Chapter 1: Operating System Concepts

1. **What are the two main functions of an operating system?**
   An operating system must provide the users with an extended (i.e., virtual) machine, and it must manage the I/O devices and other system resources.

2. **What is multiprogramming?**
   Multiprogramming is the rapid switching of the CPU between multiple processes in memory. It is commonly used to keep the CPU busy while one or more processes are doing I/O.

3. **What is the difference between timesharing and multiprogramming systems?**
   In a timesharing system, multiple users can access and perform computations on a computing system simultaneously using their own terminals. Multiprogramming systems allow a user to run multiple programs simultaneously. All timesharing systems are multiprogramming systems but not all multiprogramming systems are timesharing systems since a multiprogramming system may run on a PC with only one user.

4. **What is spooling? Do you think that advanced personal computers will have spooling as a standard feature in the future?**
   Input spooling is the technique of reading in jobs, for example, from cards, onto the disk, so that when the currently executing processes are finished, there will be work waiting for the CPU. Output spooling consists of first copying printable files to disk before printing them, rather than printing directly as the output is generated. Input spooling on a personal computer is not very likely, but output spooling is.

5. **There are several design goals in building an operating system, for example, resource utilization, timeliness, robustness, etc. Give an example of two design goals that may contradict one another.**
   Consider the goals of fairness and real time. Fairness requires that each process be allocated its resources in a fair way, with no process getting more than its fair share. On the other hand, real time requires that resources be allocated based on the times when different process must complete their execution. A high priority real-time process may get a disproportionate share of the resources.

6. **What is the difference between kernel and user mode? Explain how having two distinct modes aids in designing an operating system.**
   Most modern CPUs provide two modes of execution: kernel mode and user mode. The CPU can execute every instruction in its instruction set and use every feature of the hardware when executing in kernel mode. However, it can execute only a subset of instructions and use only subset of features when executing in the user mode. Having two modes allows designers to run user programs in user mode and thus deny them access to critical instructions.

7. **Which of the following instructions should be allowed only in kernel mode?**
   (a) Disable all interrupts.
   (b) Read the time-of-day clock.
   (c) Set the time-of-day clock.
   (d) Change the memory map.

   Choices (a), (c), and (d) should be restricted to kernel mode.
8. List some differences between personal computer operating systems and mainframe operating systems.
   Personal computer systems are always interactive, often with only a single user. Mainframe systems nearly always emphasize batch or timesharing with many users. Protection is much more of an issue on mainframe systems, as is efficient use of all resources.

9. What is the purpose of a system call in an operating system? Give examples of system calls.
   A system call allows a user process to access and execute operating system functions inside the kernel. User programs use system calls to invoke operating system services.
   - fork: a (UNIX) system call used for creating child processes.
   - open: a system call used for initializing access to a file.
   - kill: a (UNIX) system call used for terminating a process.

10. In order to use a computer system, you have to give commands to its operating system. Name two types of user interfaces used to issue commands to an operating system and explain how each work.
   Your answer

Chapter 2: Processes

1. A computer system has enough room to hold four programs in its main memory. These programs are idle waiting for I/O half the time. What fraction of the CPU time is wasted?
   The chance that all four processes are idle is 1/16, so the CPU idle time is 1/16.

2. A computer has 2 GB of RAM of which the operating system occupies 256 MB. The processes are all 128 MB (for simplicity) and have the same characteristics. If the goal is 99% CPU utilization, what is the maximum I/O wait that can be tolerated?
   There is enough room for 14 processes in memory. If a process has an I/O of \( p \), then the probability that they are all waiting for I/O is \( p^{14} \). By equating this to 0.01, we get the equation \( p^{14} = 0.01 \). Solving this, we get \( p = 0.72 \), so we can tolerate processes with up to 72% I/O wait.

3. What is a race condition?
   A race condition is a situation in which two (or more) processes are about to perform some action involving some shared data. Depending on the exact timing, one or the other goes first. If one of the processes goes first, everything works, but if another one goes first, a fatal error occurs.

4. A computer has enough room to hold three processes in its main memory. First process spends 40% of its time for CPU utilization This measure is 50% and 60% for second and third processes, respectively.
   a) Calculate overall wasted time (I/O wait time). Your answer
   b) What is the CPU utilization in this case? Your answer

5. Calculate overall CPU utilization if the three processes located in memory spend 20%, 40%, and 60% of their time in memory for CPU utilization, respectively.
   Your answer

6. What is a process? What does a process consist of?
   A process is basically a program in execution. It consists of the executable program, the program’s data and stack, program counter, stack pointer, and other registers and all the other information needed to run the program.
7. Name the three process states. Explain all possible transitions among these states.  
   Your answer

8. List four conditions that are necessary for multiple processes to cooperate correctly and efficiently using shared data.
   1. No two processes may be simultaneously inside their critical sections.
   2. No assumptions may be made about speeds or the number of CPUs.
   3. No process running outside its critical section may block other processes.
   4. No process should have to wait forever to enter its critical section.