1. Is /proc a file system? We used one of the functions in the second project (cat /proc/$pid/maps); please enumerate five other functions that you like most. In addition to peeking (i.e., reading) kernel secrets, can you also write to /proc? If so, please give one example.

/proc is a virtual filesystem. It contains runtime system information (e.g. system memory, devices mounted, hardware configuration, etc) rather than actual "files." For this reason, it can be regarded as a control and information center for the kernel. In fact, quite a lot of system utilities are simply calls to files in this directory.

In addition to /proc/$pid/maps, here are five other functions:

- /proc/$pid/cmdline
    - Command line arguments.
- /proc/$pid/cpu
    - Current and last cpu in which it was executed.
- /proc/$pid/cwd
    - Link to the current working directory.
- /proc/$pid/environ
    - Values of environment variables.
- /proc/$pid/exe
    - Link to the executable of this process.

You can write to a file in /proc using a command such as:
echo "HelloWorld" > /proc/helloworld.txt

2. What are the uses of Loadable Kernel Modules? Is it possible to implement Project 2 (i.e., reading through the user space of a given process) using LKM?

Loadable Kernel Modules are typically used for device drivers, filesystem drivers, and adding system calls. Since Loadable Kernel Modules can be used to add system calls, you could implement Project 2 using LKM since it requires you to implement a new system call.