Birla Institute of Technology & Science Pilani, Hyderabad Campus
First Semester 2023-2024
CS F372: Operating Systems
Comprehensive Examination (Regular)

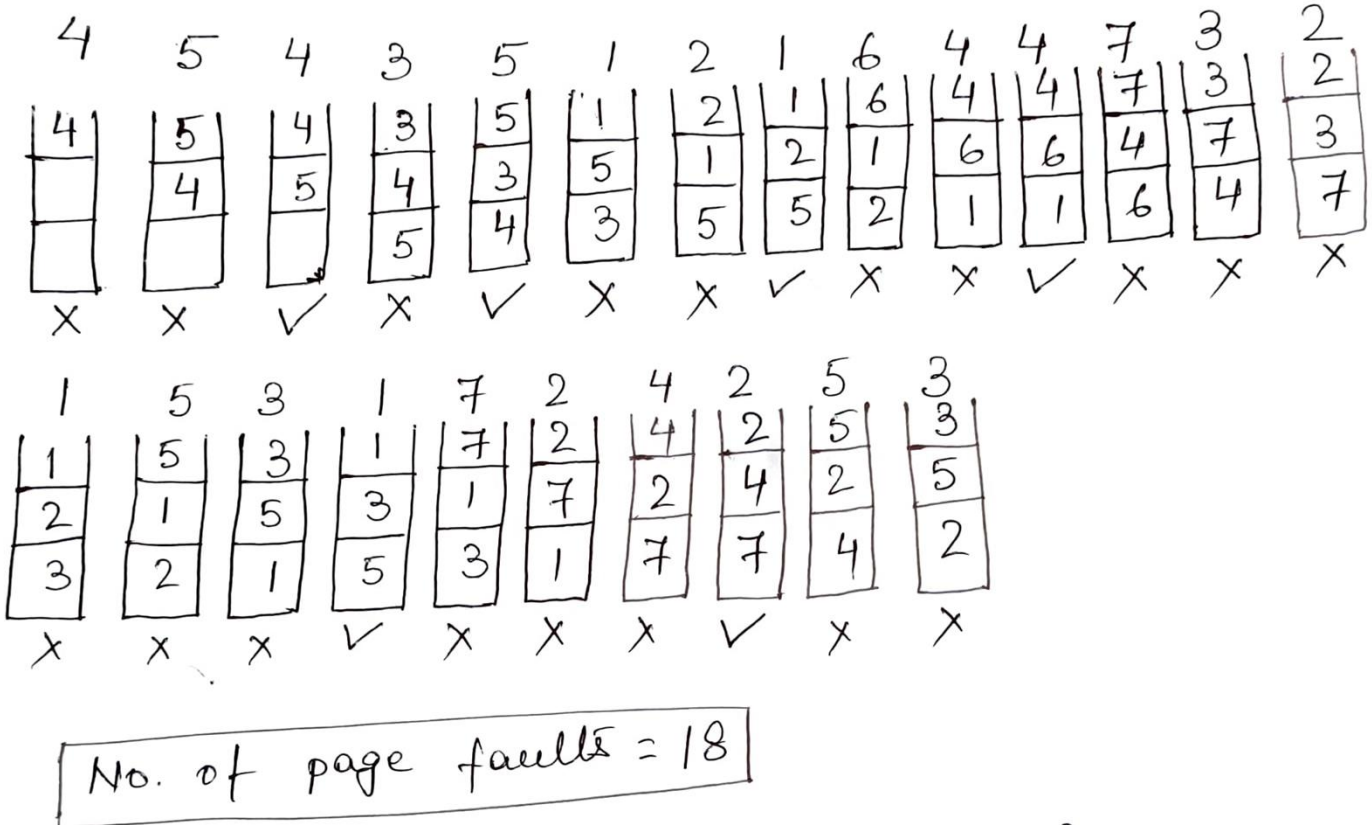
Type: Closed Book **Time: 180 minutes** **Max Marks: 120** **Date: 14/12/2023**

Part B (70 Marks)

---

1. Consider the following sequence of memory references to the given page numbers:
4, 5, 4, 3, 5, 1, 2, 1, 6, 4, 4, 7, 3, 2, 1, 5, 3, 1, 7, 2, 4, 2, 5, 3.
If **LRU** page replacement algorithm is used, determine **the number of page faults** with 3 physical frames. Assuming that the system uses pure demand paging, show how the pages are put/replaced in case of a replacement for a particular page reference. **Assume stack implementation of LRU**. **No marks will be awarded if you do not solve the problem by assuming stack implementation of LRU.** You should update and write the stack contents appropriately for every page reference (both page hits and page faults). Your answer should show all the steps in detail and you need to **explicitly write the number of page faults separately**. **[13 marks]**

Ans.



Note – 0.5 marks for each page reference, 12 marks for all page references. No partial marking for blank or wrong stack contents. Hits and misses may not be marked. No marks will be awarded if stack based approach is not used. 1 mark for writing the no. of page faults.

2. A computer system contains 4 types of resources - **R1**, **R2, R3** and **R4**. A total of 13 instances of **R1**, 18 instances of **R2**, 13 instances of **R3** and 8 instances of **R4** are present in the system. Currently, 5 processes, **P1**, **P2**, **P3, P4** and **P5** are active in the system. The current snapshot of the system is shown in the following table.

Process	**Allocation**				**Max**			
	R1	**R2**	**R3**	**R4**	**R1**	**R2**	**R3**	**R4**
P1	3	5	0	1	12	5	10	3
P2	4	0	2	0	4	4	4	6
P3	0	0	1	1	3	2	4	5
P4	2	4	3	0	7	6	5	6
P5	1	6	3	2	3	8	6	6

i) Determine the contents of the Need and Available data structures for the given system snapshot.

ii) Now **P3** requests for 1 instance of **R1**, 1 instance of **R2** and 2 instances of **R4**. Determine whether **P3's** request can be satisfied now or it should be deferred. If the request can be satisfied, then you need to determine the corresponding safe sequence considering the request and this safe sequence is to be written explicitly. If not, you should give proper explanation. Clearly show all the steps by determining the contents of the relevant data structures at each step and all the conditions that should be checked. Marks will be deducted if you do not write the safe sequence (if any) explicitly as a single list and/or if you miss any step or condition. **[18 marks]**

Ans.

(i) Available = (3, 3, 4, 4) → 1 mark

Note – 0.5 marks to be given if partially correct.

Need

	R1	R2	R3	R4
P1	9	0	10	2
P2	0	4	2	6
P3	3	2	3	4
P4	5	2	2	6
P5	2	2	3	4

→ 5 marks

Note – 1 mark for each row. 0.5 marks to be given for a row if partially correct. Need matrix contents may also be written as vectors like $Need_1 = (9, 0, 10, 2)$, $Need_2 = (0, 4, 2, 6)$.

(ii) $Request_3 = (1, 1, 0, 2)$

$Request_3 \leq Need_3 \rightarrow$ Yes $\rightarrow$ 0.5 marks

$Request_3 \leq Available \rightarrow$ Yes $\rightarrow$ 0.5 marks

Pretend to allocate requested resources to P3.

System state is modified as

Updated Available $= (3,3,4,4) - (1,1,0,2)$ [Available - $Request_3$]

$= (2,2,4,2)$ — 1 mark

Note - 0.5 marks to be given if partially correct. Calculation not required.

Updated $Allocation_3 = (0,0,1,1) + (1,1,0,2)$ [$Allocation_3 + Request_3$]

$= (1,1,1,3)$ — 1 mark

Note - 0.5 marks to be given if partially correct. Calculation not required. Writing entire Allocation matrix is also alright. But only 3rd row will be checked.

Updated Allocation

	R1	R2	R3	R4
P1	3	5	0	1
P2	4	0	2	0
P3	1	1	1	3
P4	2	4	3	0
P5	1	6	3	2

Updated $Need_3 = Need_3 - Request_3$

$= (3,2,3,4) - (1,1,0,2)$

$= (2,1,3,2)$ — 1 mark

Note - 0.5 marks to be given if partially correct. Calculation not required. Writing entire Need matrix is also alright. But only 3rd row will be checked

Updated Need

	R1	R2	R3	R4
P1	9	0	10	2
P2	0	4	2	6
P3	2	1	3	2
P4	5	2	2	6
P5	2	2	3	4

Safety algo. using Bankers

Work = (2, 2, 4, 2)

Step 1: P3 is selected as $Need_3 \leq Work$ — 0.5 marks

Work = (2,2,4,2) + (1,1,1,3)

= (3,3,5,5) — 1mark

Step 2: P5 is selected as $Need_5 \leq Work$ - 0.5 marks

Work = (3,3,5,5) + (1,6,3,2)

= (4,9,8,7) — 1mark

Step 3: P2 is selected as $Need_2 \leq Work$ - 0.5 marks

Work = (4,9,8,7) + (4,0,2,0)

= (8,9,10,7) — 1mark

Step 4 : P4 is selected as $Need_4 \leq Work$ – 0.5 marks

$Work = (8, 9, 10, 7) + (2, 4, 3, 0)$

$= (10, 13, 13, 7)$ – 1 mark

Step 5 : P1 is selected as $Need_1 \leq Work$ – 0.5 marks

$Work = (10, 13, 13, 7) + (3, 5, 0, 1)$

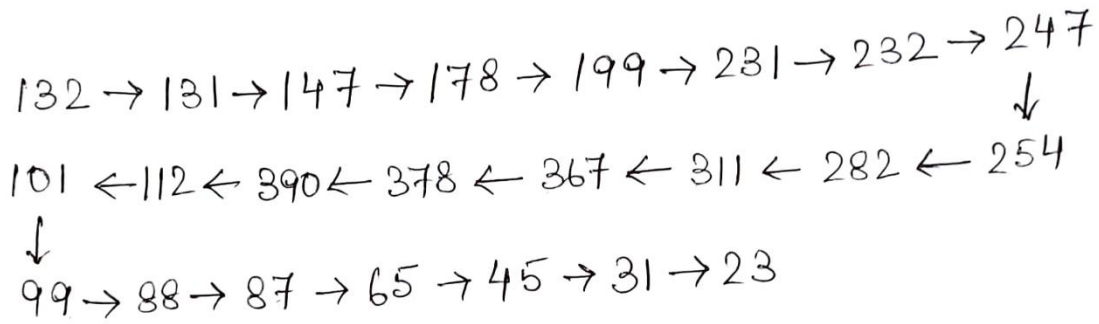
$= (13, 18, 13, 8)$ – 1 mark

Safe sequence is $\langle P3, P5, P2, P4, P1 \rangle$ – 0.5 marks

Note : Instead of Work vector, Available can also be written. If any Work vector value is partially correct, 0.5 marks to be given. No calculation is required for Work vectors. Marks will be deducted if any condition is missed, any vector contents are not shown.

3. Consider a disk queue with I/O requests to blocks on the following cylinders: 178, 232, 87, 65, 199, 45, 247, 112, 282, 31, 367, 101, 88, 311, 254, 147, 231, 378, 390, 23, 131, 99. Currently the disk head is at cylinder 132. The system uses **SSTF** disk scheduling algorithm. Assume that the range of cylinder numbers on the disk is from 0 to 399. Calculate the total head movement for servicing all the I/O requests. Clearly show in detail the sequence in which the I/O requests are serviced. No need to show any calculation for determining the total head movement. **[13 marks]**

Ans.



Total head movement = 627

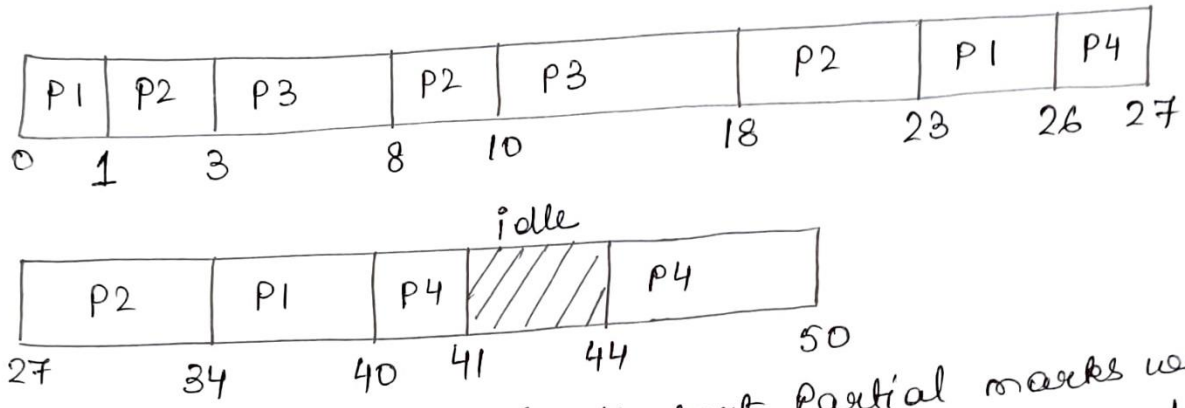
0.5 for each head transition.
2 marks for total head mvmt.
partially correct ans.

Note – 0.5 marks for each head movement starting from the first I/O request. 11 marks for the entire sequence. 2 marks for the total head movement. No calculation is required for total head movement.

4. Consider the processes shown in the following table along with their priorities, arrival times and CPU burst cycles-I/O burst cycles-CPU burst cycles (last column of the table, the CPU burst cycles are underlined) in milliseconds. The processes are scheduled using the **preemptive priority scheduling** algorithm. Lower integer values indicate higher priorities with priority 0 being the highest priority. Assume that a process does not have to wait for I/O device access after its CPU burst cycle is over. Show a Gantt chart depicting the order of process execution**.** Calculate the waiting time and turnaround time of each process, and the average waiting time and the average turnaround time. You are not required to show calculations for waiting time, turnaround time, average waiting time and average turnaround time. Marks will not be awarded if final answers are incorrect. **[14 marks]**

Process	Priority	Arrival Time	CPU Burst Cycle-I/O Burst Cycle-CPU Burst Cycle (milliseconds)
P1	2	0	4 – 3 – 6
P2	1	1	9 – 4 – 7
P3	0	3	5 – 2 – 8
P4	3	6	2 – 3 – 6

Ans.



Note – 4 marks for Gantt chart. Partial marks will be given if Gantt chart is partially correct. No marks will be given if process execution order is incorrect.

Waiting Time

P1 → 27
P2 → 13
P3 → 0
P4 → 33

1 mark for each WT, no partial marking, no calculations required

Turnaround Time

P1 → 40
P2 → 33
P3 → 15
P4 → 44

1 mark for each TAT, no partial marking, no calculations required.

Average Waiting Time = 18.25 — 1 mark

Average Turnaround Time = 33 — 1 mark

no partial marking, no calc. reqd.

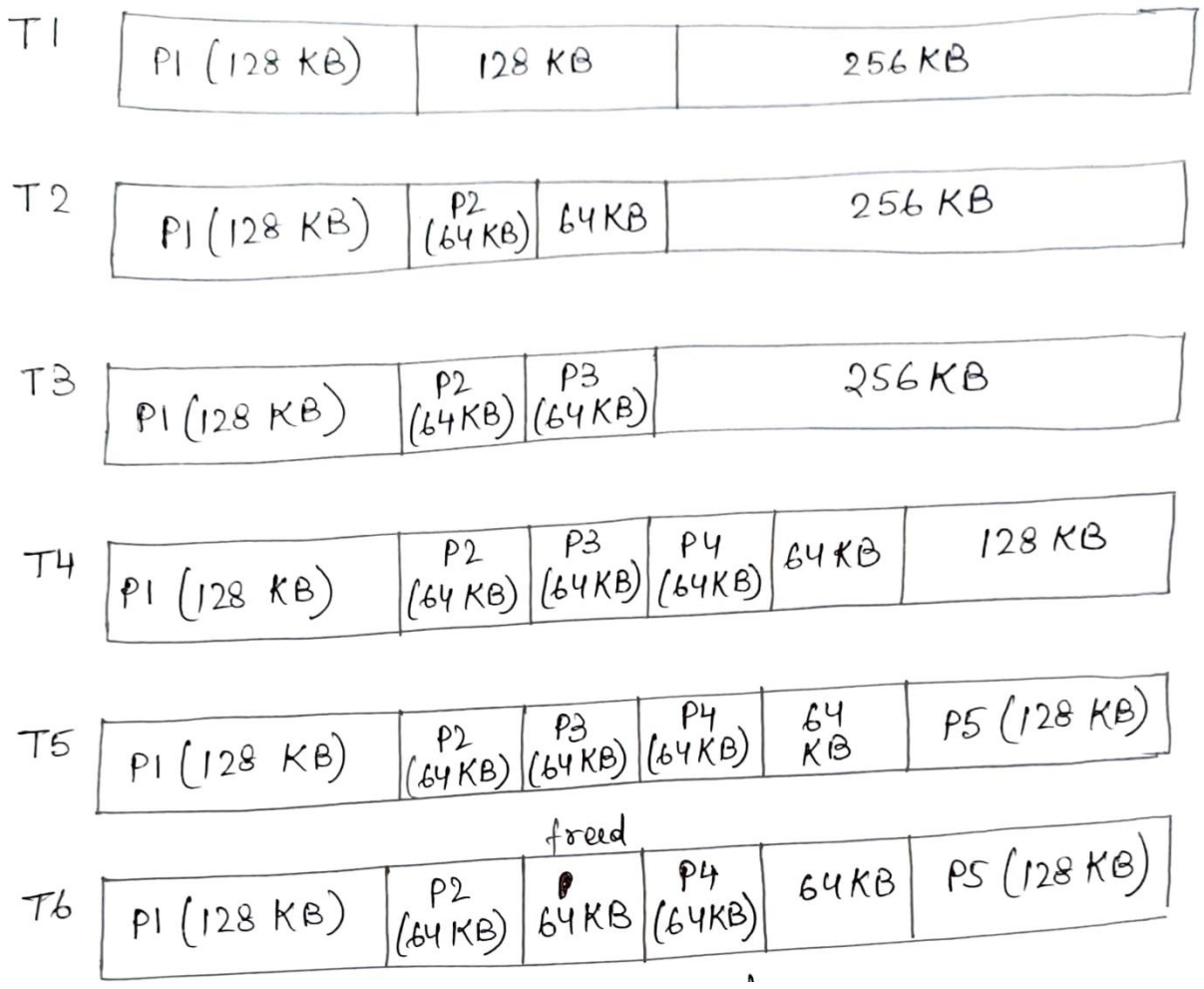
Note – Unit not required for any time.

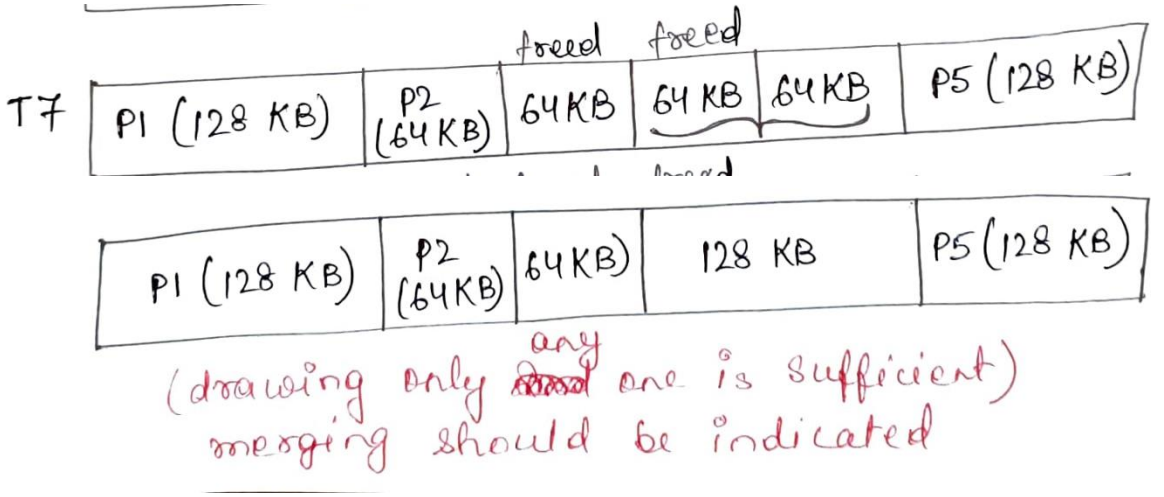
5. A system uses the buddy system for allocating kernel memory to processes. Initially, a 512 KB block of memory is available and is free. The following table shows the amount of kernel memory requested and freed by different processes at various time instants.

Time Instant	**Memory requested or freed**
T1	Process P1 requests a block of size 112 KB
T2	Process P2 requests a block of size 37 KB
T3	Process P3 requests a block of size 52 KB
T4	Process P4 requests a block of size 60 KB
T5	Process P5 requests a block of size 121 KB
T6	Process P3 frees its allocated block
T7	Process P4 frees its allocated block
T8	Process P2 frees its allocated block

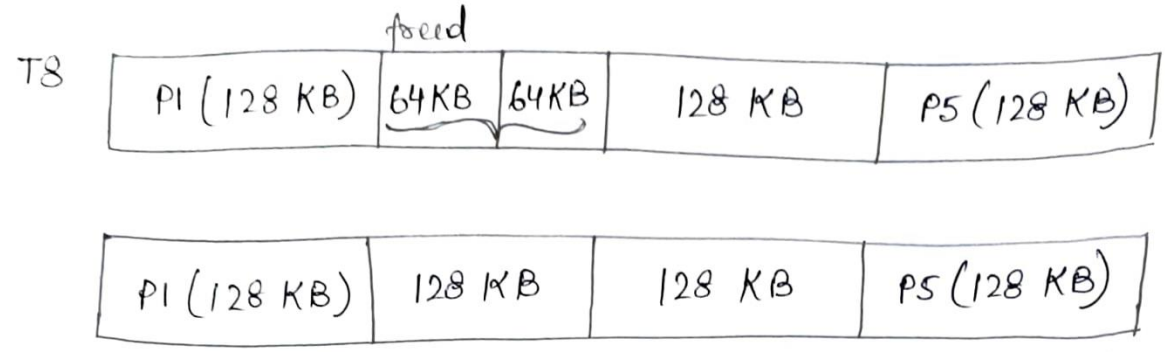
Determine the state of kernel memory at each time instant. You need to clearly indicate the free/available/unallocated blocks along with their sizes at each time instant. In case of kernel memory being allocated to a process, clearly show the block and mention the amount of memory allocated to the process. You can show the memory state using a diagram. The diagram should be legible. Other than the processes mentioned in the table, no other process frees its allocated block. **[8 marks]**

Ans.





(drawing only ~~one~~ any one is sufficient)
merging should be indicated



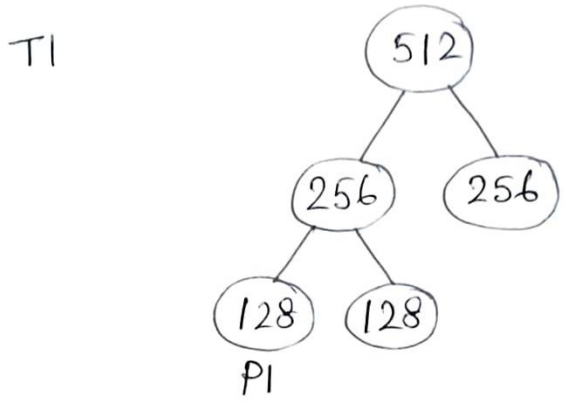
(drawing either one is sufficient, but merging should be indicated)

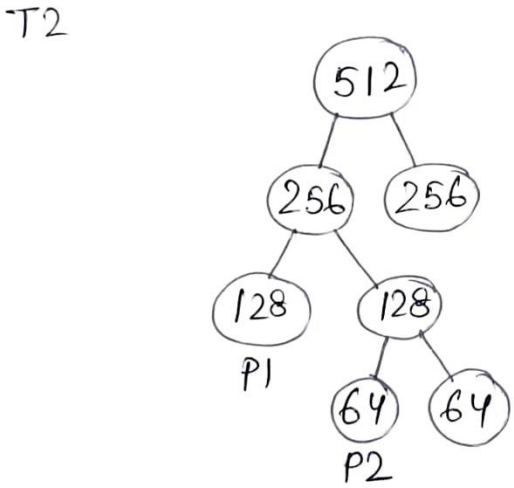
At T8, only one merging occurs. If any further merging is shown, 1 mark will be deducted

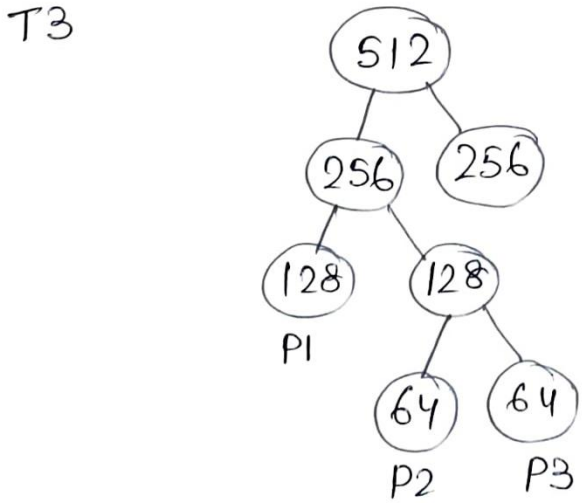
Note – 1 mark for each time instant. If only one diagram is given, only 3 marks will be awarded. If merging is not indicated, 0.5 marks will be given for that time instant. If free and allocated blocks are not shown properly, only 4 marks will be awarded.

Note – 1 mark for each time instant. If only one single diagram is given, only 4 marks will be awarded provided it is correct. If the two merging are not indicated, 0.5 marks each will be awarded for those time instants. If free and allocated blocks are not shown properly, only 4 marks will be awarded. If block size is not indicated, only 4 marks will be awarded.

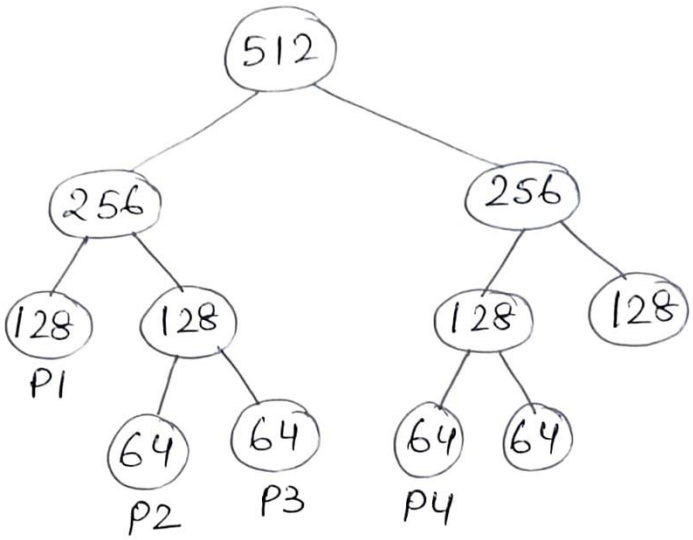
Alternate Diagram:



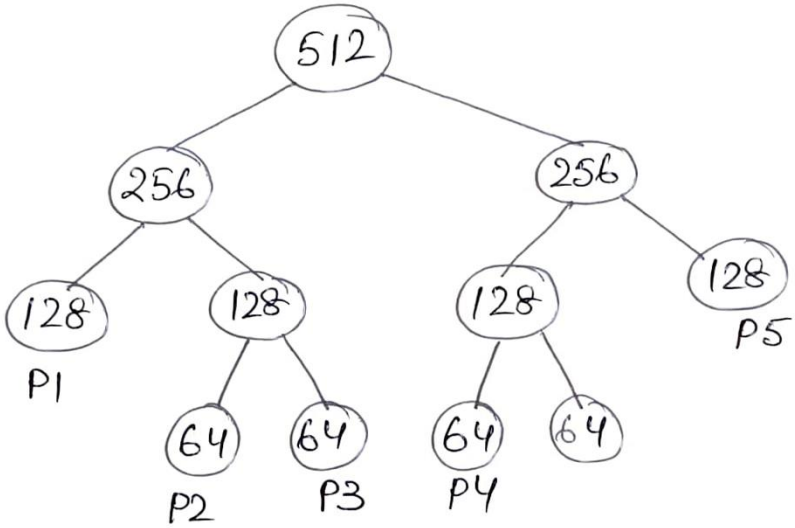




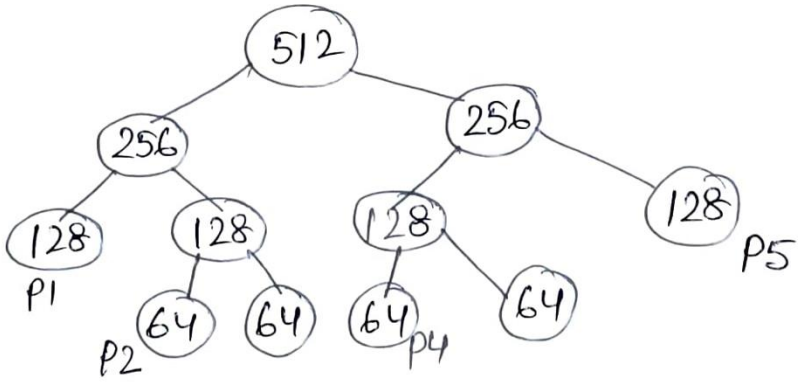
T4


512
256
256
128
128
128
128
P1
64
64
64
64
P2
P3
P4

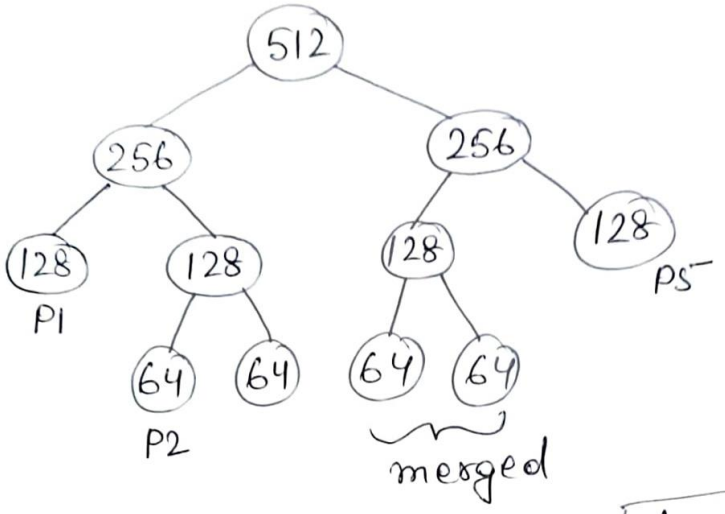
T5

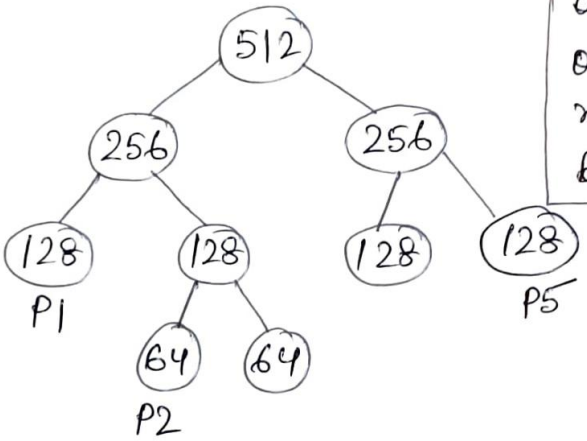

512
256
256
128
128
128
128
P1
P5
64
64
64
64
P2
P3
P4

T6


512
256
256
128
128
128
128
P1
P5
64
64
64
64
P2
P4

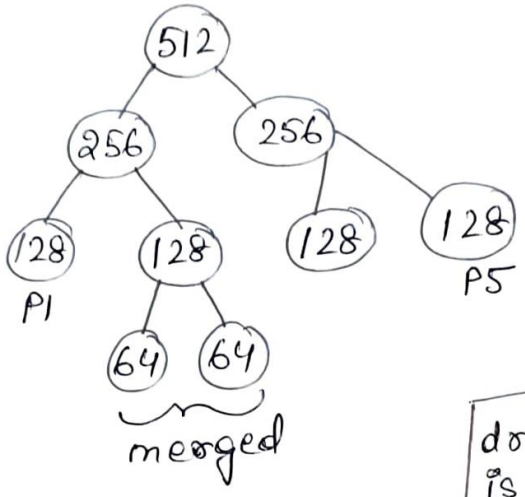
T7



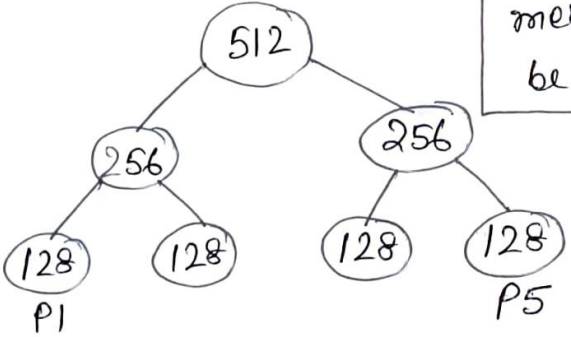


drawing only one is sufficient, merging should be indicated

T8



drawing only one is sufficient, merging should be shown



6. Consider a system that uses variable-partition memory management scheme. Initially, the main memory contains the following free partitions/holes – Partition A of size 210 KB, Partition B of size 180 KB, Partition C of size 150 KB, Partition D of size 145 KB, Partition E of size 340 KB and Partition F of size 250 KB. The partitions present in memory are depicted in the figure below.

A	B	C	D	E	F

The following processes arrive in memory in order – P1 of size 140 KB, P2 of size 90 KB, P3 of size 310 KB and P4 of size 190 KB. The system uses the **worst-fit strategy** for memory allocation for the arriving processes. Determine the partition that is allocated to each process if at all one can be allocated. In case the memory requirement of a process cannot be satisfied, you need to mention that as well. Assume that on this system, when a partition/hole is created due to splitting, it is treated as an independent partition/hole and adjacent free partitions cannot be combined together. Also, remember that none of the processes finish execution and leave the system. Pay attention to the order in which the processes arrive while answering this question. **[4 marks]**

Ans.

6. P1 → E (340 KB)
P2 → F (250 KB)
P3 → can't allocate a partition
P4 → A (210 KB)

Note - 1 mark for each process. No partial marking for any process.

****