Section 3
October 19th

**How to load user memory from disk:**
First need to call `setupkvm` to create a new page table. The page table will have the kernel portion mapped already. Call load_program_from_disk to load the program and get the program counter.

**How to setup user stack:**
Use `initustack`. It sets up one page at the bottom of 2G to be a user stack. Also it sets up a guard page above ustack to ensure xk can capture stack overflow failure.

**How to pass argument to an exec'ed process:**
Put arguments on user stack (in the newly allocated page table) so the loaded process has access to it. This will require moving data from the current page table (in `%cr3`) into one in memory that isn't loaded (see `copyout()`). You can assume one page allocated by `initustack` is enough to store user arguments. You can also assume there are no more than 32 (`MAX_ARG`, inc/params.h) arguments passed into the user program. Set `%rdi` and `%rsi` for the first two arguments in the process's trap frame. Remember to set up the trap frame to allow the process to start executing when it gets scheduled (instruction pointer, stack pointer, etc.)

**Memory Region Layout:**

| |
|---|
| .... |
| STACK (4KB) at 2GB-4KB |
| Guard Page (4KB) at 2GB-8KB |
| ... |
| ... |
| HEAP (Page aligned) |
| CODE |

**Note:**
Look at the Piazza note: "Clarification on exec" To see an in depth explanation of required actions.

Section 3
October 19th