Operating system 5 marks old question paper answer key

MAY 2021

PART B — (5 × 5 = 25 marks)

Answer any FIVE questions.

13. Explain about System Components

14. What is system boot process? Explain.

15. Explain about Segmentation scheme.

16. Illustrate the priority scheduling algorithm.

17. State the conditions to occur deadlock.

18. Explain the goals of I/O Management.

19. What is mutual Exclusion? Explain.

Answers

13. **System Components:**
A computer system comprises several essential components that work together to enable its operation.

- **Central Processing Unit (CPU):** The CPU is the "brain" of the computer, responsible for executing instructions and performing calculations.

- **Memory (RAM):** Memory is the temporary storage that holds data and program instructions for quick access by the CPU.

- **Input/Output Devices:** These include devices like keyboards, mice, monitors, and printers, allowing users to interact with the computer and receive output.

- **Storage Devices:** These are used for long-term data storage, such as hard drives or SSDs, which store programs and data.

- **Motherboard:** The motherboard connects and facilitates communication between all hardware components.

- **Operating System (OS):** The OS manages hardware resources, runs software applications, and provides user interfaces.

- **Software:** Software includes applications and system programs that instruct the computer to perform specific tasks.

14. **System Boot Process:**
The system boot process is the sequence of actions a computer takes when it's powered on or restarted to prepare it for use. It typically involves the following steps:

- **Power-On Self-Test (POST):** The system's firmware checks hardware components to ensure they are functioning correctly.

- **Boot Loader:** The BIOS or UEFI firmware loads a boot loader, such as GRUB in Linux or NTLDR in Windows.

- **Operating System Kernel Load:** The boot loader loads the OS kernel into memory.

- **Initialization:** The kernel initializes hardware, mounts file systems, and starts essential system services.

- **User Space Initiation:** The OS proceeds to start user space processes and user interfaces.

15. **Segmentation Scheme:**
Segmentation is a memory management scheme where memory is divided into segments, each representing a logical unit. These segments can have different sizes and are allocated dynamically to processes. It provides flexibility and simplifies memory protection and sharing. For example, a program's code, data, and stack can each be in separate segments. Segmentation simplifies address space management and enhances protection.

16. **Priority Scheduling Algorithm:**
Priority scheduling assigns priorities to processes, and the CPU is given to the process with the highest priority. It can be implemented using various algorithms, such as preemptive or non-preemptive priority scheduling. In preemptive priority scheduling, a higher-priority process can interrupt a lower-priority process. The algorithm ensures that higher-priority tasks are executed first, optimizing task completion based on importance.

17. **Conditions for Deadlock:**
Deadlock occurs when four conditions are met simultaneously:

- Mutual Exclusion: Processes must request exclusive access to resources.

- Hold and Wait: Processes hold resources while waiting for others.

- No Preemption: Resources cannot be forcibly taken from a process.

- Circular Wait: A cycle of processes exists, each waiting for a resource held by the next.

18. **Goals of I/O Management:**
The goals of I/O management include:

- Efficiency: Ensuring that I/O operations are performed as efficiently as possible to minimize delays.

- Reliability: Ensuring data integrity and minimizing the risk of data loss during I/O.

- Abstraction: Providing a high-level interface to I/O devices, simplifying application development.

- Security: Controlling access to I/O devices and data for security purposes.

19. **Mutual Exclusion:**
Mutual exclusion is a synchronization concept in concurrent programming that ensures that only one process or thread can access a critical section of code at a time. This is achieved to prevent data corruption or conflicts when multiple processes attempt to access shared resources simultaneously. Techniques like semaphores and mutexes are commonly used to implement mutual exclusion, allowing processes to safely access shared resources one at a time. It plays a crucial role in maintaining data consistency in multi-process or multi-threaded environments.

MAY 2022

PART B — (5 × 5 = 25 marks)

Answer any FIVE questions.

13. Explain the various types of operating system services.

14. Describe the features of system calls and system programs.

15. Discuss on the operating on processes.

16. Explain the characterisation of deadlock and method of handling it.

17. Describe the paging memory management scheme.

18. Explain the concept of files.

19. Describe the directory structures

Answers

13. **Various Types of Operating System Services:**
Operating systems provide a wide range of services to both users and application programs. These services can be categorized into several types:

- **Program Execution:** The OS loads programs into memory and schedules them for execution.

- **I/O Operations:** It manages input and output devices, allowing processes to read and write data.

- **File System Manipulation:** The OS provides services for creating, deleting, reading, and writing files.

- **Communication Services:** It facilitates communication between processes through mechanisms like message passing or shared memory.

- **Error Detection and Handling:** The OS monitors system activities for errors and handles them gracefully.

- **User Interface Services:** It offers various interfaces for user interaction, including command-line, graphical, and web-based interfaces.

- **Security and Access Control:** The OS enforces user access permissions and ensures data security.

- **Networking Services:** It manages network connections and supports network protocols.

- **Resource Allocation:** The OS allocates resources like CPU time, memory, and I/O devices to processes.

- **Accounting and Performance Monitoring:** It tracks resource usage and system performance.

14. **Features of System Calls and System Programs:**
System calls are interfaces through which user-level programs interact with the operating system. System programs are provided by the OS to help users utilize system resources efficiently. Features of system calls and programs include:

- **Abstraction:** They provide high-level abstractions, hiding complex system details.

- **Standardization:** System calls have standard interfaces, making them portable across different OSes.

- **Isolation:** They provide controlled access to the underlying hardware, ensuring system stability.

- **Resource Management:** System calls manage resources like memory, files, and devices.

- **Error Handling:** They return error codes for handling exceptions and errors.

- **Security:** They implement access control mechanisms to ensure system and data security.

- **Efficiency:** System calls are optimized for efficient resource utilization.

15. **Operating on Processes:**
Operating on processes involves managing and controlling the execution of processes within the operating system. Key operations include process creation, termination, scheduling, and communication. The OS provides mechanisms to create new processes, manage their resources, schedule them for execution, and enable inter-process communication, allowing them to work together or independently. Process management is crucial for efficient resource utilization and multitasking.

16. **Characterization of Deadlock and Handling Methods:**
Deadlock is a situation where multiple processes are unable to proceed because they are waiting for each other to release resources. Characterization of deadlock involves identifying the four necessary conditions: mutual exclusion, hold and wait, no preemption, and circular wait. Handling methods include prevention, where one or more of these conditions are eliminated; avoidance, where resources are allocated with caution; and recovery, where processes are terminated or resources forcibly released to resolve the deadlock.

17. **Paging Memory Management Scheme:**
Paging is a memory management scheme that divides physical memory into fixed-sized blocks called pages and logical memory into fixed-sized blocks called frames. The OS maintains a page table to map pages to frames. Paging provides efficient memory allocation and eliminates external fragmentation. It simplifies address space management and enables non-contiguous memory allocation for processes.

18. **Concept of Files:**
A file is a logical unit for data storage in a computer. It represents data or information and can be organized into various formats, such as text, binary, or multimedia. Files can be created, read, written, and deleted, and they provide a means to store and retrieve data. File systems manage files and their attributes, and files can be organized into directories or folders to create a hierarchical structure for efficient data organization and retrieval.

19. **Directory Structures:**
Directory structures are used to organize and manage files on storage devices. They create a hierarchy of directories or folders, providing a structured way to store and locate files. Common directory structures include tree structures, where directories can contain subdirectories and files, and linear structures, where files are stored in a linear sequence. Directory structures enable efficient organization and access to files and are a fundamental component of file systems.

DECEMBER 2022

PART B — (5 × 5 = 25 marks)

Answer any FIVE questions.

13. Explain about the protection and security in OS.

14. Write notes on simple structure of OS.

15. Write about synchronization.

16. Write about the methods for handling deadlocks.

17. Explain about segmentation.

18. Explain about file attributes.

19. Explain about Kernel I/O.

Answers

13. **Protection and Security in OS:**
Protection and security are critical aspects of an operating system. Protection mechanisms ensure that one process cannot interfere with the execution of other processes or access their data. Security mechanisms prevent unauthorized access, safeguard data integrity, and protect the system against threats. Key components of protection and security include access control lists (ACLs), user authentication, encryption, firewalls, antivirus software, and intrusion detection systems. The OS enforces user permissions, controls access to system resources, and monitors system activities to prevent unauthorized access or malicious actions. Effective protection and security measures are essential for maintaining system integrity and user privacy.

14. **Simple Structure of OS:**
The simple structure of an operating system is characterized by its minimalistic design, suitable for small-scale or embedded systems. It typically consists of a kernel, which manages essential hardware and provides basic services, and a few system programs. This design is lightweight, efficient, and easy to maintain. However, it lacks the advanced features and scalability found in complex operating systems. Simple OS structures are often used in devices like embedded systems, IoT devices, or single-purpose systems where a full-featured OS is not necessary.

15. **Synchronization:**
Synchronization is a fundamental concept in operating systems that ensures orderly and coordinated execution of multiple processes or threads. It prevents issues like data races and deadlocks. Synchronization mechanisms include semaphores, mutexes, and condition variables. These tools enable processes to communicate, cooperate, and access shared resources safely. Synchronization ensures that only one process can access a critical section of code at a time, preventing data corruption and ensuring data consistency in multi-process or multi-threaded environments.

16. **Methods for Handling Deadlocks:**
Handling deadlocks involves several methods:

- **Prevention:** Modify the system to prevent at least one of the four necessary conditions for deadlock (mutual exclusion, hold and wait, no preemption, circular wait).

- **Avoidance:** Use algorithms like Banker's algorithm to ensure safe resource allocation, avoiding deadlock.

- **Detection:** Allow deadlocks to occur and use algorithms to detect and recover from them. Processes may be terminated or resources forcibly released.

- **Timeouts:** Implement timeout mechanisms to break potential deadlocks if processes wait too long.

- **Resource Allocation Graph:** Use a graph-based model to detect and resolve deadlocks by finding circular wait conditions.

17. **Segmentation:**
Segmentation is a memory management scheme where memory is divided into segments, each representing a logical unit or address space. Segments can have different sizes and are allocated dynamically to processes. This provides flexibility in managing memory and simplifies memory protection and sharing. Segmentation can be used for both code and data, making it easier to expand or contract memory areas dynamically.

18. **File Attributes:**
File attributes are properties associated with files in a file system. These attributes include file name, file type, size, location, creation date, modification date, owner, access permissions, and file content. File attributes help manage and track files, and they are used by the operating system to enforce security, access control, and efficient data retrieval.

19. **Kernel I/O:**
Kernel I/O refers to input/output operations managed by the operating system's kernel. The kernel interacts with hardware devices to perform I/O tasks, including reading from and writing to storage devices, handling network communication, and managing peripheral devices like keyboards and printers. Kernel I/O ensures efficient and secure communication between applications and hardware, abstracting complex hardware details to provide a simplified interface for user-level programs. This abstraction enhances system stability, security, and resource management, allowing applications to perform I/O operations without direct hardware interaction.

DECEMBER 2020

PART B — (5 × 5 = 25 marks)

Answer any FIVE questions.

13. Discuss about the System Programs.

14. Explain about the I/O System Management.

15. Explain about any one CPU Scheduling algorithm.

16. Explain the Paging Scheme with example.

17. Explain about the access methods in file.

18. Explain the basic concept of Segmentation.

19. Discuss the various methods for implementing directory.

Answers

13. **System Programs:**
System programs are a vital part of an operating system and serve various essential functions. These programs are separate from the operating system kernel and are designed to perform specific system-related tasks. Some common types of system programs include:

- **File Management Programs:** These programs allow users to create, copy, delete, and manipulate files and directories. Examples include file copy, delete, and rename utilities.

- **I/O Programs:** They handle input and output operations, ensuring efficient data transfer between the computer and its peripherals, such as disk drives, printers, and networks.

- **Communication Programs:** These support communication between users or processes. Examples include email clients and instant messaging applications.

- **Utility Programs:** Utility programs perform tasks like disk cleanup, data compression, and antivirus scanning.

- **Text Editors:** Text editors enable users to create, edit, and manipulate text-based documents.

- **Shells:** Shells provide a command-line interface for users to interact with the operating system, issuing commands and managing processes.

- **Compiler and Interpreters:** These translate high-level programming code into machine code and allow for its execution.

System programs simplify user interaction with the computer, ensuring efficient resource management, data manipulation, and system maintenance.

14. **I/O System Management:**
I/O System Management is a critical component of the operating system responsible for managing input and output operations. It ensures efficient data transfer between the CPU and I/O devices. Key functions of I/O System Management include:

- **Device Independence:** It abstracts hardware details, allowing applications to perform I/O operations without knowledge of specific devices.

- **Buffering:** Data is often transferred between devices and memory through buffers to optimize data flow and prevent bottlenecks.

- **Caching:** Frequently used data is stored in cache memory to improve response time and reduce the need for frequent disk access.

- **Error Handling:** The system must detect and handle errors during I/O operations to maintain data integrity.

- **Concurrency Control:** The system manages concurrent access to I/O devices to prevent data corruption.

- **Device Drivers:** Specific device drivers are used to facilitate communication between the OS and hardware devices.

Efficient I/O management is crucial for overall system performance and user satisfaction.

15. **CPU Scheduling Algorithm - Round Robin:**
Round Robin is a simple CPU scheduling algorithm that assigns each process an equal time quantum for execution. Once a process exhausts its time quantum, it's moved to the back of the queue, and the next process in line gets a turn. Round Robin ensures fairness in process execution and is suitable for time-sharing systems. However, it may not be efficient for long-running or CPU-bound processes, as frequent context switches can lead to overhead.

16. **Paging Scheme with Example:**
Paging is a memory management scheme that divides physical memory into fixed-sized blocks called frames and logical memory into fixed-sized blocks called pages. For example, consider a system with 16 frames (Frame 0 to Frame 15) and 32 pages (Page 0 to Page 31). The page table maps pages to frames. When a process references Page 5, the page table indicates that Page 5 is located in Frame 10. This mapping allows for efficient memory allocation and management.

17. **Access Methods in Files:**
Access methods define how data is read from and written to files. Common access methods include:

- **Sequential Access:** Data is read or written in a linear, sequential manner. This method is efficient for tapes and some types of files.

- **Direct Access:** Data is accessed by specifying a unique identifier or address, such as a record number. It's commonly used for indexed files or databases.

- **Indexed Access:** Data is accessed using an index or key, which points to the location of the desired data within the file.

18. **Basic Concept of Segmentation:**
Segmentation is a memory management scheme that divides memory into segments, where each segment represents a logical unit, such as a code segment, data segment, or stack segment. Each segment can have its own size, permissions, and address space. Segmentation simplifies memory protection and sharing by isolating different parts of a program. For example, an application's code and data segments can be protected from unauthorized access, and data sharing can be controlled at the segment level.

19. **Methods for Implementing Directory:**
Various methods can be used to implement directory structures in file systems. These methods include:

- **Single-Level Directory:** In this method, all files are stored in a single directory with unique names.

- **Two-Level Directory:** Each user has a separate directory, and users can have files with the same name in their directories.

- **Tree-Structured Directory:** Directories are organized hierarchically, allowing for subdirectories. This structure is similar to a file system with folders.

- **Acyclic-Graph Directory:** Directories are structured as a directed acyclic graph, allowing for shared subdirectories among users.

The choice of directory structure depends on the requirements of the operating system and user organization.

DECEMBER 2021

PART B — (5 × 5 = 25 marks)

Answer any FIVE questions.

13. Give the view of Operating System as Resource Management.

14. What is the purpose of system calls? Explain.

15. Explain the classic problems of Synchronization.

16. What are the methods involved in recovery from deadlock?

17. Write about the techniques for structuring the page table.

18. Explain the basic concepts of Segmentation.

19. Explain the various file access methods.

Answers

13. **Operating System as Resource Management:**
One of the fundamental views of an operating system is as a resource management tool. In this context, the OS acts as an intermediary between the computer hardware and the software applications running on it. It manages and allocates various resources to ensure efficient and fair utilization. Key resources managed by the OS include:

- **CPU:** The operating system schedules processes to run on the CPU, ensuring that each gets a fair share of processing time.

- **Memory:** It allocates and manages memory space for processes, loading and unloading them as needed.

- **I/O Devices:** The OS controls input and output operations, including data transfer between peripherals and memory.

- **File System:** It manages files and directories, ensuring data integrity and providing access control.

- **Network Resources:** The OS controls network communication, ensuring proper data exchange between devices.

By effectively managing these resources, the operating system ensures that applications can run concurrently, access necessary data and hardware, and communicate with users and other systems efficiently.

14. **Purpose of System Calls:**
System calls are the interface through which user-level programs request services from the operating system. They serve several purposes:

- **Abstraction:** System calls abstract complex low-level operations, allowing users and applications to interact with the OS in a standardized, high-level manner.

- **Resource Access:** They enable programs to access system resources such as files, devices, and memory.

- **Control:** System calls allow processes to interact with the operating system, controlling their execution and resource usage.

- **Error Handling:** They provide a mechanism for error reporting and handling, ensuring robust and reliable application behavior.

- **Security:** System calls implement access control mechanisms, ensuring data security and protecting the system from unauthorized access.

Overall, system calls simplify application development, enhance security, and provide a consistent and efficient way for software to utilize system resources.

15. **Classic Problems of Synchronization:**
Synchronization problems arise when multiple processes or threads access shared resources concurrently. The classic synchronization problems include:

- **Producer-Consumer Problem:** A scenario where producers add data to a shared buffer, and consumers retrieve data from it.

- **Reader-Writer Problem:** Involves multiple readers and writers accessing a shared data structure. Readers can work concurrently, but writers must have exclusive access.

- **Dining Philosophers Problem:** Models a situation where philosophers sitting around a dining table can think and eat but need to coordinate their actions to avoid conflicts over limited resources (forks).

These problems illustrate the need for synchronization mechanisms like semaphores and mutexes to prevent issues like data corruption and race conditions.

16. **Methods Involved in Recovery from Deadlock:**
Deadlocks can be resolved through various methods:

- **Process Termination:** Terminate one or more processes involved in the deadlock to release resources.

- **Resource Preemption:** Temporarily preempt resources from one process to allocate them to another. Preempted processes may need to be rolled back and restarted.

- **Wait-Die and Wound-Wait Schemes:** These are techniques for controlling resource access based on process priorities and ages. In the Wait-Die scheme, older processes wait, while in the Wound-Wait scheme, younger processes wait.

- **Detection and Recovery:** Use algorithms to detect deadlocks, after which processes can be terminated or resources forcibly released.

Each method has its advantages and trade-offs, and the choice depends on the system's requirements and policies.

17. **Techniques for Structuring the Page Table:**
Page tables are used in paging memory management. Different techniques can be employed to structure page tables, including:

- **Hierarchical Paging:** The page table itself is divided into multiple levels, reducing its size and making it easier to manage.

- **Hashed Page Tables:** A hash table is used to index page table entries, which allows for faster page lookup but can still become large in size.

- **Inverted Page Tables:** Instead of having an entry for each page, an entry is created for each frame, reducing the table size but requiring additional processing to find a page.

The choice of structuring technique depends on the specific memory management requirements and constraints of the operating system.

18. **Basic Concepts of Segmentation:**
Segmentation is a memory management scheme where memory is divided into segments, each representing a logical unit or address space, such as code, data, and stack segments. Segmentation provides several advantages:

- **Memory Protection:** Each segment can have its own access permissions, preventing unauthorized access.

- **Sharing:** Segments can be shared among processes, allowing for code or data sharing.

- **Dynamic Growth:** Segments can be dynamically allocated and resized, adapting to the memory needs of processes.

Segmentation simplifies memory management and protection compared to traditional contiguous memory allocation.

19. **Various File Access Methods:**
File access methods define how data is read from and written to files. Common methods include:

- **Sequential Access:** Data is read or written in a linear, sequential manner, suitable for tapes and some types of files.

- **Direct Access:** Data is accessed by specifying a unique identifier or address, often used for indexed files or databases.

- **Indexed Access:** Data is accessed using an index or key, which points to the location of the desired data within the file.

The choice of access method depends on the type of data and the application's requirements. Sequential access is suitable for linear data, while direct and indexed access provide efficient retrieval for structured or indexed data.