1.

a) For detecting sequence "010" with a Mealy machine, we need 4 states:

- **S0**: Initial state (no progress)
- **S1**: Just received '0'
- **S2**: Received "01"
- **S3**: Received "010" (output Z=1)

S0 --0/0--> S1
S0 --1/0--> S0
S1 --0/0--> S1
S1 --1/0--> S2 (Output Z=1 when completing "010")
S2 --0/1--> S1
S2 --1/0--> S0

b)

Current State	Input A	Next State	Output Z
S0	0	S1	0
S0	1	S0	0
S1	0	S1	0
S1	1	S2	0
S2	0	S1	1
S2	1	S0	0

C.

Using state encoding: S0=00, S1 = 01, S2= 10
For Q1(MSB):

A/Q1Q0	00	01	10	11
0	0	0	0	X
1	0	1	0	X

Q1+ = A* Q0* Q'1

For Q0(LSB):

A/Q1Q0	00	01	10	11
0	1	1	0	X
1	0	0	0	X

Q0+ = A’ * Q1’

d. For Output Z:

A/Q1Q0	00	01	10	11
0	0	0	1	X
1	0	0	0	X

Z = A’ * Q1 * Q0’

2.

a)

For Moore machine, we need 4 states with outputs assigned to states:

- **S0**: Initial state (Z=0)
- **S1**: Just received '0' (Z=0)
- **S2**: Received "01" (Z=0)
- **S3**: Received "010" (Z=1)

Current State	Input A	Next State	Output Z
S0	0	S1	0
S0	1	S0	0
S1	0	S1	0
S1	1	S2	0
S2	0	S3	0

Current State	Input A	Next State	Output Z
S2	1	S0	0
S3	0	S1	1
S3	1	S0	1

c)

Using state encoding: S0=00, S1=01, S2=10, S3=11

For Q1+ (MSB):

A\Q1Q0	**00**	**01**	**10**	**11**
0	0	0	1	0
1	0	1	0	0

$Q1+ = \bar{A} \cdot Q1 \cdot \bar{Q}0 + A \cdot \bar{Q}1 \cdot Q0$

For Q0+ (LSB):

A\Q1Q0	**00**	**01**	**10**	**11**
0	1	1	1	1
1	0	1	0	0

$Q0+ = \bar{A} + A \cdot \bar{Q}1 \cdot Q0$

d) Output Equation

For Output Z:

Q1Q0 Z

00 0

01 0

10 0

11 1

Z = Q1 * Q0

3)

a) State Transition Table

Current State	Input A	Next State	Output Z
000	0	000	0
000	1	001	0
001	0	010	1
001	1	011	1
010	0	100	0
010	1	101	0
011	0	110	1
011	1	111	1
100	0	000	0
100	0	001	0
101	0	010	1
101	0	011	1
110	0	100	0
110	0	101	0
111	0	110	1

111	0	111	1

b) This is a Mealy Machine because the output depends on both the current state and the input. From the diagram we can see that transitions are labeled with "input/output" format, indicating the output changes based on the input-state combination, not just the state alone.

4.

a) Counting down from 12 (1100) to 1 (0001

Current State	Next State	Decimal
1100	1011	12->11
1011	1010	11->10
1010	1001	10->9
1001	1000	9->8
1000	0111	8->7
0111	0110	7->6
0110	0101	6->5
0101	0100	5->5
0100	0011	4->3
0011	0010	3->2
0010	0001	2->1
0001	0001	1->1

b)

For J-K flip-flops: J=1,K=0 → Set, J=0,K=1 → Reset, J=1,K=1 → Toggle, J=0,K=0 → Hold

For Q3 (MSB): K-map analysis shows:

- J3 = 0 (never need to set Q3 from 0→1 in countdown)
- K3 = $\bar{Q}2 \cdot \bar{Q}1 \cdot \bar{Q}0$ (clear when reaching 0111→0110)

For Q2:

- $J2 = \bar{Q}3\cdot\bar{Q}1\cdot\bar{Q}0$ (set when going from 0111→1000 doesn't happen in countdown)
- $K2 = Q3\cdot Q1\cdot Q0$ (clear when going from 1xxx→0xxx at 1000→0111)

For Q1:

- $J1 = \bar{Q}3\cdot\bar{Q}2\cdot\bar{Q}0 + Q3\cdot Q2\cdot\bar{Q}0$ (toggle pattern analysis)
- $K1 = Q0$ (clear when Q0=1 in most countdown steps)

For Q0 (LSB):

- $J0 = \bar{Q}3\cdot\bar{Q}2\cdot\bar{Q}1 + Q3\cdot Q2\cdot Q1$ (set in alternating pattern)
- $K0 = 1$ (always try to clear since we're counting down)

5
a)
Lucky(WG=1, GS=0)
- $RE\cdot 4LC$ → Lucky(stay lucky with Rainbow's End OR 4-Leaf Clover)
- $\bar{R}E\cdot\bar{4}LC$ → Unlucky(lose both to become unlucky)

Unlucky(WG=0, GS=1)
- $RE\cdot 4LC$ → Lucky(need both to become lucky)
- $\bar{R}E + \bar{4}LC$ → Unlucky(stay unlucky without both)

b)
This is a Moore Machine because the outputs (WG and GS) depend only on the current state (Lucky or Unlucky), not on the inputs. The outputs are constant for each state regardless of input values.
c)

Current State (Q)	RE	4LC	Next State(Q+)	WG	GS
0(Unlucky	0	0	0	0	1
0(Unlucky	0	1	0	0	1
0(Unlucky	1	0	0	0	1
0(Unlucky	1	1	1	0	1

1(Lucky)	0	0	0	1	0
1(Lucky)	0	1	1	1	0
1(Lucky)	1	0	1	1	0
1(Lucky)	1	1	1	1	0

d.
Using K-Map

Q/ RE* 4LC	00	01	10	11
0	0	0	0	1
1	0	1	1	1

Q+ = RE*4LC + Q(RE + 4LC)

e.Output Equations
WG = Q (Wish granted when Lucky)
GS = $\bar{Q}$(Gold stolen when Unlucky)

6.

Traffic Light Controller System- Real-World FSM Application

Traffic light controllers are excellent examples of finite state machines used in real-world applications. Modern traffic management systems use sophisticated FSM implementations to control intersection timing, pedestrian crossings, and emergency vehicle prioritization.

System Description: A typical four-way intersection controller manages traffic flow using multiple sensors and timing mechanisms. The system includes inductive loop detectors embedded in the road surface, pedestrian push buttons, emergency vehicle preemption detectors, and sometimes camera-based vehicle detection systems.

States and Transitions: The basic controller operates through several primary states:

- **North-South Green (NSG)**: North-south traffic flows while east-west traffic stops
- **North-South Yellow (NSY)**: Warning phase before stopping north-south traffic
- **East-West Green (EWG)**: East-west traffic flows while north-south traffic stops
- **East-West Yellow (EWY)**: Warning phase before stopping east-west traffic
- **All Red (AR)**: Safety clearance interval between conflicting movements

State Transition Table:

Current State	Timer Expired	Vehicle Detected	Emergency Signal	Next State
NSG	No	Don't Care	No	NSG
NSG	Yes	EW Direction	No	NSY
NSG	Don't Care	Don't Care	Yes	NSY
NSY	Yes	Don't Care	No	AR
AR	Yes	Don't Care	No	EWG
EWG	No	Don't Care	No	EWG
EWG	Yes	NS Direction	No	EWY
EWY	Yes	Don't Care	No	AR

Advanced Features: Modern implementations include adaptive timing based on traffic density, pedestrian crossing integration, and emergency vehicle preemption. The FSM can dynamically adjust green light duration based on queue length detected by sensors, improving traffic flow efficiency.

Implementation: These systems typically use programmable logic controllers (PLCs) or dedicated microcontrollers running FSM algorithms. The state machine processes inputs from various sensors and timing circuits, determining appropriate output signals for traffic lights, pedestrian signals, and system status indicators.

Benefits: FSM-based traffic controllers provide predictable, safe operation with clear state transitions that can be easily analyzed and modified. They ensure proper timing sequences, prevent conflicting signals, and can be easily expanded to handle complex intersection geometries or special traffic patterns.

This application demonstrates how FSMs provide robust, reliable control for critical infrastructure systems where safety and predictability are paramount concerns.