SPOS Viva Theory Notes 1. Two Pass Assembler **Definition:** An assembler translates assembly language into machine code in two passes.
Pass-I: Scans source code, builds Symbol Table, Literal Table, and generates Intermediate Code (IC).
Pass-II: Uses IC, Symbol, and Literal Tables to generate actual machine code.
Key Tables: SYMTAB: Stores labels and their addresses. **LITTAB:** Stores literals and their addresses. **MOT:** Machine Operation Table - mnemonics with opcodes. **POT:** Pseudo Operation Table - assembler directives. **Directives:** START, END, DS, DC, EQU, ORIGIN
Advantages: Handles forward references, efficient translation.
Example Question: Explain Pass-I and Pass-II of Assembler with data structures and outputs. 2. Two Pass Macro Processor **Definition:** A macro processor expands user-defined macros before assembly.
Pass-I: Identifies macro definitions and builds: **MNT:** Macro Name Table – stores macro name and MDT index. **MDT:** Macro Definition Table – stores actual macro lines. **ALA:** Argument List Array – stores formal parameters. Removes macro definitions from main code.
Pass-II: Expands macros using the tables created in Pass-I by substituting arguments.
Advantages: Code reusability and modularity.
Common Question: Explain MNT, MDT, and ALA with example. 3. CPU Scheduling Algorithms **Definition:** Determines order in which processes are executed by CPU. **SJF (Preemptive):** Process with shortest remaining burst time executes first. **Priority (Non-Preemptive):** Process with highest priority executes first. **Round Robin:** Each process gets equal time quantum cyclically. **Terms:** WT – Waiting Time TAT – Turnaround Time AT – Arrival Time BT – Burst Time **Key Questions:** Define starvation, preemption, time quantum, context switching. 4. Synchronization Problems **Definition:** Ensuring orderly execution of concurrent processes using semaphores/mutex. **Producer-Consumer:** Manages shared buffer using semaphores. **Reader-Writer:** Controls concurrent access to shared data. **Dining Philosophers:** Demonstrates deadlock and starvation handling. **Concepts:** Mutex, wait(), signal(), deadlock, starvation. 5. Memory Allocation Strategies **Definition:** Decides how to allocate memory blocks to processes. **First Fit:** Allocate to first sufficient block. **Best Fit:** Allocate to smallest block that fits. **Next Fit:** Continue search from last allocation. **Worst Fit:** Allocate to largest block. **Questions:** Compare all fits, define fragmentation, explain internal vs external fragmentation. 6. Page Replacement Algorithms **Definition:** When a page fault occurs, decides which page to replace in memory. **LRU:** Replaces least recently used page. **Optimal:** Replaces page not needed for longest time. **FIFO:** Replaces the oldest loaded page. **Terms:** Page fault, frame, reference string.
Key Question: Which algorithm gives minimum faults and why? (Answer: Optimal) 7. DLL and JNI **Definition:** Dynamic Link Library allows external reusable functions.
JNI: Java Native Interface connects Java with native C/C++ libraries.
Question: Explain purpose of DLL and how it interacts with applications. 8. Important Viva Questions What are the functions of an Assembler? Difference between Assembler, Compiler, and Interpreter. Explain role of Pass-I and Pass-II in Macro Processor. What are the uses of Symbol and Literal Tables? What is Deadlock? Give example. Differentiate between Paging and Segmentation. Define Page Fault. How is it handled? What is Internal and External Fragmentation? Explain Context Switching. What are Semaphores and Mutexes? Explain Preemptive vs Non-Preemptive Scheduling. What is the difference between LRU and Optimal algorithms? 9. Tips for Viva Always mention data structures used in each pass. Write outputs in table format for clarity. Use short, to-the-point definitions. Remember 2–3 examples for each algorithm. Revise definitions of WT, TAT, AT, BT.