1. Add a second reason to locate functionality outside the kernel.

* flexibility – easier to change without breaking the system call interface
* \_\_

2. Add a second reason to locate functionality inside the kernel.

* safety – better protection because validity checks cannot be skipped
* \_\_

3. Add a second benefit of a “thin waist” for the system call interface. (“Thin waist” describes the design principle of decreasing the points of contact between the inside of the OS and the outside. See Figure 3.2.)

* provides a **simple, stable** interface
  + ease of understanding for programmer (including an easier learning curve)
  + ease of portability
* \_\_

4. Add a second advantages of the “open before use” approach to file I/O. (In networking this design approach is called a **connection-oriented interface**. This is when we expect **multiple interactions** and thus initially establish a connection to reduce the overhead on each of those interactions.)

* check permissions once for the series of transactions rather than repeating the same check on each transaction
* \_\_

5. What does an explicit call to the close() function accomplish?

6. If there were separate UNIX system calls for exists(), create(), and open(), what could go wrong with

the following code?

if (!exists(name))

create(name);

fd = open(name);

7. Explain how kernel buffering decouples the execution of a producer and consumer process.

8. What is the purpose of the select() system call?

9. Identify a pro and a con of microkernel design.

pro: \_\_

con: \_\_

10. What three functions does the seL4 system implement in the microkernel?

11. What does Linus think of microkernels?

12. What is the benefit of having device drivers run in user mode?