPING        (init)
2       sh      SLEEPING        init
5       pstate  RUNNING         sh
4       pi      RUNNING         init
total: 4
cpu 0: idle
cpu 1: running process 4
cpu 2: running process 5

$ pstate > data
pid     name    state           parent
----------------------------------------
1       init    SLEEPING        (init)
2       sh      SLEEPING        init
6       pstate  RUNNING         sh
4       pi      RUNNING         init
total: 4
cpu 0: idle
cpu 1: running process 4
cpu 2: running process 6

$ cat data
$ psinfo
pid     name    state           parent
----------------------------------------
1       init    SLEEPING        (init)
2       sh      SLEEPING        init
8       psinfo  RUNNING         sh
4       pi      RUNNING         init
total: 4
cpu 0: idle
cpu 1: running process 4
cpu 2: running process 8

$ psinfo > data
$ cat data
pid     name    state           parent
----------------------------------------
1       init    SLEEPING        (init)
2       sh      SLEEPING        init
9       psinfo  RUNNING         sh
4       pi      RUNNING         init
total: 4
cpu 0: idle
cpu 1: running process 4
cpu 2: running process 9

$
```

Submission: a zipped file firstname_lastname.zip consisting of all modified xv6 files (including Makefile) and all new files (including at least one file for testing, like pi.c).