07. Technical Module Solved PYQs & Comprehensive Practice Bank

> **Module Focus**: 30 Questions in 30 Minutes
> **Topics Covered**: Data Structures, DAA, Operating Systems, Basic Embedded Systems, Computer Networks, SDLC/Agile, Software QA & Testing Tools
> **Coverage**: 24 Fully Solved PYQs with Step-by-Step Derivations, Gantt Charts, State Traces, and Explicit Formulas.

---

1. Operating Systems Solved PYQs

PYQ 1: Preemptive SRTF CPU Scheduling Gantt Chart

Problem: Consider 4 processes arriving at times specified below with their respective CPU burst times:

Process	Arrival Time (AT)	Burst Time (BT)
P_1	0 ms	7 ms
P_2	2 ms	4 ms
P_3	4 ms	1 ms
P_4	5 ms	4 ms

Calculate the **Average Waiting Time** using Shortest Remaining Time First (SRTF) scheduling.

Detailed Step-by-Step Derivation: - **At $t=0$**: Only P_1 has arrived ($BT=7$). Executes until $t=2$. (Remaining $BT(P_1) = 5$). - **At $t=2$**: P_2 arrives ($BT=4$). Compare $BT(P_2)=4$ with remaining $BT(P_1)=5$. P_2 is shorter! Preempt P_1. Execute P_2 from $t=2 \to 4$. (Remaining $BT(P_2) = 2$). - **At $t=4$**: P_3 arrives ($BT=1$). Compare $BT(P_3)=1$, $BT(P_2)=2$, $BT(P_1)=5$. P_3 is shortest! Preempt P_2. Execute P_3 from $t=4 \to 5$. (P_3 finishes at $t=5$). - **At $t=5$**: P_4 arrives ($BT=4$). Ready queue contains: $P_2 (BT=2)$, $P_4 (BT=4)$, $P_1 (BT=5)$. - Execute P_2 from $t=5 \to 7$. (P_2 finishes at $t=7$). - Execute P_4 from $t=7 \to 11$. (P_4 finishes at $t=11$). - Execute P_1 from $t=11 \to 16$. (P_1 finishes at $t=16$).

Gantt Chart Visualization:

```
+-------+-------+---+-------+---------+------------+
|  P1   |  P2   |P3 |  P2   |   P4    |     P1     |
+-------+-------+---+-------+---------+------------+
0       2       4   5       7         11           16
```

Execution Summary Table:

Process	Arrival (AT)	Finish (FT)	Turnaround Time ($TAT = FT - AT$)	Waiting Time ($WT = TAT - BT$)
P_1	0	16	$16 - 0 = 16$	$16 - 7 = 9$ ms
P_2	2	7	$7 - 2 = 5$	$5 - 4 = 1$ ms
P_3	4	5	$5 - 4 = 1$	$1 - 1 = 0$ ms
P_4	5	11	$11 - 5 = 6$	$6 - 4 = 2$ ms

$$\text{Average Waiting Time} = \frac{9 + 1 + 0 + 2}{4} = \frac{12}{4} = 3.0 \text{ ms}$$

Correct Answer: `3.0 ms`

PYQ 2: Round Robin CPU Scheduling (Time Quantum = 2 ms)

Problem: Given 4 processes with arrival times and burst times as below. Find the **Average Turnaround Time** and **Average Waiting Time** using Round Robin scheduling with **Time Quantum (TQ) = 2 ms**.

Process	Arrival Time (AT)	Burst Time (BT)
P_1	0 ms	5 ms
P_2	1 ms	3 ms
P_3	2 ms	1 ms
P_4	3 ms	4 ms

Step-by-Step Execution Trace: - **t=0**: P_1 arrives. Ready Queue: `[P1]`. P_1 executes $t=0 \to 2$ (Rem $BT=3$). - During $t=0 \to 2$, P_2 (t=1) and P_3 (t=2) arrive. Ready Queue: `[P2, P3, P1]`. - **t=2**: P_2 executes $t=2 \to 4$ (Rem $BT=1$). - During $t=2 \to 4$, P_4 (t=3) arrives. Ready Queue: `[P3, P1, P4, P2]`. - **t=4**: P_3 executes $t=4 \to 5$ (Finishes at $t=5$, $BT=1 < TQ$). Ready Queue: `[P1, P4, P2]`. - **t=5**: P_1 executes $t=5 \to 7$ (Rem $BT=1$). Ready Queue: `[P4, P2, P1]`. - **t=7**: P_4 executes $t=7 \to 9$ (Rem $BT=2$). Ready Queue: `[P2, P1, P4]`. - **t=9**: P_2 executes $t=9 \to 10$ (Finishes at $t=10$, Rem $BT=1$). Ready Queue: `[P1, P4]`. - **t=10**: P_1 executes $t=10 \to 11$ (Finishes at $t=11$, Rem $BT=1$). Ready Queue: `[P4]`. - **t=11**: P_4 executes $t=11 \to 13$ (Finishes at $t=13$). Ready Queue: `[]`.

Gantt Chart Visualization:

```
+-----+-----+---+-----+-----+---+---+-------+
| P1  | P2  |P3 | P1  | P4  |P2 |P1 |   P4  |
+-----+-----+---+-----+-----+---+---+-------+
0     2     4   5     7     9  10  11      13
```

Execution Summary Table:

Process	AT	BT	Finish Time (FT)	Turnaround Time ($TAT = FT - AT$)	Waiting Time ($WT = TAT - BT$)
P_1	0	5	11	$11 - 0 = 11$ ms	$11 - 5 = 6$ ms
P_2	1	3	10	$10 - 1 = 9$ ms	$9 - 3 = 6$ ms
P_3	2	1	5	$5 - 2 = 3$ ms	$3 - 1 = 2$ ms
P_4	3	4	13	$13 - 3 = 10$ ms	$10 - 4 = 6$ ms

$$\text{Average Turnaround Time} = \frac{11 + 9 + 3 + 10}{4} = \frac{33}{4} = 8.25 \text{ ms}$$ $$\text{Average Waiting Time} = \frac{6 + 6 + 2 + 6}{4} = \frac{20}{4} = 5.0 \text{ ms}$$

Correct Answer: Average TAT = `8.25 ms`, Average WT = `5.0 ms`

PYQ 3: Non-Preemptive Priority CPU Scheduling

Problem: Given 4 processes with priority values (lower number = higher priority).

Process	Arrival Time (AT)	Burst Time (BT)	Priority
P_1	0 ms	4 ms	3
P_2	1 ms	2 ms	1 (Highest)
P_3	2 ms	5 ms	4 (Lowest)
P_4	3 ms	1 ms	2

Find the **Average Waiting Time** under Non-Preemptive Priority Scheduling.

Step-by-Step Execution Trace: - **t=0**: Only P_1 is present. Non-preemptive, so P_1 runs to completion from $t=0 \to 4$. - **t=4**: P_1 finishes. Processes present in ready queue: P_2 (prio 1), P_4 (prio 2), P_3 (prio 4). - P_2 has highest priority (1). Runs $t=4 \to 6$. - **t=6**: P_2 finishes. Highest priority remaining is P_4 (prio 2). Runs $t=6 \to 7$. - **t=7**: P_4 finishes. Remaining process is P_3 (prio 4). Runs $t=7 \to 12$.

Gantt Chart Visualization:

```
+---------+-----+---+-----------+
|   P1    | P2  |P4 |    P3     |
+---------+-----+---+-----------+
0         4     6   7           12
```

Execution Table:

Process	AT	BT	Priority	FT	$TAT = FT - AT$	$WT = TAT - BT$
P_1	0	4	3	4	$4 - 0 = 4$	$4 - 4 = 0$ ms
P_2	1	2	1	6	$6 - 1 = 5$	$5 - 2 = 3$ ms
P_3	2	5	4	12	$12 - 2 = 10$	$10 - 5 = 5$ ms
P_4	3	1	2	7	$7 - 3 = 4$	$4 - 1 = 3$ ms

$$\text{Average Waiting Time} = \frac{0 + 3 + 5 + 3}{4} = \frac{11}{4} = 2.75 \text{ ms}$$

Correct Answer: `2.75 ms`

PYQ 4: Page Replacement Algorithm Comparison (LRU vs FIFO)

Problem: A virtual memory system has **3 page frames.** Given the page reference string:
`7, 0, 1, 2, 0, 3, 0, 4, 2, 3`
Calculate total page faults for both **FIFO** and **LRU** algorithms.

Step-by-Step Execution Traces:

1. FIFO (First-In, First-Out):

Ref	Frame 1	Frame 2	Frame 3	Status
7	7	-	-	Fault (1)
0	7	0	-	Fault (2)
1	7	0	1	Fault (3)
2	2	0	1	Fault (4) (Evict 7)
0	2	0	1	HIT
3	2	3	1	Fault (5) (Evict 0)
0	2	3	0	Fault (6) (Evict 1)
4	4	3	0	Fault (7) (Evict 2)
2	4	2	0	Fault (8) (Evict 3)
3	4	2	3	Fault (9) (Evict 0)

Total FIFO Page Faults: **9**

2. LRU (Least Recently Used):

Ref	Frame 1	Frame 2	Frame 3	Status
7	7	-	-	Fault (1)
0	7	0	-	Fault (2)
1	7	0	1	Fault (3)
2	2	0	1	Fault (4) (Evict 7)
0	2	0	1	HIT
3	2	0	3	Fault (5) (Evict 1)
0	2	0	3	HIT
4	4	0	3	Fault (6) (Evict 2)
2	4	0	2	Fault (7) (Evict 3)
3	4	3	2	Fault (8) (Evict 0)

Total LRU Page Faults: **8**

Correct Answer: FIFO Page Faults = `9`, LRU Page Faults = `8`

PYQ 5: Banker's Algorithm for Deadlock Avoidance

Problem: Consider a system with 5 processes (P_0 to P_4) and 3 resource types ($A=10, B=5, C=7$). Given the state below:

Process	Allocation (A B C)	Max (A B C)
P_0	0 1 0	7 5 3
P_1	2 0 0	3 2 2
P_2	3 0 2	9 0 2
P_3	2 1 1	2 2 2
P_4	0 0 2	4 3 3

Currently available resources: $\text{Available} = [3, 3, 2]$.
1. Compute the **Need Matrix**.
2. Determine if the system is in a **Safe State** and find the **Safety Sequence**.

Step-by-Step Solution:

1. Calculate Need Matrix ($\text{Need} = \text{Max} - \text{Allocation}$):

- $\text{Need}(P_0) = [7-0, 5-1, 3-0] = [7, 4, 3]$
- $\text{Need}(P_1) = [3-2, 2-0, 2-0] = [1, 2, 2]$
- $\text{Need}(P_2) = [9-3, 0-0, 2-2] = [6, 0, 0]$
- $\text{Need}(P_3) = [2-2, 2-1, 2-1] = [0, 1, 1]$
- $\text{Need}(P_4) = [4-0, 3-0, 3-2] = [4, 3, 1]$

2. Safety Algorithm Trace ($\text{Work} = [3, 3, 2]$):

- **Step 1**: Check P_1: $\text{Need}(P_1) = [1, 2, 2] \le \text{Work} [3, 3, 2]$ $\implies$ **TRUE**.
- P_1 completes. New $\text{Work} = \text{Work} + \text{Allocation}(P_1) = [3, 3, 2] + [2, 0, 0] = [5, 3, 2]$.
- **Step 2**: Check P_3: $\text{Need}(P_3) = [0, 1, 1] \le \text{Work} [5, 3, 2]$ $\implies$ **TRUE**.
- P_3 completes. New $\text{Work} = [5, 3, 2] + [2, 1, 1] = [7, 4, 3]$.
- **Step 3**: Check P_4: $\text{Need}(P_4) = [4, 3, 1] \le \text{Work} [7, 4, 3]$ $\implies$ **TRUE**.
- P_4 completes. New $\text{Work} = [7, 4, 3] + [0, 0, 2] = [7, 4, 5]$.
- **Step 4**: Check P_0: $\text{Need}(P_0) = [7, 4, 3] \le \text{Work} [7, 4, 5]$ $\implies$ **TRUE**.
- P_0 completes. New $\text{Work} = [7, 4, 5] + [0, 1, 0] = [7, 5, 5]$.
- **Step 5**: Check P_2: $\text{Need}(P_2) = [6, 0, 0] \le \text{Work} [7, 5, 5]$ $\implies$ **TRUE**.
- P_2 completes. New $\text{Work} = [7, 5, 5] + [3, 0, 2] = [10, 5, 7]$.

All processes finished successfully!

Correct Answer: System is in **Safe State**. Valid Safety Sequence: `<P1, P3, P4, P0, P2>`

PYQ 6: Disk Scheduling Algorithms (SSTF vs SCAN)

Problem: Disk head initially at cylinder **53**. Request queue: `98, 183, 37, 122, 14, 124, 65, 67`. Total cylinders = 200 (0-199).
Calculate total head movement using **Shortest Seek Time First (SSTF)**.

Step-by-Step SSTF Trace: 1. Current Head = 53. Distances to queue: $|53-65|=12$, $|53-37|=16$, $|53-67|=14$. - **Next Head: 65** (Seek distance $= 65 - 53 = 12$). 2. Current Head = 65. Nearest: **67** (Seek distance $= 67 - 65 = 2$). 3. Current Head = 67. Nearest: **37** (Seek distance $= 67 - 37 = 30$). 4. Current Head = 37. Nearest: **14** (Seek distance $= 37 - 14 = 23$). 5. Current Head = 14. Nearest: **98** (Seek distance $= 98 - 14 = 84$). 6. Current Head = 98. Nearest: **122** (Seek distance $= 122 - 98 = 24$). 7. Current Head = 122. Nearest: **124** (Seek distance $= 124 - 122 = 2$). 8. Current Head = 124. Nearest: **183** (Seek distance $= 183 - 124 = 59$).

$$\text{Total Head Movement} = 12 + 2 + 30 + 23 + 84 + 24 + 2 + 59 = 236 \text{ cylinders}$$

Correct Answer: `236 cylinders`

---

2. Data Structures & Algorithms (DS / DAA) Solved PYQs

PYQ 7: Binary Tree Reconstruction & Postorder Traversal

Problem: Construct a binary tree from the given traversals: - **Preorder**: `A, B, D, E, C, F` - **Inorder**: `D, B, E, A, F, C`
Determine the **Postorder Traversal** of the reconstructed tree.

Step-by-Step Tree Construction: 1. Root = First element of Preorder = `A`. 2. Locate `A` in Inorder: `[D, B, E]` is Left Subtree, `[F, C]` is Right Subtree. 3. Left Subtree Preorder: `[B, D, E]`. Root is `B`. - In Inorder `[D, B, E]`: `D` is left child of `B`, `E` is right child of `B`. 4. Right Subtree Preorder: `[C, F]`. Root is `C`. - In Inorder `[F, C]`: `F` is left child of `C`.

```
      A
     / \
    B   C
   / \  /
  D   E F
```

Postorder Traversal Algorithm (Left, Right, Root): - Left Subtree of A: `D, E, B` - Right Subtree of A: `F, C` - Root: `A` - **Postorder Sequence**: `D, E, B, F, C, A`

Correct Answer: **`D, E, B, F, C, A`**

PYQ 8: AVL Tree Insertion & Rotations

Problem: Insert keys `10, 20, 30, 40, 50, 25` sequentially into an initially empty AVL Tree. Identify all rotation types performed and draw final tree structure.

Step-by-Step Solution: 1. Insert `10`, then `20`, then `30`: - Tree becomes skewed right: $10 \to 20 \to 30$. Balance factor of root 10 is -2. - **Rotation 1**: **Left Rotation (RR Insertion)** at `10`.
- Subtree becomes: Root = `20`, Left = `10`, Right = `30`. 2. Insert `40`, then `50`: - Right child of `30` gets `40`, then right child of `40` gets `50`. - Subtree rooted at `30` becomes skewed right ($30 \to 40 \to 50$). Balance factor of `30` is -2. - **Rotation 2**: **Left Rotation (RR Insertion)** at `30`.
- Subtree becomes: Root = `40`, Left = `30`, Right = `50`. 3. Insert `25`: - Inserted as right child of `20` $\implies$ left child of `30`. - Tree nodes: Root `20`, Right child `40`. Left child of `40` is `30`, Left child of `30` is `25`. - Balance factor at root `20`: Height(Left)=1, Height(Right)=3 $\implies \text{BF} = 1 - 3 = -2$. - Right child `40` has $\text{BF} = 2 - 1 = +1$. - **Rotation 3**: **Right-Left (RL) Double Rotation** at root `20`. - First Right rotate at `40`, then Left rotate at `20`.

```
      30
     /  \
    20    40
   /  \     \
  10   25    50
```

Correct Answer: Final Root = `30`, Rotations Performed = `2 RR (Left Rotations), 1 RL Rotation`

PYQ 9: Dijkstra's Shortest Path Algorithm

Problem: Find shortest distance from source vertex **S** to all other vertices in the weighted graph: - Edges: $(S, A)=4$, $(S, B)=2$, $(A, B)=1$, $(A, C)=5$, $(B, C)=8$, $(B, D)=10$, $(C, D)=2$.

Step-by-Step Dijkstra Trace Table:

Iteration	Visited Set V	$D[S]$	$D[A]$	$D[B]$	$D[C]$	$D[D]$	Node Selected
Initial	$\{ \}$	0	∞	∞	∞	∞	**S**
1	$\{S\}$	0	4	2	∞	∞	**B** (Min = 2)
2	$\{S, B\}$	0	$\min(4, 2+1)=3$	2	$\min(\infty, 2+8)=10$	$\min(\infty, 2+10)=12$	**A** (Min = 3)
3	$\{S, B, A\}$	0	3	2	$\min(10, 3+5)=8$	12	**C** (Min = 8)
4	$\{S, B, A, C\}$	0	3	2	8	$\min(12, 8+2)=10$	**D** (Min = 10)

Final Shortest Distances from S: - $S \to A = 3$ (Path: $S \to B \to A$) - $S \to B = 2$ (Path: $S \to B$) - $S \to C = 8$ (Path: $S \to B \to A \to C$) - $S \to D = 10$ (Path: $S \to B \to A \to C \to D$)

Correct Answer: Shortest Distance to D = `10`

PYQ 10: Min-Heapify Array Construction

Problem: Given unsorted array `A = [9, 4, 7, 1, 10, 2, 6]`. Convert it into a **Min-Heap** using bottom-up `buildMinHeap()` algorithm. Show the step-by-step array modifications.

Step-by-Step Heapify Trace: Array indices (0-based): `A[0]=9, A[1]=4, A[2]=7, A[3]=1, A[4]=10, A[5]=2, A[6]=6`
Last non-leaf node index $= \lfloor n/2 \rfloor - 1 = \lfloor 7/2 \rfloor - 1 = 2$ (`A[2] = 7`).

1. **Heapify at i=2 (`A[2]=7`)**:
2. Children: Left `A[5]=2`, Right `A[6]=6`. Smallest child is `A[5]=2`.
3. Swap `A[2]` and `A[5]` $\implies$ Array: `[9, 4, 2, 1, 10, 7, 6]`.
4. **Heapify at i=1 (`A[1]=4`)**:
5. Children: Left `A[3]=1`, Right `A[4]=10`. Smallest child is `A[3]=1`.
6. Swap `A[1]` and `A[3]` $\implies$ Array: `[9, 1, 2, 4, 10, 7, 6]`.
7. **Heapify at i=0 (`A[0]=9`)**:
8. Children: Left `A[1]=1`, Right `A[2]=2`. Smallest child is `A[1]=1`.
9. Swap `A[0]` and `A[1]` $\implies$ Array: `[1, 9, 2, 4, 10, 7, 6]`.

10. Re-heapify at $i=1$ (`A[1]=9`): Children `A[3]=4` , `A[4]=10` . Smallest is `A[3]=4` .
11. Swap `A[1]` and `A[3]` $\implies$ Array: `[1, 4, 2, 9, 10, 7, 6]` .

Correct Answer: Final Min-Heap Array = **`[1, 4, 2, 9, 10, 7, 6]`**

PYQ 11: Infix to Postfix Conversion using Stack

Problem: Convert infix expression `(A + B * C) / (D - E)` to postfix using standard operator precedence and stack operations.

Step-by-Step Operator Stack Trace:

Symbol	Operator Stack	Output Postfix String
`(`	`(`	
`A`	`(`	`A`
`+`	`( +`	`A`
`B`	`( +`	`A B`
`*`	`( + *`	`A B`
`C`	`( + *`	`A B C`
`)`	``	`A B C * +`
`/`	`/`	`A B C * +`
`(`	`/ (`	`A B C * +`
`D`	`/ (`	`A B C * + D`
`-`	`/ ( -`	`A B C * + D`
`E`	`/ ( -`	`A B C * + D E`
`)`	`/`	`A B C * + D E -`
End	empty	**`A B C * + D E - /`**

Correct Answer: **`A B C * + D E - /`**

---

3. Basic Embedded Systems Solved PYQs

PYQ 12: 8051 UART Baud Rate & Timer Reload Calculation

Problem: An 8051 microcontroller operates with crystal frequency $f_{\text{osc}} = 11.0592 \text{ MHz}$. Timer 1 in Mode 2 (8-bit auto-reload) is used to generate a baud rate of **9600 bps** (with `SMOD = 0`). Calculate the reload value to be placed in `TH1` register.

Detailed Derivation: - 8051 Machine cycle frequency: $$f_{\text{machine}} = \frac{f_{\text{osc}}}{12} = \frac{11.0592 \text{ MHz}}{12} = 921.6 \text{ kHz}$$ - Timer 1 clock frequency to UART (SMOD=0): $ $f_{\text{timer}} = \frac{f_{\text{machine}}}{32} = \frac{921.6 \text{ kHz}}{32} = 28800 \text{ Hz}$$ - Baud rate equation: $$\text{Baud Rate} = \frac{28800}{256 - \text{TH1}}$$ $$9600 = \frac{28800}{256 - \text{TH1}} \implies 256 - \text{TH1} = \frac{28800}{9600} = 3$$ $$\text{TH1} = 256 - 3 = 253 = \text{FD}_{\text{hex}} = -3_{10}$$

Correct Answer: `253 (0xFD)`

PYQ 13: 12-Bit ADC Step Voltage & Output Calculation

Problem: A 12-bit ADC has reference voltage $V_{\text{ref}} = 3.3 \text{ V}$. 1. Calculate the **Step Size (LSB Resolution)**. 2. What is the digital output code when input analog voltage $V_{\text{in}} = 1.65 \text{ V}$?

Detailed Solution: 1. Step Size calculation: $$\text{Step Size} = \frac{V_{\text{ref}}}{2^n} = \frac{3.3 \text{ V}}{2^{12}} = \frac{3.3}{4096} \approx 0.00080566 \text{ V} = 0.80566 \text{ mV}$$ 2. Digital Output Code: $$\text{Digital Code} = \left\lfloor \frac{V_{\text{in}}}{\text{Step Size}} \right\rfloor = \left\lfloor \frac{1.65}{3.3} \times 4096 \right\rfloor = \lfloor 0.5 \times 4096 \rfloor = 2048 = 0\text{x}800$$

Correct Answer: Step Size = `0.80566 mV`, Digital Output Code = `2048 (0x800)`

PYQ 14: Interrupt Latency Calculation

Problem: A microcontroller runs at CPU clock frequency $f_{\text{clk}} = 16 \text{ MHz}$. Upon receiving an external interrupt request, the CPU takes: - 4 cycles to finish the current instruction - 12 cycles to push context onto the stack and jump to ISR address - The ISR preamble takes 8 cycles before executing the main handler code.

Calculate the total **Interrupt Latency** in microseconds (μs).

Detailed Solution: 1. Total clock cycles lost before executing main ISR logic: $$\text{Total Cycles} = 4 + 12 + 8 = 24 \text{ clock cycles}$$ 2. Clock Period T_{clk}: $$T_{\text{clk}} = \frac{1}{16 \text{ MHz}} = \frac{1}{16 \times 10^6 \text{ Hz}} = 0.0625 \text{ }\mu\text{s}$$ 3. Total Interrupt Latency: $$\text{Latency} = 24 \times 0.0625 \text{ }\mu\text{s} = 1.50 \text{ }\mu\text{s}$$

Correct Answer: `1.50 µs`

PYQ 15: Timer PWM Frequency & Duty Cycle Calculation

Problem: An ARM Cortex-M timer clocked at $f_{\text{timer}} = 48 \text{ MHz}$ uses a prescaler value $PSC = 47$ (so prescaler divisor is $PSC + 1 = 48$). 1. Find the Auto-Reload Register (ARR) value needed to produce a PWM frequency of **1 kHz**. 2. Find the Capture Compare Register (CCR) value for a **75% Duty Cycle**.

Detailed Solution: 1. Counter Clock Frequency f_{cnt}: $$f_{\text{cnt}} = \frac{f_{\text{timer}}}{PSC + 1} = \frac{48 \text{ MHz}}{48} = 1 \text{ MHz}$$ 2. Desired PWM Period $T_{\text{pwm}} = \frac{1}{1 \text{ kHz}} = 1 \text{ ms} = 1000 \text{ }\mu\text{s}$. $$\text{PWM Frequency} = \frac{f_{\text{cnt}}}{ARR + 1} \implies 1000 = \frac{1,000,000}{ARR + 1}$$ $$ARR + 1 = 1000 \implies ARR = 999$$ 3. CCR for 75% Duty Cycle: $$

\text{Duty Cycle} = \frac{CCR + 1}{ARR + 1} \implies 0.75 = \frac{CCR + 1}{1000}$$ $$CCR + 1 = 750 \implies CCR = 749$$

Correct Answer: $ARR = \mathbf{999}$, $CCR = \mathbf{749}$

---

4. Computer Networks Solved PYQs

PYQ 16: CIDR Subnetting & IP Range Calculation

Problem: An IP address is `192.168.10.138/27`. Determine: 1. Subnet Mask 2. Network Address (Network ID) 3. Broadcast Address 4. Total number of Usable Host IPs

Detailed Solution: 1. Subnet Mask (`/27`): - First 27 bits are `1`s: `11111111.11111111.11111111.11100000` - In decimal: `255.255.255.224` 2. Block Size in 4th octet $= 256 - 224 = 32$. 3. Multiples of 32 for 4th octet: $0, 32, 64, 96, 128, 160, 192, 224$. - The 4th octet `138` falls between `128` and `159`. - **Network ID**: `192.168.10.128` - **Broadcast ID**: `192.168.10.159` - **First Usable Host**: `192.168.10.129` - **Last Usable Host**: `192.168.10.158` 4. Usable Host Count $= 2^{(32 - 27)} - 2 = 2^5 - 2 = 32 - 2 = 30$ hosts.

Correct Answer: Network ID = **`192.168.10.128`**, Broadcast ID = **`192.168.10.159`**, Usable Hosts = **`30`**

PYQ 17: Sliding Window Protocol Efficiency

Problem: A point-to-point link has bandwidth $B = 10 \text{ Mbps}$, Round Trip Time $RTT = 40 \text{ ms}$, and packet frame size $L = 1000 \text{ bytes} = 8000 \text{ bits}$. 1. Calculate Transmission Time T_t. 2. Find the minimum Window Size N required for **100% Channel Utilization**.

Detailed Solution: 1. Transmission Time T_t: $$T_t = \frac{L}{B} = \frac{8000 \text{ bits}}{10 \times 10^6 \text{ bps}} = 0.0008 \text{ s} = 0.8 \text{ ms}$$ 2. Propagation Delay $T_p = \frac{RTT}{2} = \frac{40 \text{ ms}}{2} = 20 \text{ ms}$. 3. For 100% Efficiency ($\eta = 1.0$): $$\eta = \frac{N \cdot T_t}{T_t + 2 T_p} \ge 1.0$$ $$N \ge \frac{T_t + 2 T_p}{T_t} = 1 + \frac{2 T_p}{T_t} = 1 + \frac{40 \text{ ms}}{0.8 \text{ ms}} = 1 + 50 = 51 \text{ frames}$$

Correct Answer: $T_t = \mathbf{0.8 \text{ ms}}$, Min Window Size $N = \mathbf{51 \text{ frames}}$

PYQ 18: TCP Congestion Window Tracing

Problem: A TCP connection starts with initial congestion window $\text{cwnd} = 1 \text{ MSS}$ and slow start threshold $\text{ssthresh} = 16 \text{ MSS}$. Tracing transmission rounds: - Rounds 1 to 6 complete successfully. - At Round 7, a **Timeout** occurs.

What is the cwnd and ssthresh value at Round 8?

Step-by-Step Transmission Round Trace: - **Round 1**: $\text{cwnd} = 1$ (Slow Start: doubles each round) - **Round 2**: $\text{cwnd} = 2$ - **Round 3**: $\text{cwnd} = 4$ - **Round 4**: $\text{cwnd} = 8$ - **Round 5**: $\text{cwnd} = 16$ (Reaches $\text{ssthresh} = 16$. Switch to Congestion Avoidance: $+1$ per round) - **Round 6**: $\text{cwnd} = 17$ - **Round 7**: Timeout occurs at $\text{cwnd} = 17$! - New $\text{ssthresh} = \lfloor

\frac{\text{cwnd}}{2} \rfloor = \lfloor \frac{17}{2} \rfloor = 8 \text{ MSS}$. - cwnd resets to 1 MSS. - **Round 8**: System enters Slow Start with $\text{cwnd} = 1 \text{ MSS}$ (will double to 2, 4, 8).

Correct Answer: $\text{cwnd} = \mathbf{1 \text{ MSS}}$, $\text{ssthresh} = \mathbf{8 \text{ MSS}}$

PYQ 19: Cyclic Redundancy Check (CRC) Calculation

Problem: Given message bit sequence $M = 110101$ and generator polynomial divisor $G = 1011$ ($x^3 + x + 1$). Compute the **CRC Remainder Bits** to be appended to the message.

Step-by-Step Modulo-2 Division: 1. Degree of generator $G = 1011$ is $k = 3$. Append 3 zeros to message: $M' = 110101000$. 2. Modulo-2 Division ($110101000 \div 1011$ using XOR):

```
         111101
       _________________
1011 | 110101000
       1011
       ----
        1100
        1011
        ----
         1111
         1011
         ----
          1000
          1011
          ----
           0110
           0000
           ----
            1100
            1011
            ----
             011 (Remainder)
```

1. Remainder bits $= 011$. Transmitted codeword $= 110101011$.

Correct Answer: CRC Remainder = `011`, Transmitted Codeword = `110101011`

---

5. Software Engineering / SDLC Solved PYQs

PYQ 20: Basic COCOMO Model Estimation

Problem: Estimate the Effort (E) and Development Time (D) for an **Organic Mode** software project with size **50 KLOC**.
COCOMO Constants for Organic Mode:
$a_b = 2.4, \quad b_b = 1.05, \quad c_b = 2.5, \quad d_b = 0.38$

Detailed Solution: 1. **Effort Calculation (E)** in Person-Months (PM): $$E = a_b \times (\text{KLOC})^{b_b} = 2.4 \times (50)^{1.05}$$ $$(50)^{1.05} = 50^1 \times 50^{0.05} \approx 50 \times 1.216 = 60.8$$ $$E = 2.4 \times 60.8 \approx 145.92 \text{ Person-Months}$$

1. **Development Time (D)** in Months: $$D = c_b \times (E)^{d_b} = 2.5 \times (145.92)^{0.38}$$ $$(145.92)^{0.38} \approx 6.64$$ $$D = 2.5 \times 6.64 \approx 16.6 \text{ Months}$$

2. **Average Staffing Level**: $$\text{Staff} = \frac{E}{D} = \frac{145.92}{16.6} \approx 8.79 \approx 9 \text{ Persons}$$

Correct Answer: Effort = `145.92 Person-Months`, Development Time = `16.6 Months`

PYQ 21: Function Point (FP) Analysis

Problem: A software project has the following element counts (all average complexity): - External Inputs (EI): 15 (weight 4) - External Outputs (EO): 10 (weight 5) - External Inquiries (EQ): 8 (weight 4) - Internal Logical Files (ILF): 4 (weight 10) - External Interface Files (EIF): 2 (weight 7)

If the sum of 14 Value Adjustment Factors $\sum F_i = 42$, compute the **Final Adjusted Function Points (FP)**.

Detailed Solution: 1. **Unadjusted Function Points (UFP)**: $$UFP = (15 \times 4) + (10 \times 5) + (8 \times 4) + (4 \times 10) + (2 \times 7)$$ $$UFP = 60 + 50 + 32 + 40 + 14 = 196 \text{ UFP}$$

1. **Value Adjustment Factor (VAF / CAF)**: $$\text{VAF} = 0.65 + (0.01 \times \sum F_i) = 0.65 + (0.01 \times 42) = 0.65 + 0.42 = 1.07$$

2. **Final Adjusted Function Points (FP)**: $$FP = UFP \times \text{VAF} = 196 \times 1.07 = 209.72 \text{ FP}$$

Correct Answer: UFP = `196`, Final FP = `209.72`

---

6. Software QA & Testing Solved PYQs

PYQ 22: Cyclomatic Complexity & Basis Path Testing

Problem: Consider the following code function:

```
void process_data(int x, int y) {
    while (x > 0) {          // Decision 1
        if (y == 10) {       // Decision 2
            do_something();
        } else if (y > 20) { // Decision 3
            do_other();
        }
        x--;
```

```
    }
}
```

1. Compute **Cyclomatic Complexity $V(G)$**.
2. State the number of **Independent Basis Paths**.

Detailed Solution: 1. Identify Predicate Nodes (P): - P_1: `while (x > 0)` - P_2: `if (y == 10)` - P_3: `else if (y > 20)` - Total Predicate Nodes $P = 3$.

1. Formula for Cyclomatic Complexity: $$V(G) = P + 1 = 3 + 1 = 4$$

(Alternatively, using Regions in Control Flow Graph: $R = 4$).

1. Number of Independent Basis Paths $= V(G) = 4$.

Correct Answer: Cyclomatic Complexity $V(G) = \mathbf{4}$, Independent Paths = $\mathbf{4}$

PYQ 23: Boundary Value Analysis (BVA) & Equivalence Partitioning

Problem: A input field in an online application form accepts age values between **18** and **65** inclusive ($18 \le \text{Age} \le 65$).
Define Equivalence Partitions and construct **Boundary Value Analysis (BVA) Test Cases**.

Step-by-Step Partitioning & Test Suite:

1. Equivalence Partitions (EP):

- **Invalid Partition 1**: $\text{Age} < 18$ (e.g., Age = 10)
- **Valid Partition**: $18 \le \text{Age} \le 65$ (e.g., Age = 30)
- **Invalid Partition 2**: $\text{Age} > 65$ (e.g., Age = 70)

2. Boundary Value Analysis (BVA - 2-Value & 3-Value Boundary Points):

- Boundaries are at 18 and 65.

Boundary Point	Test Value	Expected Output	Partition Type
Min - 1	17	REJECT / Error	Invalid Boundary
Min	18	ACCEPT	Valid Min Boundary
Min + 1	19	ACCEPT	Valid Nominal
Nominal	40	ACCEPT	Valid Midpoint
Max - 1	64	ACCEPT	Valid Nominal
Max	65	ACCEPT	Valid Max Boundary
Max + 1	66	REJECT / Error	Invalid Boundary

Correct Answer: Total Minimal BVA Test Suite = `7 Test Cases (17, 18, 19, 40, 64, 65, 66)`

PYQ 24: Decision Table Reduction for Test Suite Minimization

Problem: An e-commerce system determines shipping discount based on 3 conditions: - C_1: Is Premium Member? (T/F) - C_2: Order Amount > $100? (T/F) - C_3: Coupon Code Valid? (T/F)

Discount Rules: - If Premium Member ($C_1 = T$), discount is ALWAYS applied regardless of C_2 or C_3. - If Not Premium Member ($C_1 = F$), discount is applied ONLY IF Order Amount > $100 AND Coupon is Valid ($C_2=T \land C_3=T$).

Construct the initial 8-rule Decision Table and reduce it to the **Minimal Required Test Cases**.

Step-by-Step Reduction:

Initial Decision Table (8 Combinations $2^3 = 8$):

Condition / Action	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**
C_1: Premium Member	T	T	T	T	F	F	F	F
C_2: Order > $100	T	T	F	F	T	T	F	F
C_3: Valid Coupon	T	F	T	F	T	F	T	F
Action: Apply Discount	**YES**	**YES**	**YES**	**YES**	**YES**	**NO**	**NO**	**NO**

Rule Reduction:

- Rules R1, R2, R3, R4 all have $C_1 = T$ and result in Action = **YES**. They collapse into **1 Combined Rule**: ($C_1=T, C_2=D, C_3=D \implies \text{YES}$).
- Rule R5 ($C_1=F, C_2=T, C_3=T$) $\implies$ **YES**.
- Rules R6, R7, R8 ($C_1=F$ and not both $C_2=T, C_3=T$) $\implies$ **NO**.

Minimal Test Suite (3 Consolidated Test Scenarios):

1. **Scenario 1**: Any Premium Member $\implies$ **Discount YES**
2. **Scenario 2**: Non-Premium + Order > $100 + Valid Coupon $\implies$ **Discount YES**
3. **Scenario 3**: Non-Premium + (Order $\le$ $100 OR Invalid Coupon) $\implies$ **Discount NO**

Correct Answer: Reduced Minimal Test Suite = `3 Consolidated Rules`

---

7. 🎯 Real CoCubes Technical Placement PYQs & Exact Question Patterns Bank

PYQ 25: Operating Systems - Counting Semaphore State & Queue Tracing

Problem: A counting semaphore S is initialized to 7. A series of process operations are executed in sequence: - 10 processes execute `wait(S)` (P operation) - 4 processes execute `signal(S)` (V operation)

Calculate: 1. The final numerical value of semaphore S. 2. The number of processes currently blocked in the waiting queue.

Detailed Step-by-Step Solution: 1. Initial value: $S = 7$. 2. Executing 10 `wait(S)` operations: - Each `wait(S)` decrements S by 1. - $S_{\text{new}} = 7 - 10 = -3$. - The negative value -3 indicates that 3 processes could not acquire a resource and were placed in the blocked queue. 3. Executing 4 `signal(S)` operations: - Each `signal(S)` increments S by 1 and wakes up a blocked process if $S \le 0$. - $S_{\text{final}} = -3 + 4 = +1$. 4. Queue Evaluation: - Since $S_{\text{final}} = +1 > 0$, all blocked processes have been unblocked and 1 resource instance remains available! - Number of blocked processes $= 0$.

Correct Answer: Final $S = \mathbf{+1}$, Blocked Processes = $\mathbf{0}$

PYQ 26: Operating Systems - FIFO Belady's Anomaly Numerical Proof

Problem: A process references pages in the following sequence:
`1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5`
Calculate the total number of page faults using the **FIFO** page replacement algorithm for: 1. Frame capacity $= 3$ 2. Frame capacity $= 4$
Does this reference string demonstrate **Belady's Anomaly**?

Detailed Step-by-Step Execution Trace:

1. Frame Capacity = 3 Frames:

Ref	F0	F1	F2	Status
1	1	-	-	Fault (1)
2	1	2	-	Fault (2)
3	1	2	3	Fault (3)
4	4	2	3	Fault (4) (Evict 1)
1	4	1	3	Fault (5) (Evict 2)
2	4	1	2	Fault (6) (Evict 3)
5	5	1	2	Fault (7) (Evict 4)

Ref	F0	F1	F2	Status
1	5	1	2	HIT
2	5	1	2	HIT
3	5	3	2	Fault (8) (Evict 1)
4	5	3	4	Fault (9) (Evict 2)
5	5	3	4	HIT

- Total Faults (3 Frames) $= \mathbf{9 \text{ Page Faults}}$.

2. Frame Capacity = 4 Frames:

Ref	F0	F1	F2	F3	Status
1	1	-	-	-	Fault (1)
2	1	2	-	-	Fault (2)
3	1	2	3	-	Fault (3)
4	1	2	3	4	Fault (4)
1	1	2	3	4	HIT
2	1	2	3	4	HIT
5	5	2	3	4	Fault (5) (Evict 1)
1	5	1	3	4	Fault (6) (Evict 2)
2	5	1	2	4	Fault (7) (Evict 3)
3	5	1	2	3	Fault (8) (Evict 4)
4	4	1	2	3	Fault (9) (Evict 5)
5	4	5	2	3	Fault (10) (Evict 1)

- Total Faults (4 Frames) $= \mathbf{10 \text{ Page Faults}}$.

Conclusion: Since increasing frames from 3 to 4 increased page faults from 9 to 10, **Belady's Anomaly IS DEMONSTRATED**.

Correct Answer: 3 Frames = `9 Faults`, 4 Frames = `10 Faults`, Belady's Anomaly = `YES`

PYQ 27: Basic Embedded Systems - 8051 IVT Address & Priority Resolution

Problem: An 8051 microcontroller receives an external interrupt on pin `P3.3` (`INT1`) while simultaneously Timer 0 overflows (`TF0`).

The Interrupt Priority register `IP` is set to `0x04` (`PX1 = 1` , all other bits `0`).
To which program memory address will the CPU vector first?

Detailed Derivation: 1. Standard Vector Addresses: - `INT0` : `0x0003` - `TF0` : `0x000B` - `INT1` : `0x0013` - `TF1` : `0x001B` 2. Default Natural Hardware Priority: `INT0 > TF0 > INT1 > TF1 > Serial` . - Without software priority, `TF0` (`0x000B`) would execute before `INT1` (`0x0013`). 3. Software Priority Override (`IP = 0x04`): - Bit `IP.2` (`PX1`) is set to `1` , raising `INT1` to **High Priority**. - High priority interrupts always override low priority interrupts regardless of natural hardware order. 4. Therefore, CPU vectors immediately to `INT1` address `0x0013` .

Correct Answer: **`0x0013 (0013H)`**

PYQ 28: Basic Embedded Systems - 8051 UART Mode 1 Reload Register Math

Problem: Calculate the exact value (in Hexadecimal) to be loaded into `TH1` to establish a **4800 Baud Rate** for an 8051 operating with crystal $f_{\text{osc}} = 11.0592 \text{ MHz}$ and `SMOD = 0` .

Detailed Derivation: 1. Machine Clock Frequency $f_{\text{mc}} = \frac{11.0592 \text{ MHz}}{12} = 921.6 \text{ kHz}$. 2. Timer 1 Input Frequency to UART $= \frac{921.6 \text{ kHz}}{32} = 28800 \text{ Hz}$. 3. Baud Rate Equation: $$\text{Baud Rate} = \frac{28800}{256 - \text{TH1}}$$ $$4800 = \frac{28800}{256 - \text{TH1}} \implies 256 - \text{TH1} = \frac{28800}{4800} = 6$$ $$\text{TH1} = 256 - 6 = 250_{10} = \text{0xFA}$$

Correct Answer: **`0xFA`**

PYQ 29: Computer Networks - TCP 3-Way Handshake & Sequence Number Tracing

Problem: Host A opens a TCP connection to Host B: - Host A sends `SYN` with $Seq = 4000$. - Host B accepts and replies with `SYN-ACK` with $Seq = 8000$. - Host A completes handshake with `ACK` and immediately transmits a data segment containing 600 bytes of payload.

Calculate: 1. Acknowledgment number in Host B's `SYN-ACK` . 2. Sequence number of Host A's data segment. 3. Acknowledgment number expected in Host B's reply to Host A's data segment.

Detailed Step-by-Step Solution: 1. **Host B's `SYN-ACK` Ack Number**: - `SYN` flag consumes 1 sequence number. - $Ack_{\text{Host B}} = 4000 + 1 = \mathbf{4001}$. 2. **Host A's Data Segment Sequence Number**: - Host A's final handshake `ACK` uses $Seq = 4001$. - The subsequent data segment starts at $Seq = \mathbf{4001}$. 3. **Host B's Data ACK Number**: - Payload size $= 600$ bytes. - $Ack_{\text{expected}} = Seq_{\text{start}} + \text{Payload Size} = 4001 + 600 = \mathbf{4601}$.

Correct Answer: 1) **`4001`** , 2) **`4001`** , 3) **`4601`**

PYQ 30: Software QA - Cyclomatic Complexity with Switch Statements

Problem: Calculate the Cyclomatic Complexity $V(G)$ for a C function containing: - 1 `while` loop - 2 `if-else` statements - 1 `switch` statement with 4 `case` labels (and 1 `default`)

Detailed Solution: 1. Predicate (Decision) Node Breakdown: - `while` loop $\implies 1$ decision node. - 2 `if-else` statements $\implies 2$ decision nodes. - `switch` statement with $N = 4$ cases has $N - 1 = 3$ decision branches. - Total Predicate Nodes $P = 1 + 2 + 3 = 6$. 2. Formula: $$V(G) = P + 1 = 6 + 1 = 7$$

Correct Answer: Cyclomatic Complexity $V(G) = \mathbf{7}$