Ans 1.) Machine : 16-bit words, Instruction format : 4-bit opcode + 12-bit direct address.

Opcode given :-

- 0001 = Load Ac from memory
- 0010 = Store Ac to memory
- 0110 = Subtract from AC

Program (3 instructions). PC is set to 300, so instruction at 301, 302, 303:-

1) Load AC, 940
   → opcode = 0001
   → address 940 in 12-bit binary = 001110101100
   → Instruction word: 0001 001110101100
2) Subtract M[941] from AC
   → opcode = 0110
   → address 941 = 001110101101
   → instruction: 0110 001110101101
3) Store AC to 941
   → opcode = 0010
   → address 941 = 001110101101
   → instruction = 0010 001110101101

Assume Memory : M[940] = 10, M[941] = 5. Ac initial unknown.

Fetch :-

Inst @ 301: Load AC, 940

- Fetch –
  - MAR ← PC(301), memory read M[301] → MBR ← memory[301], IR ← MBR, PC ← 302.
- Decode –
  → opcode = 940

Name: Poushali

Regd. Number: 2341013223

→ MAR ← 940, read memory: MBR ← M[940]
→ AC ← MBR (AC = 10)
→ Result : AC = 10

Inst @ 302:

- Fetch:~
  :→ MAR ← PC (302), MBR ← memory [302], IR ← MBR, PC ← 303.
- Decode
  :→ MAR ← 941, read M[941] → MBR = 5
  :→ AC ← AC − MBR = 10 − 5 = 5
  :→ Result: AC = 5

Inst @ 303:

- Fetch:
  MAR ← PC(303), read instruction to MBR → IR ← MBR, PC ← 304.
- Decode:
  → MAR ← 941, MBR ← AC(5), write memory: M[941] ← MBR(5).
  → Result: M[941] becomes 5

Final State : AC :~ 5; M[940] = 10 ; M[941] = 5

Ans 2.)

Steps	operations	Explanation
1	MAR ← PC, MBR ← M[MAR], IR ← MBR, PC ← PC+1	Fetch instruction
2	Decode opcode & address	Identify operation
3	For Load	MAR ← 940, MBR ← M[940], AC ← MBR
4	For Subtract	MAR ← 941, MBR ← M[941], AC ← AC − MBR
5	For Store	MAR ← 941, MBR ← AC, M[MAR] ← MBR

Name: Paushali Regd. Number: 2341013223

Ans 3.) Given,
Cache access time = 50 ns
Main memory access = 550 ns
Reads = 80%
Write = 20%
Read hit = 0.9

Now,
AAT = 0.8 [0.9 × 50 + 0.1 × 550] + 0.2 × 550
= 0.8 × 100 + 110
= 190 ns
i.e. Average Access Time = 190 ns.

Ans 4.)

Job	CPU %	time (min)	Memory (MB)
Job 1	70	10	150
Job 2	10	20	100
Job 3	10	15	125

Total mem = 375 MB.

(a) Uniprogramming :~ Runs one by one
CPU time used = 10 × 0.7 + 20 × 0.1 + 15 × 0.1
= 10.5 min
Total time = 45 min
→ CPU util = 10.5/45 = 23.3%.
→ Mem util = (150 × 10 + 100 × 20 + 125 × 15)/45
= 29.9%.
→ Throughput = 3/45 = 0.067 job/min

(b) Multiprogramming :~ All jobs in memory → total mem = mem = 375/400 = 93.7(5) %.
→ CPU util = 1 − (0.3 × 0.9 × 0.9) = 75.7%.
→ Throughput = 3/20 = 0.15 job/min

Name: Paushali Regd. Number: 2341013223

Ans 5.)

Job	Total	CPU	I/O (each)
Job1	23	3	10
Job2	29	5	12
Job3	14	4	5

Uniprogramming :~
Total = 66 ms, CPU = 12 ms
→ CPU util = 18.2 %.

Multiprogramming :~
overlapping CPU/I/O reduces idle time
→ CPU busy 12 ms / total 30 ms
= 40 %
∴ CPU utilisation improves from 18.2% → 40%

Ans 6.) a) Trace:
→ Start at P0
→ C1 = fork(); → creates P1 (child)
→ In P0: C1 > 0 → skip C2 = fork(); (so C2 stays 0).
→ In P1: C1 == 0 → executes C2 = fork(); which creates P2.
→ In P1: C2 > 0 (true) → executes fork() producing P3.
→ In P2: C2 == 0 → does not execute the if (C2 > 0) fork.

Graph: -

P0
└ P1
  ├ P2
  └ P3 (P3 is child created by P1's "if (C2>0)")

Name: Paushali Regd. Number: 23410133223

fork();

b.) → Start: P0 (C1 = 1, C2 = 1).
→ C1 = fork(); → creates P1.
→ In parent P0: C1 != 0 true → do C2 = fork(); creates P2.
→ In P0: after that C2 > 0 so if (C2 == 0) false → no extra fork.
→ In P2: C2 == 0 true → executes fork() creating P3.
→ In child P1: C1 == 0 → skip second fork();
C2 stays 1 → if (C2 == 0) false.

Graph:

P0
├ P1 (child from first fork)
└ P2 (child from second fork by P0)
  └ P3 (child from P2's "if (C2 == 0) fork();")

c) Trace :-
• Start P0
• First fork() creates A.
• In parent P0: first fork() returns > 0 → (true || ...) short-circuits; parent then executes fork() inside the if (the third fork) producing C.
• In child A: first fork() returned 0 → evaluate second fork() (the right operand of ||) this second fork() (call it B) creates B (child of A). Results:

Name: Paushali Regd. Number: 23410133223

→ In A: second fork() returned >0 → (0||>0) = true → A executes third fork() producing D.

→ In B: second fork() returned 0 → (0||0) = false → B doesn't execute the third fork.

Concept:

P0
- A
  - B
  - D (A's third fork)
- C (P0's third fork)

d) Trace:
→ Start: P0. First fork() creates A.
→ In P0(parent): first fork() > 0 (true) → evaluate !fork():
→ P0 executes second fork() (call it C):- In parent P0, second fork() returns >0 → !>0 ⇒ false → first if condition false → P0 doesn't enter inner if.
→ In child C: inherits the state where the first fork() returns was > 0 (true), second fork() returned 0 → !0 → true → overall true && true so, C enters the if inner if.
→ In A: first fork() returned 0 (false) → Short-circuit && so A does not execute second fork and does not enter the inner if.

Name: Poushali Regd. Number: 23410132223

→ Now only process C continues into the inner if:
→ C executes if (fork() || fork()) fork();
→ First fork here creates D.
→ In C: first inner-fork returned >0 (true || ...) short-circuits → condition true → C runs final fork() producing F.
→ In D: first inner-fork returned 0 → D evaluates the second fork() producing E.
→ In D: second inner-fork returns >0 → (0||>0) = true → D executes final fork() producing G.
→ In E: second inner-fork returns 0 → (0||0) false → E does not execute final fork.

Concept:

P0
- A (from first fork)
- C (child created by P0's second fork)
  - D (C's first inner fork)
    - E (D's second inner fork)
  - F (C's final fork)
  - G (D's final fork)

Name: Poushali Regd. Number: 23410132223

Ans7:)

Process	Arrival	Burst
P1	0	3
P2	1	1
P3	3	3
P4	4	x

Average waiting time = 1 → total waiting = 4.
After simulation under SRTF:

- $w_1 = 1, w_2 = 0, w_3 = 1 + x, w_4 = 0$
  → $1 + 0 + (1 + x) + 0 = 4 \Rightarrow x = 2$

Ans : $x = 2$ ms

Ans 8:)

Process	Arrival	Burst	Priority
P1	0	11	1
P2	0	8	0
P3	12	2	3
P4	2	6	2
P5	9	16	4

Algorithm	Avg turnaround	Avg waiting	Avg Response
FCFS	23.2	14.6	14.6
SJF	17.0	8.4	8.4
SRTF	17.0	8.4	8.4
Non-preemptive Priority	22.4	15.8	15.8
Preemptive Priority	23.0	14.4	14.4
HRRN	18.8	10.2	10.2
RR (q=2)	23.8	15.2	15.2

Name: Poushali Regd. Number: 23410132223

Minimum Avg Waiting Time : 8.4 (SJF/SRTF)

Ans9:)

Process	Arrival	Burst	Priority	Algorithm
P1	0	4	3	FCFS
P2	0	2	1	SJF
P3	1	3	2	SRTF
P4	2	2	4	~~Non-Preemptive~~ priority

Algorithm	Avg turnaround	Avg waiting
FCFS	6.75	4.00
SJF	5.25	2.50
SRTF	5.25	2.50
Non-preemptive Priority	6.00	3.25
Preemptive Priority	6.00	3.25
HRRN	6.50	3.75
RR (q=2)	7.00	4.25

Minimum Avg waiting Time: 2.5 (SJF/SRTF)

Ans10:) Given

- Queues:
  → Q1 → Rond Robin (quantum=3)
  → Q2 → Rond Robin (quantum=5)
  → Q3 → FCFS
- All process arrive at time 0

Name: Poushali Regd. Number: 23410132223

Process	Burst Time
P1	8
P2	22
P3	4
P4	12

Stepwise Execution

Q1 (RR, q=3):

→ P1: 0–3 → rem 5 → to Q2
→ P2: 3–6 → rem 19 → to Q2
→ P3: 6–9 → rem 1 → to Q2
→ P4: 9–12 → rem 9 → to Q2

Q2 (RR, q=5):

→ P1: 12–17 → finishes
→ P2: 17–22 → rem 14 → to Q3
→ P3: 22–33 → finishes
→ P4: 23–28 → rem 4 → to Q3

Q3 (FCFS):

- P2: 28–42 → finishes
- P4: 42–46 → finishes

Result

Process	completion	Turnaround	waiting
P1	17	17	9
P2	42	42	20
P3	23	23	19
P4	46	46	34

Average Turnaround Time = 32.0
Average waiting Time = 20.5

Name: Poorshali Regd. Number: 2341013223