**CIT601 Operating Systems (4 CH) – Week 2 Notes**

**Course**: Operating Systems
**Credit Hours**: 4 CH
**Course Outline**: OS basics, processes, memory management, file systems, basic system administration.
**Assessment**: Labs (30%), midterm (30%), final exam/project (40%)
**Resources**:

- *Operating System Concepts* by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne (Chapter 3 for process management).
- Online tutorials: Search "Process Management" or "Linux Command Line Basics" on Microsoft Learn or YouTube (e.g., freeCodeCamp or The Linux Foundation videos).
- Tools: VirtualBox (free for OS experimentation), Canva (free for creating diagrams).
  **Week 2 Topic**: Process Management and Basic Command-Line Operations
  **Objective**: Understand how operating systems manage processes (running programs) and learn basic command-line operations in Windows and Linux to interact with the OS for practical tasks.

---

**1. Recap of Week 1: Introduction to Operating Systems**

- **Key Concepts**: An operating system (OS) manages hardware and software resources, providing a user interface (e.g., GUI, command line). Functions include process management, memory management, file systems, device management, and security. Common OS types include Windows (desktop) and Linux (open-source).
- **Examples**: Using Windows File Explorer to access files or Linux Terminal to list directories.
- **Week 2 Focus**: Dive into process management (how the OS handles running programs) and introduce basic command-line operations in Windows (Command Prompt) and Linux (Terminal) for practical system interaction.

---

**2. Overview of Process Management**

**Process management** is the OS's ability to manage running programs, called **processes**, which are instances of programs in execution (e.g., a web browser or text editor). The OS ensures processes run efficiently, share resources, and don't conflict.

- **Core Idea**: A process is like a task the OS juggles, allocating CPU time and memory to each. For example, running Microsoft Word and a browser simultaneously requires the OS to prioritize and schedule them.
- **Importance**:
  - Enables multitasking (running multiple programs at once).
  - Ensures system stability (e.g., prevents crashes from resource conflicts).
  - Optimizes performance (e.g., prioritizes critical tasks).

- **Real-World Applications**:
    - Running multiple apps (e.g., Word, Chrome) in a school lab.
    - Managing server processes for a business website.
    - Prioritizing tasks on a smartphone.

---

**3. Key Concepts of Process Management**

A **process** is a program in execution, including its code, data, and state (e.g., running, waiting). The OS manages processes through several mechanisms.

- **Process States**:
    - **New**: Process is being created.
    - **Running**: Executing on the CPU.
    - **Waiting**: Awaiting resources (e.g., disk I/O).
    - **Ready**: Waiting for CPU allocation.
    - **Terminated**: Process completed or stopped.
- **Process Control Block (PCB)**:
    - A data structure storing process information (e.g., ID, state, memory allocation).
    - Example: The OS uses the PCB to track a browser's memory usage.
- **Scheduling**:
    - The OS decides which process gets CPU time (e.g., round-robin scheduling).
    - Example: Prioritizes a video call over a background download.
- **Context Switching**:
    - The OS switches between processes, saving and restoring their states.
    - Example: Pauses a game to run an antivirus scan.
- **Process Termination**:
    - Ends a process when complete or if it crashes.
    - Example: Closing a frozen app via Task Manager.

**Visual (Text-Based Diagram)**:

[Process Management]

|--> [Process States: New, Running, Waiting, Ready, Terminated]

|--> [PCB: Stores Process ID, State, Memory]

|--> [Scheduling: Allocates CPU Time]

|--> [Context Switching: Switches Processes]

|--> [Termination: Ends Process]

---

**4. Tools for Monitoring Processes**

The OS provides tools to monitor and manage processes, allowing users to view or control running programs.

- **Windows Task Manager**:
    - Access: Ctrl+Shift+Esc or right-click taskbar → Task Manager.
    - Features: View processes, CPU/memory usage, end tasks.
    - Example: End a frozen Notepad process.
- **Linux Top/HTOP**:
    - Access: Open Terminal, type top or htop.
    - Features: Display running processes, CPU/memory usage, kill processes.
    - Example: Use top to monitor a Python script's resource usage.
- **Windows Command Prompt (Tasklist/Taskkill)**:
    - Commands: tasklist (list processes), taskkill /PID <number> (end process).
    - Example: taskkill /PID 1234 to close a program.
- **Linux Kill Command**:
    - Command: kill <PID> (end process by ID).
    - Example: kill 5678 to stop a stuck application.

---

**5. Basic Command-Line Operations**

The **command line** is a text-based interface for interacting with the OS, offering precise control compared to the GUI. Week 2 introduces basic commands in Windows Command Prompt and Linux Terminal.

**5.1 Windows Command Prompt**

- Access: Start → type cmd → Enter.
- **Common Commands**:
    - dir: List files and folders (like ls in Linux).
        - Example: dir shows files in C:\Users.
    - cd <directory>: Change directory.
        - Example: cd Documents navigates to Documents folder.

- mkdir <name>: Create a folder.
    - Example: mkdir Projects creates a Projects folder.
- del <file>: Delete a file.
    - Example: del temp.txt deletes a file.
- tasklist: List running processes.
    - Example: tasklist shows all active programs.
- taskkill /PID <number>: End a process.
    - Example: taskkill /PID 1234 closes a program.

**5.2 Linux Terminal**

- Access: Open Terminal in Linux (e.g., Ubuntu).
- **Common Commands**:
    - ls: List files and folders.
        - Example: ls shows files in the home directory.
    - cd <directory>: Change directory.
        - Example: cd projects navigates to projects folder.
    - mkdir <name>: Create a folder.
        - Example: mkdir assignments creates a folder.
    - rm <file>: Delete a file.
        - Example: rm old.txt deletes a file.
    - ps: List running processes.
        - Example: ps aux shows all processes.
    - kill <PID>: End a process.
        - Example: kill 5678 stops a process.

**Example**: A student uses dir in Command Prompt to list files, creates a folder with mkdir Study, and uses tasklist to check running programs on a Windows PC.

---

**6. Practical Examples**

**Example 1: Managing School Lab Processes**

- **Scenario**: A student notices a slow PC in a lab.
- **Process**:
    1. Opens Task Manager (Ctrl+Shift+Esc).

2. Identifies a high-CPU process (e.g., frozen browser).
3. Ends the process by clicking “End Task.”
4. Uses Command Prompt (tasklist) to confirm the process is gone.

- **Outcome**: Restored PC performance for classwork.

**Example 2: Linux File and Process Management**

- **Scenario**: Organizing files and checking processes on Ubuntu.
- **Process**:
    1. Opens Terminal and types mkdir Projects to create a folder.
    2. Uses ls to verify the folder.
    3. Runs ps aux to list processes.
    4. Uses kill 1234 to stop a stuck Python script.
- **Outcome**: Organized files and resolved a process issue.

**Example 3: Business PC Troubleshooting**

- **Scenario**: A business PC runs multiple apps slowly.
- **Process**:
    1. Opens Command Prompt and runs tasklist to view processes.
    2. Identifies a high-memory app (e.g., accounting software).
    3. Uses taskkill /PID 5678 to close it.
    4. Creates a folder with mkdir Records for organization.
- **Outcome**: Improved performance and organized files.

---

**7. In-Class and Self-Study Exercises**

**In-Class Exercises**:

1. **Process Identification**:
    - In groups, open Task Manager (Windows) or top (Linux) and identify 3 running processes. Write a 50-word description of their purpose.
2. **Command-Line Practice**:
    - Practice dir, cd, and mkdir in Command Prompt or ls, cd, mkdir in Linux Terminal. Discuss steps in pairs.
3. **Process Termination**:
    - Use Task Manager or kill to safely end a non-critical process. Write a 50-word summary of the steps.

**Self-Study Exercises**:

1. **Scenario Analysis**:
    - Write a 150-word description of troubleshooting a slow PC using Task Manager and Command Prompt commands. Include 3 commands.
2. **Command Research**:
    - Research one command (e.g., taskkill, ps). Write a 100-word summary of its use and syntax.
3. **Linux Practice**:
    - Install Ubuntu in VirtualBox. Try 3 commands (e.g., ls, mkdir, ps). Write a 100-word summary.
4. **Process Monitoring**:
    - Use Task Manager or top to monitor CPU/memory usage. Write a 100-word summary of findings.

---

**10. Connection to Course Assessments**

- **Labs (30%)**: The Week 2 diagram assignment contributes to the 30% lab component by teaching process management and command-line skills. Future labs will involve tasks like file system configuration or system administration.
- **Midterm (30%)**: The Week 2 quiz prepares students for midterm questions on process management. Later midterm tasks will include analyzing scenarios (e.g., "Describe how the OS schedules processes").
- **Final Exam/Project (40%)**: Week 2 concepts (processes, commands) will be tested with theoretical questions or a project (e.g., scripting basic OS tasks).

---

**11. Glossary of Key Terms**

- **Process Management**: OS function to handle running programs.
- **Process**: A program in execution (e.g., browser, Word).
- **Process States**: Stages like New, Running, Waiting, Ready, Terminated.
- **Process Control Block (PCB)**: Stores process information (e.g., ID, state).
- **Scheduling**: Allocates CPU time to processes.
- **Context Switching**: Switches between processes.
- **Task Manager**: Windows tool for monitoring processes.
- **Top/HTOP**: Linux tools for process monitoring.
- **Command Line**: Text-based interface for OS interaction.

- **Terminal**: Linux command-line interface.

---

**12. Frequently Asked Questions**

- **What's the difference between a process and a program?**
    - A program is a static file (e.g., chrome.exe); a process is the program running with active resources (e.g., Chrome browser open).
- **How do I create a good diagram for the assignment?**
    - Use Canva's flowchart templates, show process states and 3 commands, and include visuals (e.g., Task Manager screenshot).
- **What if I'm unsure about command-line operations?**
    - Practice dir, mkdir, or ls in Command Prompt/Terminal. Review the examples for guidance.
- **How do I prepare for the quiz?**
    - Study these notes, Chapter 3 of the textbook, and practice commands in Windows or Linux.
- **Why is process management important?**
    - It enables multitasking, ensures stability, and optimizes performance for tasks like studying or business operations.

---

**13. Troubleshooting Tips**

- **Problem**: My diagram lacks detail.
    - **Solution**: Include process states, a tool (e.g., Task Manager), and 3 commands (e.g., dir, taskkill). Check the example for guidance.
- **Problem**: I don't understand process states.
    - **Solution**: Compare to tasks (e.g., Running is a program executing, Waiting is paused for input). List one example for each state.
- **Problem**: My explanation is too short.
    - **Solution**: Describe how process management and commands support a task (e.g., "Task Manager closes frozen apps"). Use the school lab example as a guide.
- **Problem**: I'm stuck on the application question.
    - **Solution**: Think of scenarios like studying or business. Explain how process management improves efficiency.

---

**14. Self-Study and Portfolio Tips**

- **Practice**:
    - Use Task Manager to monitor 3 processes and end one safely.
    - Try 5 commands in Command Prompt (dir, cd) and Linux Terminal (ls, mkdir).
    - Create a practice diagram for process states.
- **Resources**:
    - Read Chapter 3 of *Operating System Concepts* by Silberschatz et al.
    - Watch freeCodeCamp's "Linux Command Line Basics" videos on YouTube.
    - Explore Microsoft Learn's "Windows Command Prompt" tutorials.
    - Use VirtualBox tutorials for Ubuntu setup.
- **Portfolio**:
    - Create a GitHub repository for IT assignments.
    - Upload the diagram PDF and explanation with a README (e.g., "Process Management Diagram").
    - Include a screenshot of Task Manager or Terminal output to show practical skills.
- **Applications**: Consider how process management supports multitasking in schoolwork, programming, or server management to appreciate its impact.

---

**15. Additional Notes for Success**

- **Engagement**: Participate in class discussions to clarify process management or command-line operations. Ask questions during tutorials to deepen understanding.
- **Practice**: Practice commands in Command Prompt or Linux Terminal to build confidence.
- **Preparation for Future Weeks**: Week 2 prepares students for memory management (Week 3) and file systems (Week 4). Mastering processes now will simplify later topics.
- **Portfolio Building**: Treat the diagram and explanation as professional artifacts. A clear design and thorough explanation can showcase technical skills to employers.