Name → AJAY NATH JHA   Roll. No → 2301010170
Course → B.Tech CSE Core
Section → C   Semester → V

Subject :→ Operating System

Assignment-2 :→ Part A : Short Answer Type →

(1.) Address translation converts the logical address generated by the CPU into a physical address using the page table maintained by the OS. The logical address is divided into page number and offset. The page number is used to index the page table to obtain the corresponding frame number. The offset remains unchanged and is combined with the frame number to form the physical address.

Logical Address : [Page No | Offset]

↓ (Page Table Lookup)

Physical Address : [Frame No | Offset]

It allows multiple processes to coexist by giving each process its own page table and isolated logical address space.

(2.) A memory layout with fixed-size partitions can cause internal fragmentation when a process does not fully use the allocated partition, and external fragmentation when free spaces between allocated blocks are too small to fit incoming processes.

To mitigate these issues, a modern OS can use technique:

1. Paging, which removes external fragmentation by using fixed-size frames.
2. Segmentation with paging, which reduces fragmentation while supporting logical divisions.
3. Buddy system, which allocates memory in power-of-two blocks to reduce external fragmentation.

(3.) In a paging-based model, physical memory is divided into fixed-size frames and processes are divided into pages of same size. Each process maintains a page table that maps its pages to available frames in memory. Trade-offs:

1. Memory overhead: Page tables consume additional memory, especially for large address spaces.
2. Speed: Address translation may take longer without hardware support like TLB, but fast with it.
3. Fragmentation: External fragmentation is eliminated; internal fragmentation may occur only within last page.

(4.) When using virtual memory, the OS provides page tables, and hardware mechanisms like the Memory Management Unit (MMU) perform real-time address translation. The Translation Lookaside Buffer (TLB) stores recently used page table entries to speed up translation. Hardware also enforces memory protection through bits that specify read, write and execute permissions. On a page fault, the hardware traps to the OS, which loads the required page from secondary storage and updates the page table.

(5.) Virtual pages and page table size.
Given: Virtual address = 16 bits → 2^{16} = 65536 bytes
• Page Size = 1KB = 1024 bytes
(a). Number of virtual pages = 65536 ÷ 1024 = 64 pages
(b). Page table Size: Entries = 64, Entry size = 2 bytes
Page table size = (64 × 2) = 128 bytes

Name → AJAY NATH JHA
Roll.No → 2301010170

Part - B : Application / Numerical →

(6) Memory Allocation Simulation
(First-Fit, Best-Fit, Worst-Fit) →
Free memory = 1000 KB
$P_1 = 212$, $P_2 = 417$, $P_3 = 112$, $P_4 = 426$

(a) Allocation Steps →

First-Fit :	Best-Fit :	Worst-Fit :
$1000 \to P_1 \to 788$	$1000 \to P_3 \to 888$	$1000 \to P_2 \to 583$
$788 \to P_2 \to 371$	$888 \to P_1 \to 676$	$583 \to P_4 \to 157$
$371 \to P_3 \to 259$	$676 \to P_2 \to 259$	$157 \to P_3 \to 45$
$259 \to P_4$ (fails)	$259 \to P_4$ (fails)	P_1 (fails)
Unused = 259 KB	Unused = 259 KB	Unused = 45 KB

(b) Unused memory →
First-Fit : 259 KB ; Best-Fit : 259 KB, Worst-Fit : 45 KB

(c) Best Utilization :
Worst-Fit provides best utilization as it leaves the smallest amount of unused memory.

(7) Page Replacement Algorithms (FIFO, Optimal, LRU) →
Reference string : 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2
Frames : 3

(a) Page Faults : FIFO → 10, Optimal → 7, LRU → 9

(b) No. of faults : 5

(c) Best performance & Belady's anomaly.
Optimal performs best and does not suffer from Belady's anomaly. FIFO may show the anomaly, while LRU generally avoids it.

(8) Dirty Page Overhead →
Disk write = 10 ms, Memory write = negligible,
Dirty pages = 30% of 1000 = 300 pages.

(a) Overhead : $300 * 10\,ms = 3000\,ms = 3$ Seconds

(b) Optimization : A write-back strategy combined with background cleaning (flushing dirty pages in advance) reduces blocking overhead.

(9.) Autonomous Vehicle Case Study :→

(a) Working Set model and page replacement :
The OS assigns larger or stable working sets to critical tasks such as object detection so they retain the pages they frequently access, preventing thrashing. Less critical processes receive dynamically adjusted working sets so they adapt according to memory availability. A policy like LRU or working-set based replacement ensures that most essential pages for real-time tasks remain in memory.

(b) Suitable memory allocation strategy:
A priority-based dynamic allocation strategy is appropriate. It reserves guaranteed memory for real-time tasks to ensure responsiveness while allowing remaining memory to be shared among non-critical processes: This approach maintains real-time performance while ensuring efficient system-wide memory utilization.