Introduction

Practice Exercises

1.1 What are the three main purposes of an operating system?
Answer: The three main puropses are:

- To provide an environment for a computer user to execute programs on computer hardware in a convenient and efficient manner.

- To allocate the separate resources of the computer as needed to solve the problem given. The allocation process should be as fair and efficient as possible.

- As a control program it serves two major functions: (1) supervision of the execution of user programs to prevent errors and improper use of the computer, and (2) management of the operation and control of I/O devices.

1.2 What are the main differences between operating systems for mainframe computers and personal computers?
Answer: Generally, operating systems for batch systems have simpler requirements than for personal computers. Batch systems do not have to be concerned with interacting with a user as much as a personal computer. As a result, an operating system for a PC must be concerned with response time for an interactive user. Batch systems do not have such requirements. A pure batch system also may have not to handle time sharing, whereas an operating system must switch rapidly between different jobs.

1.3 List the four steps that are necessary to run a program on a completely dedicated machine—a computer that is running only that program.
Answer: The four steps are:

a. Reserve machine time.

b. Manually load program into memory.

c. Load starting address and begin execution.

      d.   Monitor and control execution of program from console.

1.4 We have stressed the need for an operating system to make efficient use of the computing hardware. When is it appropriate for the operating system to forsake this principle and to "waste" resources? Why is such a system not really wasteful?

  Answer: Single-user systems should maximize use of the system for the user. A GUI might "waste" CPU cycles, but it optimizes the user's interaction with the system.

1.5 What is the main difficulty that a programmer must overcome in writing an operating system for a real-time environment?

  Answer: The main difficulty is keeping the operating system within the fixed time constraints of a real-time system. If the system does not complete a task in a certain time frame, it may cause a breakdown of the entire system it is running. Therefore when writing an operating system for a real-time system, the writer must be sure that his scheduling schemes don't allow response time to exceed the time constraint.

1.6 Consider the various definitions of *operating system*. Next, consider whether the operating system should include applications such as Web browsers and mail programs. Argue both that it should and that it should not, and support your answers.

  Answer: **Point**. Applications such as web browsers and email tools are performing an increasingly important role in modern desktop computer systems. To fulfill this role, they should be incorporated as part of the operating system. By doing so, they can provide better performance and better integration with the rest of the system. In addition, these important applications can have the same look-and-feel as the operating system software.

  Counterpoint. The fundamental role of the operating system is to manage system resources such as the CPU, memory, I/O devices, etc. In addition, it's role is to run software applications such as web browsers and email applications. By incorporating such applications into the operating system, we burden the operating system with additional functionality. Such a burden may result in the operating system performing a less-than-satisfactory job at managing system resources. In addition, we increase the size of the operating system thereby increasing the likelihood of system crashes and security violations.

1.7 How does the distinction between kernel mode and user mode function as a rudimentary form of protection (security) system?

  Answer: The distinction between kernel mode and user mode provides a rudimentary form of protection in the following manner. Certain instructions could be executed only when the CPU is in kernel mode. Similarly, hardware devices could be accessed only when the program is executing in kernel mode. Control over when interrupts could be enabled or disabled is also possible only when the CPU is in kernel mode. Consequently, the CPU has very limited capability when executing in user mode, thereby enforcing protection of critical resources.

1.8 Which of the following instructions should be privileged?

    a. Set value of timer.

    b. Read the clock.

    c. Clear memory.

    d. Issue a trap instruction.

    e. Turn off interrupts.

    f. Modify entries in device-status table.

    g. Switch from user to kernel mode.

    h. Access I/O device.

Answer: The following operations need to be privileged: Set value of timer, clear memory, turn off interrupts, modify entries in device-status table, access I/O device. The rest can be performed in user mode.

1.9 Some early computers protected the operating system by placing it in a memory partition that could not be modified by either the user job or the operating system itself. Describe two difficulties that you think could arise with such a scheme.

Answer: The data required by the operating system (passwords, access controls, accounting information, and so on) would have to be stored in or passed through unprotected memory and thus be accessible to unauthorized users.

1.10 Some CPUs provide for more than two modes of operation. What are two possible uses of these multiple modes?

Answer: Although most systems only distinguish between user and kernel modes, some CPUs have supported multiple modes. Multiple modes could be used to provide a finer-grained security policy. For example, rather than distinguishing between just user and kernel mode, you could distinguish between different types of user mode. Perhaps users belonging to the same group could execute each other's code. The machine would go into a specified mode when one of these users was running code. When the machine was in this mode, a member of the group could run code belonging to anyone else in the group.

Another possibility would be to provide different distinctions within kernel code. For example, a specific mode could allow USB device drivers to run. This would mean that USB devices could be serviced without having to switch to kernel mode, thereby essentially allowing USB device drivers to run in a quasi-user/kernel mode.

1.11 Timers could be used to compute the current time. Provide a short description of how this could be accomplished.

Answer: A program could use the following approach to compute the current time using timer interrupts. The program could set a timer for some time in the future and go to sleep. When it is awakened by the interrupt, it could update its local state, which it is using to keep track of the number of interrupts it has received thus far. It could then repeat this process of continually setting timer interrupts and updating its local state when the interrupts are actually raised.

1.12 Is the Internet a LAN or a WAN?

Answer: The Internet is a WAN as the various computers are located at geographically different places and are connected by long-distance network links.

Exercises:

1.13 In a multiprogramming and time-sharing environment, several users share the system simultaneously. This situation can result in various security problems.
a. What are two such problems?
b. Can we ensure the same degree of security in a time-shared machine as in a dedicated machine? Explain your answer.
Answer:
a. Stealing or copying one's programs or data; using system resources (CPU, memory, disk space, peripherals) without proper accounting.
b. Probably not, since any protection scheme devised by humans can inevitably be broken by a human, and the more complex the scheme, the more difficult it is to feel confident of its correct implementation.

1.14 The issue of resource utilization shows up in different forms in different types of operating systems. List what resources must be managed carefully in the following settings:
a. Mainframe or minicomputer systems
b. Workstations connected to servers
c. Handheld computers
Answer:
a. Mainframes: memory and CPU resources, storage, network bandwidth.
b. Workstations: memory and CPU resources.
c. Handheld computers: power consumption, memory resources.

1.15 Under what circumstances would a user be better off using a timesharing system rather than a PC or single-user workstation?
Answer: When there are few other users, the task is large, and the hardware is fast, time-sharing makes sense. The full power of the system can be brought to bear on the user's problem. The problemcan be solved faster than on a personal computer. Another case occurs when lots of other users need resources at the same time.
A personal computer is best when the job is small enough to be executed reasonably on it and when performance is sufficient to execute the program to the user's satisfaction.

1.16 Which of the functionalities listed below need to be supported by the operating system for the following two settings: (a) handheld devices and (b) real-time systems.
a. Batch programming
b. Virtual memory
c. Time sharing
Answer: For real-time systems, the operating system needs to support virtual memory and time sharing in a fair manner. For handheld systems, the operating system needs

1.17 Describe the differences between symmetric and asymmetric multiprocessing. What are three advantages and one disadvantage of multiprocessor systems?
Answer: Symmetric multiprocessing treats all processors as equals, and I/O can be processed on any CPU. Asymmetric multiprocessing has one master CPU and the remainder CPUs are slaves. The master distributes tasks among the slaves, and I/O is usually done by the master only. Multiprocessors can save money by not duplicating power supplies, housings, and peripherals. They can execute programs more quickly and can have increased reliability. They are also more complex in both hardware and software than uniprocessor systems.

1.18 How do clustered systems differ from multiprocessor systems? What is required for two machines belonging to a cluster to cooperate to provide a highly available service?
Answer: Clustered systems are typically constructed by combining multiple computers into a single system to perform a computational task distributed across the cluster. Multiprocessor systems on the other hand could be a single physical entity comprising of multiple CPUs. A clustered system is less tightly coupled than a multiprocessor system. Clustered systems communicate using messages, while processors in a multiprocessor system could communicate using shared memory.
In order for two machines to provide a highly available service, the state on the two machines should be replicated and should be consistently updated. When one of the machines fail, the other could then take-over the functionality of the failed machine.

1.19 Distinguish between the client–server and peer-to-peer models of distributed systems.
Answer: The client-server model firmly distinguishes the roles of the client and server. Under this model, the client requests services that are provided by the server. The peer-to-peer model doesn't have such strict roles. In fact, all nodes in the system are considered peers and thus may act as *either* clients or servers - or both. A node may request a service from another peer, or the node may in fact provide such a service to other peers in the system. For example, let's consider a system of nodes that share cooking recipes. Under the client-server model, all recipes are stored with the server. If a client wishes to access a recipe, it must request the recipe from the specified server. Using the peer-to-peer model, a peer node could ask other peer nodes for the specified recipe. The node (or perhaps nodes) with the requested recipe could provide it to the requesting node. Notice how each peer may act as both a client (i.e. it may request recipes) and as a server (it may provide recipes.)

1.20 Consider a computing cluster consisting of two nodes running a database. Describe two ways in which the cluster software can manage access to the data on the disk. Discuss the benefits and disadvantages of each.

Answer: Consider the following two alternatives: **asymmetric clustering** and **parallel clustering**. With asymmetric clustering, one host runs the database application with the other host simply monitoring it. If the server fails, the monitoring host becomes the active server. This is appropriate for providing redundancy. However, it does not utilize the potential processing power of both hosts. With parallel clustering, the database application can run in parallel on both hosts. The difficulty implementing parallel clusters is providing some form of distributed locking mechanism for files on the shared disk.

1.21 How are network computers different from traditional personal computers? Describe some usage scenarios in which it is advantageous to use network computers.

Answer: A network computer relies on a centralized computer for most of its services. It can therefore have a minimal operating system to manage its resources. A personal computer on the other hand has to be capable of providing all of the required functionality in a standalone manner without relying on a centralized manner. Scenarios where administrative costs are high and where sharing leads to more efficient use of resources are precisely those settings where network computers are preferred.

1.22 What is the purpose of interrupts? What are the differences between a trap and an interrupt? Can traps be generated intentionally by a user program? If so, for what purpose?

Answer: An interrupt is a hardware-generated change-of-flow within the system. An interrupt handler is summoned to deal with the cause of the interrupt; control is then returned to the interrupted context and instruction. A trap is a software-generated interrupt. An interrupt can be used to signal the completion of an I/O to obviate the need for device polling. A trap can be used to call operating system routines or to catch arithmetic errors.

1.23 Direct memory access is used for high-speed I/O devices in order to avoid increasing the CPU´s execution load.
a. How does the CPU interface with the device to coordinate the transfer?
b. How does the CPU know when the memory operations are complete?
c. The CPU is allowed to execute other programs while the DMA controller is transferring data. Does this process interfere with the execution of the user programs? If so, describe what forms of interference are caused.

Answer: The CPU can initiate a DMA operation by writing values into special registers that can be independently accessed by the device. The device initiates the corresponding operation once it receives a command from the CPU. When the device

is finished with its operation, it interrupts the CPU to indicate the completion of the operation. Both the device and the CPU can be accessing memory simultaneously.

The memory controller provides access to the memory bus in a fair manner to these two entities. A CPU might therefore be unable to issue memory operations at peak speeds since it has to compete with the device in order to obtain access to the memory bus.

1.24 Some computer systems do not provide a privileged mode of operation in hardware. Is it possible to construct a secure operating system for these computer systems? Give arguments both that it is and that it is not possible.

Answer: An operating system for a machine of this type would need to remain in control (or monitor mode) at all times. This could be accomplished by two methods:

a. Software interpretation of all user programs (like some BASIC, Java, and LISP systems, for example). The software interpreter would provide, in software, what the hardware does not provide.

b. Require meant that all programs be written in high-level languages so that all object code is compiler-produced. The compiler would generate (either in-line or by function calls) the protection checks that the hardware is missing.

1.25 Give two reasons why caches are useful. What problems do they solve? What problems do they cause? If a cache can be made as large as the device for which it is caching (for instance, a cache as large as a disk), why not make it that large and eliminate the device?

Answer: Caches are useful when two or more components need to exchange data, and the components perform transfers at differing speeds. Caches solve the transfer problem by providing a buffer of intermediate speed between the components. If the fast device finds the data it needs in the cache, it need not wait for the slower device. The data in the cache must be kept consistent with the data in the components. If a component has a data value change, and the datum is also in the cache, the cache must also be updated. This is especially a problem on multiprocessor systems where more than one process may be accessing a datum. A component may be eliminated by an equal-sized cache, but only if: (a) the cache and the component have equivalent state-saving capacity (that is, if the component retains its data when electricity is removed, the cache must retain data as well), and (b) the cache is affordable, because faster storage tends to be more expensive.

1.27 Discuss, with examples, how the problem of maintaining coherence of cached data manifests itself in the following processing environments:

a. Single-processor systems

b. Multiprocessor systems

c. Distributed systems

Answer: In single-processor systems, the memory needs to be updated when a processor issues updates to cached values. These updates can be performed

immediately or in a lazy manner. In a multiprocessor system, different processors might be caching the same memory location in its local caches. When updates are made, the other cached locations need to be invalidated or updated. In distributed systems, consistency of cached memory values is not an issue. However, consistency problems might arise when a client caches file data.

1.28 Describe a mechanism for enforcing memory protection in order to prevent a program from modifying the memory associated with other programs.
Answer: The processor could keep track of what locations are associated with each process and limit access to locations that are outside of a program's extent. Information regarding the extent of a program's memory could be maintained by using base and limits registers and by performing a check for every memory access.

1.29 What network configuration would best suit the following environments?
a. A dormitory floor
b. A university campus
c. A state
d. A nation
Answer:
a. **A dormitory floor** - A LAN.
b. **A university campus** - A LAN, possible a WAN for very large campuses.
c. **A state** - A WAN.
d. **A nation** - A WAN.

1.30 Define the essential properties of the following types of operating systems:
a. Batch
b. Interactive
c. Time sharing
d. Real time
e. Network
f. Parallel
g. Distributed
h. Clustered
i. Handheld
Answer:
a. **Batch**. Jobs with similar needs are batched together and run through the computer as a group by an operator or automatic job sequencer. Performance is increased by attempting to keep CPU and I/O devices busy at all times through buffering, off-line operation, spooling, and multiprogramming. Batch is good for executing large jobs that need little interaction; it can be submitted and picked up later.

b. **Interactive**. This system is composed of many short transactions where the results of the next transaction may be unpredictable. Response time needs to be short (seconds) since the user submits and waits for the result.

c. **Time sharing**. This systems uses CPU scheduling and multiprogramming to provide economical interactive use of a system. The CPU switches rapidly from one user to another. Instead of having a job defined by spooled card images, each program reads its next control card from the terminal, and output is normally printed immediately to the screen.

d. **Real time**. Often used in a dedicated application, this system reads information from sensors and must respond within a fixed amount of time to ensure correct performance.

e. **Network**. Provides operating system features across a network such as file sharing.

f. **SMP**. Used in systems where there are multiple CPU's each running the same copy of the operating system. Communication takes place across the system bus.

g. **Distributed**. This system distributes computation among several physical processors. The processors do not share memory or a clock. Instead, each processor has its own local memory. They communicate with each other through various communication lines, such as a high-speed bus or local area network.

h. **Clustered**. A clustered system combines multiple computers into a single system to perform computational task distributed across the cluster.

i. **Handheld**. A small computer system that performs simple tasks such as calendars, email, and web browsing. Handheld systems differ from traditional desktop systems with smaller memory and display screens and slower processors.

1.31 What are the tradeoffs inherent in handheld computers?

Answer: Handheld computers are much smaller than traditional desktop PC's. This results in smaller memory, smaller screens, and slower processing capabilities than a standard desktop PC. Because of these limitations, most handhelds currently can perform only basic tasks such as calendars, email, and simple word processing. However, due to their small size, they are quite portable and, when they are equipped with wireless access, can provide remote access to electronic mail and the World Wide Web.