Ans: no. 1

(a) 25 bits in virtual address.

(b) physical address = $\log_2(512 \times 2^{20})$ = 29 bits

(c) offset = $\log_2(128)$ = 7 bits

(d) PFN size = 29 − 7 = 22 bits

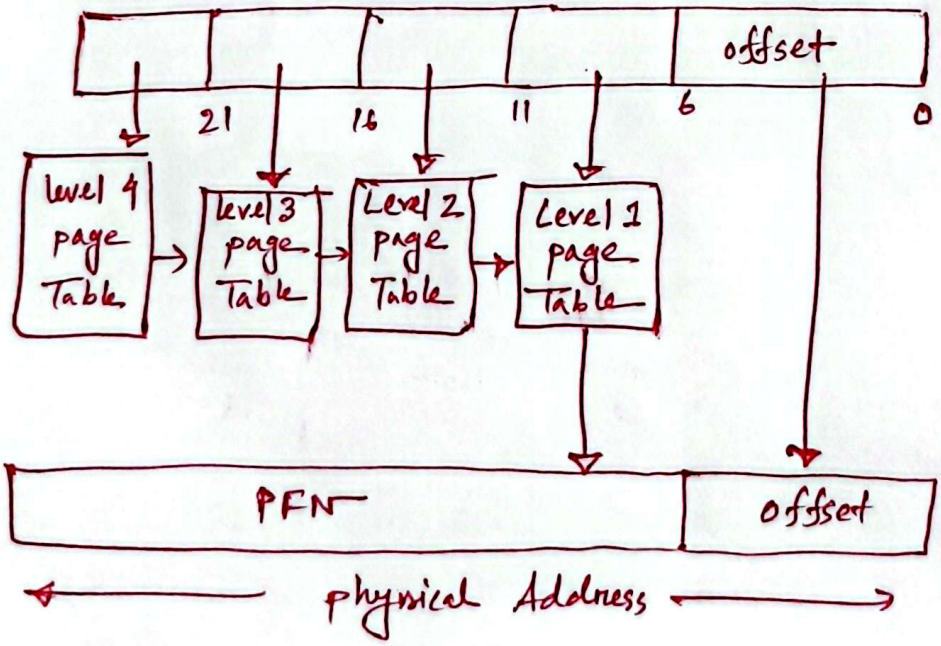
Flag size = 7 bits

∴ PTE size = 22 + 7 = 29 bits ≃ 4 bytes

(e) Number of bits required at each level of page table

$= \log_2\left(\frac{\text{page size}}{\text{PTE size}}\right) = \log_2\left(\frac{128}{4}\right) = 5$ bits

VPN size = 25 − 7 = 18 bits



(f) Minimum memory for a single process

= Process's minimum memory + page table's minimum memory

= 1 page + 4 pages

= 5 pages

= (5×128) Bytes [∵ 1 page = 128 Bytes]

∴ Maximum number of processes that can run concurrently = $\frac{\text{RAM size}}{\text{Minimum Memory for one process}}$

= $\left\lfloor \frac{512 \times 2^{20}}{5 \times 128} \right\rfloor$

= [illegible]

= 8,38,860 Ans.

(3)

Page ref	Hit/Fault	Evicted	Memory state
1	Fault	x	1
2	"	x	1, 2
3	"	x	1, 2, 3
4	"	x	1, 2, 3, 4
1	Hit	x	1, 2, 3, 4
2	Hit	x	1, 2, 3, 4
5	Fault	3	1, 2, 5, 4
1	Hit	x	1, 2, 5, 4
2	Hit	x	1, 2, 5, 4
3	Fault	4	1, 2, 5, 3
4	Fault	5	1, 2, 4, 3
5	Fault	1	5, 2, 4, 3

∴ Number of page faults = 8

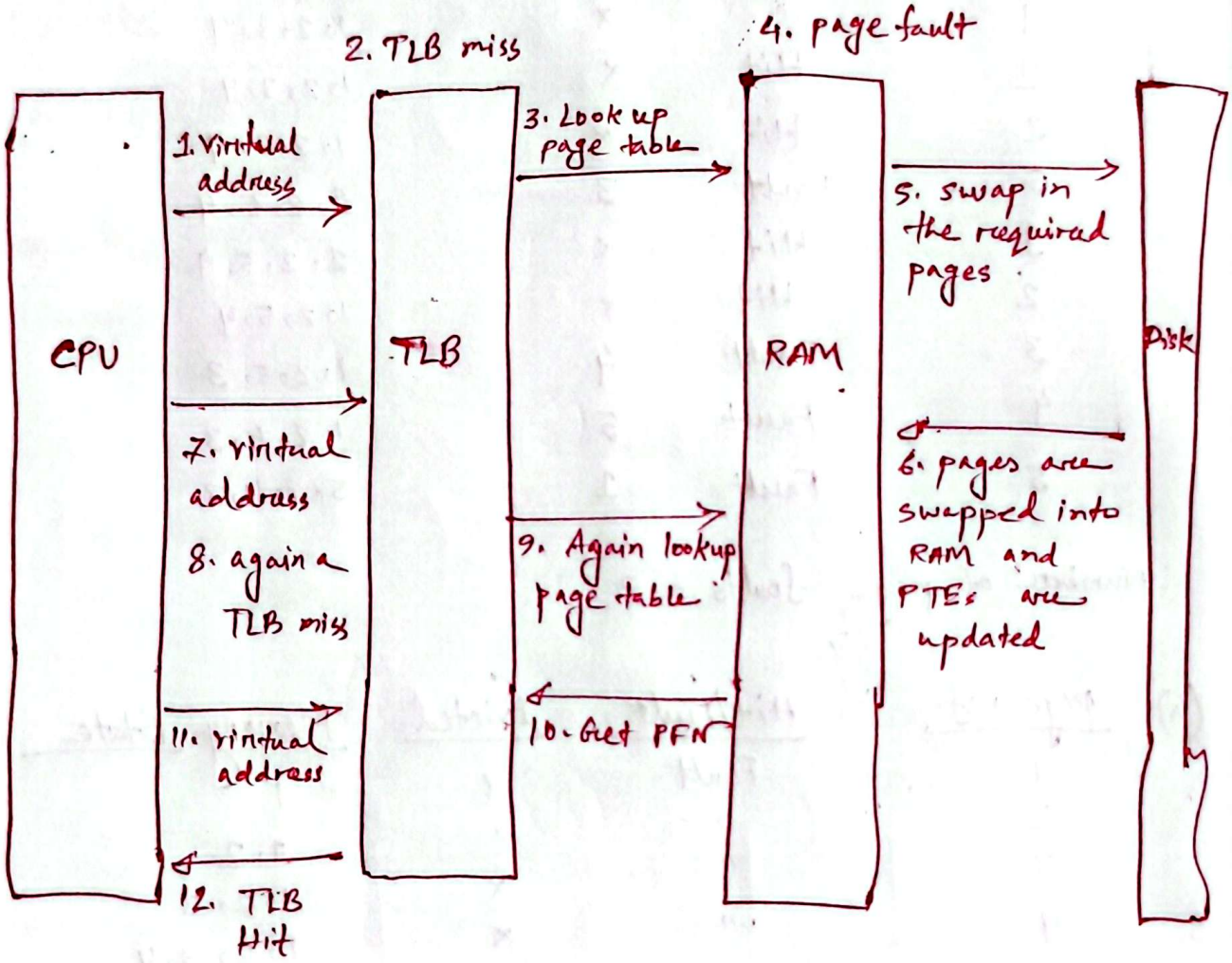
(4)

Page ref	Hit/Fault	Evicted	Memory State
1	Fault	x	1
2	"	x	1, 2
3	"	x	1, 2, 3
4	"	x	1, 2, 3, 4
1	Hit	x	1, 2, 3, 4
2	Hit	x	1, 2, 3, 4
5	Fault	2	1, 5, 3, 4
1	Hit	x	1, 5, 3, 4
2	Fault	1	2, 5, 3, 4
3	Hit	x	2, 5, 3, 4
4	Hit	x	2, 5, 3, 4
5	Hit	x	2, 5, 3, 4

∴ Number of page faults = 6

(i) Reversing the recency logic increased Hits or decreased page faults for this example but in general LRU is better policy than MRU.

(j)



Ans: no. 2

(a)

Strategy	Deadlock Condition
1	Hold and wait
2	No preemption
3	Circular wait

(b) Most appropriate strategy: Strategy – 3 (prevent circular wait)

Reason: Setting priorities to transactions, is free from starvation and roll-back overhead.

(c)

Process	Need				Available				Safe Sequence
	A	B	C	D	A	B	C	D	
T_0	2	1	0	3	0	3	0	0	T_2
✓ T_1	1	0	0	1	3	5	2	1	T_1
✓ T_2	0	1	0	0	5	7	3	1	
T_3	4	1	0	2					
T_4	2	1	1	3					

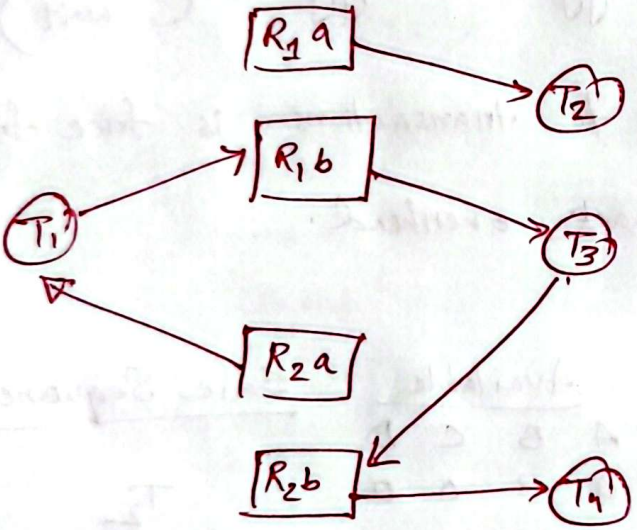
Minimum number of additional available resources

$= \{0, 0, 0, 2\}$

(d) I do not agree with this approach because the transformation to single instance RAG is incorrect.

T_1 has requested one instance of R1 but the transformation shows two instance.

One possible correct transformation:-



If we can find one possible single instance RAG with no cycle, then it's deadlock free.

Ans: no. 3

(a) Dangling Pointer: A pointer that points to a memory location that has been de-allocated or freed.

Solutions: 1) Back pointers
2) Daisy chain with Back-pointers
3) Entry-Hold Count.

Discussing any one solution is sufficient.

(b) Advantage of Contiguous FA:

1. Easy to implement.
2. Fast Random Access

Disadvantage

1. External Fragmentation 2) Low utilization of Disk

(c) Indexed File Allocation Advantage:

1. No External Fragmentation
2. Fast Random Access

Disadvantage:

1. Additional Space for index blocks
2. Multi-level indexing for large files.

(d) Jenny's idea is more appropriate.

Factor	Tom's idea	Jenny's idea
1. Performance	Finding contiguous free blocks for large files reduce performance	Easy to find contiguous free blocks for small files.
2. Space utilization	Lower utilization as we may not find large contiguous free blocks	Better utilization so it is more likely to find contiguous free blocks for small files.
3. Fragmentation	External fragmentation is high.	External fragmentation is low
4. Growth flexibility	Less flexible for large files.	More flexible for small files.

United International University (UIU)
Department of Computer Science and Engineering
CSE 4509: OPERATING SYSTEMS, Final Exam, Summer 2025
Total Marks: 40, Duration: **2 hours**

Any examinee found adopting unfair means will be expelled from the trimester/program as per UIU disciplinary rules.

Answer all the questions. Marks are indicated on the right.

1. Consider a system utilizing a **multi-level page table** scheme where **multiple processes can run concurrently**. Here any running process can generate at most 2^{25} different virtual addresses. The system is equipped with 512 **MB of RAM** (1 MB = 2^{20} bytes). A **single frame** can access 128 **different addresses**, and Page Table Entries (PTEs) must store **metadata** (read, write, execution, dirty, valid, present, referenced). The size of a PTE is **rounded up** to the nearest multiple of bytes. Determine the following values:

   a. How many bits are there in the virtual address? [1]
   b. How many bits are there in the physical address? [1]
   c. How many bits are there for the offset? [1]
   d. What is the size of a page table entry (PTE) in bytes? [1]
   e. Design a multi level page table block diagram showing necessary calculations. [3]
   f. What is the maximum number of processes that can run concurrently? [3]

   Consider the following context for questions g to j
   A system uses the LRU page replacement policy with 4 page frames. The page reference string is: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5
   Assume all frames are initially empty.

   g. Simulate the execution and show the page frame contents after each reference and count the number of page faults. [3]
   h. Now assume a programmer mistakenly implements LRU by replacing the most recently used page instead of the least recently used page (MRU instead of LRU). Recalculate the number of page faults for the same reference string. [2]
   i. Compare the results and explain why reversing the recency logic can sometimes increase or decrease page faults depending on access patterns. [2]
   j. With the help of block diagrams, show what happen when there is a page fault in the system. [3]

2. A database management system allows multiple transactions to access and update shared resources such as tables and data blocks. Each transaction must lock resources before updating them and release the locks after committing. Occasionally, transactions hold one lock while waiting for another, causing the system to freeze due to deadlocks.
To prevent this issue, the database team considers several strategies:

1. Forcing each transaction to request all required resources at once before execution.
2. Allowing preemption — i.e., if a transaction can't obtain a resource, it must release all held resources and try again later.
3. Assigning unique numeric priorities to transactions so that resources are always requested in increasing priority order.

a. Which of the four deadlock conditions (**mutual exclusion, hold and wait, no preemption, circular wait**) each strategy targets? [2]

b. Which of the above strategies is the most appropriate deadlock prevention technique for this scenario, and why? [2]

Consider the Table-1 snapshot of a system for question c:

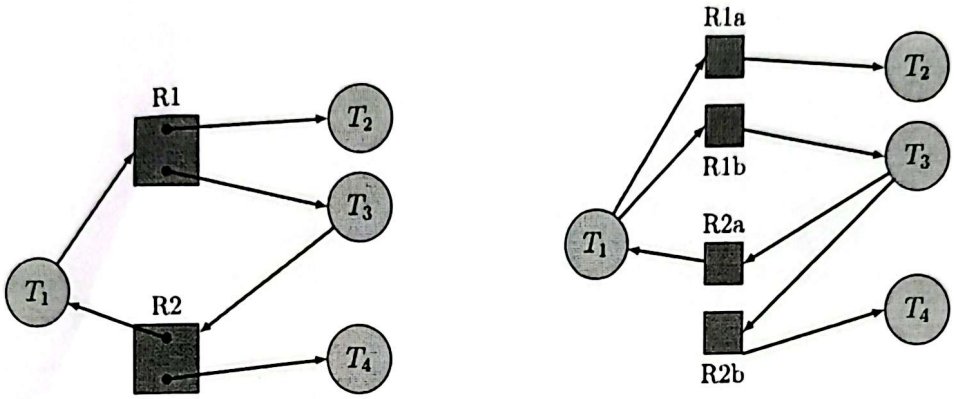
Process	Allocation				Max			
	A	B	C	D	A	B	C	D
T0	3	0	1	4	5	1	1	7
T1	2	2	1	0	3	2	1	1
T2	3	2	2	1	3	3	2	1
T3	0	5	1	0	4	6	1	2
T4	4	2	1	2	6	3	2	5

c. Determine whether the system is in a safe state or not if the available resources are {0,3,0,0}? If the state is safe, illustrate the order in which the threads may be completed. Otherwise,Calculate the minimum number of additional available resources required to ensure the execution of all threads. [4]

Consider the following context for question d

Professor X's analysis of a multi-instance RAG (Figure 1: Left) showed that a cycle is non-fatal; the system is safe because a sequence exists (e.g., T_4 releases R_2, allowing T_3 to proceed).

A student proposes transforming this RAG into a single-instance RAG (Figure 2: Right) by replacing a request $\mathbf{T} \rightarrow \mathbf{R}_j$ with concurrent requests $\mathbf{T} \rightarrow \mathbf{R}_{j,a}$ and $\mathbf{T} \rightarrow \mathbf{R}_{j,b}$ to every resource instance. The student concludes that a cycle in the resulting single-instance graph must imply deadlock.



d. Do you agree with this approach? Explain your justification. [2]

3. Tom and Jerry have learned about contiguous and indexed file allocation methods. They now want to design a hybrid allocation system that combines both.

- Tom's idea: Store large files in contiguous space and use indexing for small files.
- Jerry's idea: Store small files in contiguous space and use indexing for large files.

a. What is a dangling pointer in the context of acyclic graph-based or tree directory structures? Discuss how the issue of dangling pointers can be resolved. [2]

b. What are the advantages and disadvantages of **Contiguous file allocation**? [2]

c. What are the advantages and disadvantages of **Indexed file allocation**? [2]

d. Which design is more appropriate? You can consider the following factors: **performance, space utilization, fragmentation, and file growth flexibility**. [4]