Your Name: _____

1. Resource abstraction serves the following purposes:

- ☐ Enables similar resources to export a common interface.
- ☐ Complicates the application-programming interface.
- ☐ Provides the necessary mathematical underpinning to make the resource definition precise.
- ☐ Enables the system to use Java (and other language abstractions) as an OS interface.
- ☐ Saves a programmer from having to learn the detailed interface to each different resource.
- ☐ All of the above
- ☐ None of the above

2. If you rented a parking space in a downtown area for the hours 8:00 am to 6:00 pm, but the parking lot owner rents the space to other people in the evening, this is an example of

- ○ Space multiplexed sharing of the parking space
- ○ Double dipping (fraud)
- ○ Time multiplexed sharing of the parking space
- ○ All of the above
- ○ None of the above

3. Which of the following statements are true about multiprogramming?

- ☐ The use of threads makes multiprogramming obsolete.
- ☐ It forces the OS to incorporate processor scheduling.
- ☐ It encourages the abstraction of process so that the OS can distinguish among different running programs.
- ☐ It is not used on contemporary personal computers.
- ☐ It was very popular in the 1970s, but is rarely used in contemporary operating systems.

4. How is a resource characterized in an operating system?

- The totality of allocatable hardware units
- Memory, normally held by a parent process/thread
- Anything that a process/thread can request, and the process may be suspended if it is not available
- The remaining CPU time allocation of a process/thread
- CPU time and primary memory.

5. A file descriptor is:

- A POSIX-specific data structure used by the kernel.
- The user's mechanism for specifying actions to be performed on a file.
- A resource abstraction used to implement icons on a desktop.
- An abstract data type for secondary storage devices.
- An OS data structure used to keep the state of file.

6. In a POSIX file system, the *open* command is used to:

- Create an instance of a file descriptor.
- Differentiates between preparing for read use and write use.
- Specifies the path name of a file to be used.
- All of the above.
- None of the above.

7. A thread differs from a lightweight process in that

- Threads share a process's resources but lightweight processes do not.
- There is no difference.
- Threads run in user space, but lightweight threads run in kernel space.
- Lightweight processes execute their own programs.
- Threads have a parent process.