Moore Machine	Mealy Machine
Output depends only upon the present state.	Output depends on the present state as well as present input.
Moore machine also places its output on the transition.	Mealy Machine places its output on the transition.
More states are required.	Less number of states are required.
There is less hardware requirement for circuit implementation.	There is more hardware requirement for circuit implementation.
They react slower to inputs(One clock cycle later).	They react faster to inputs.
Synchronous output and state generation.	Asynchronous output generation.
Output is placed on states.	Output is placed on transitions.
Easy to design.	It is difficult to design.

DFA/NFA

1. Q: finite set of states
2. ∑: finite set of the input symbol
3. q0: initial state
4. F: **final** state
5. δ: Transition function

moore/mealy

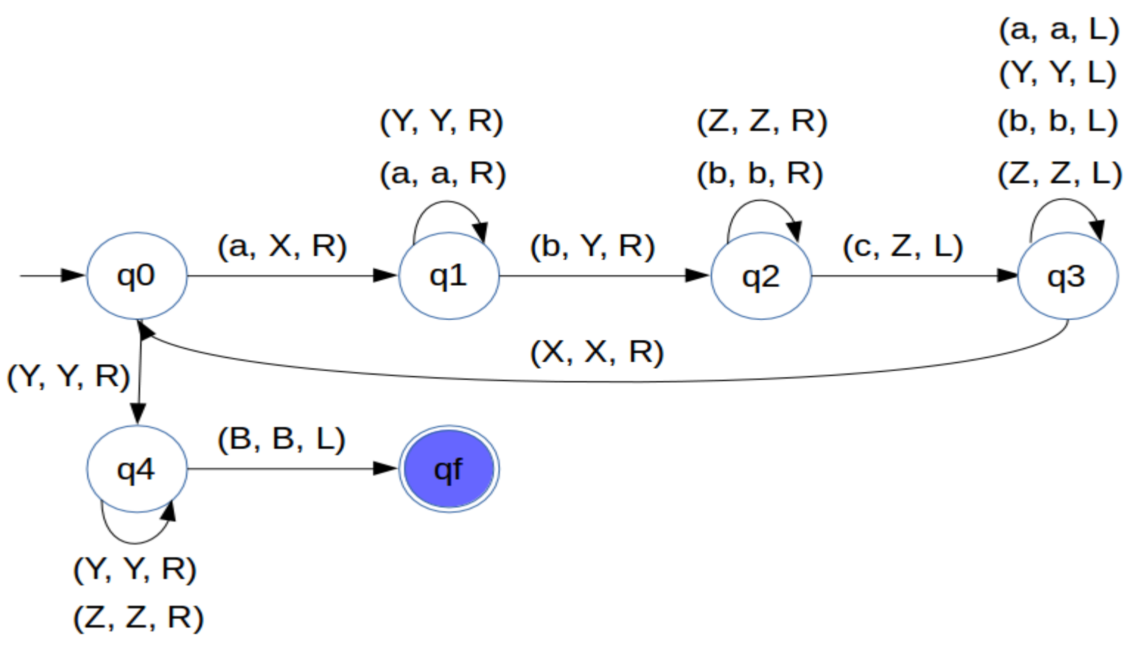
1. Q is a finite set of states
2. q0 is the initial state
3. ∑ is the input alphabet
4. ▲ is the output alphabet
5. δ is the transition function that maps Q×∑ → Q
6. λ is the output function that maps Q → ▲

A TM is expressed as a 7-tuple (Q, T, B, ∑, δ, q0, F) where:

- **Q** is a finite set of states
- **T** is the tape alphabet (symbols which can be written on Tape)
- **B** is blank symbol (every cell is filled with B except input alphabet initially)
- ∑ is the input alphabet (symbols which are part of input alphabet)
- **δ** is a transition function which maps Q × T → Q × T × {L,R}. Depending on its present state and present tape alphabet (pointed by head pointer), it will move to new state, change the tape symbol (may or may not) and move head pointer to either left or right.
- **q0** is the initial state
- **F** is the set of final states. If any state of F is reached, input string is accepted.

DFA	**NFA**
DFA stands for Deterministic Finite Automata.	NFA stands for Nondeterministic Finite Automata.
For each symbolic representation of the alphabet, there is only one state transition in DFA.	No need to specify how does the NFA react according to some symbol.
DFA cannot use Empty String transition.	NFA can use Empty String transition.
DFA can be understood as one machine.	NFA can be understood as multiple little machines computing at the same time.
In DFA, the next possible state is distinctly set.	In NFA, each pair of state and input symbol can have many possible next states.
DFA is more difficult to construct.	NFA is easier to construct.

Turing machine a^nb^nc^n



1.	B	a	a	a	b	b	b	c	c	c	B
		↑									
2.	B	X	a	a	b	b	b	c	c	c	B
					↑						
3.	B	X	a	a	Y	b	b	c	c	c	B
								↑			
4.	B	X	a	a	Y	b	b	Z	c	c	B
			↑								
5.	B	X	X	a	Y	b	b	c	c	c	B
						↑					
6.	B	X	X	a	Y	Y	b	Z	c	c	B
									↑		
7.	B	X	X	a	Y	Y	b	Z	Z	c	B
				↑							
8.	B	X	X	X	Y	Y	b	Z	Z	c	B
							↑				
9.	B	X	X	X	Y	Y	Y	Z	Z	c	B
										↑	
10.	B	X	X	X	Y	Y	b	Z	Z	Z	B
	↑										