# CS 6041
# Theory of Computation

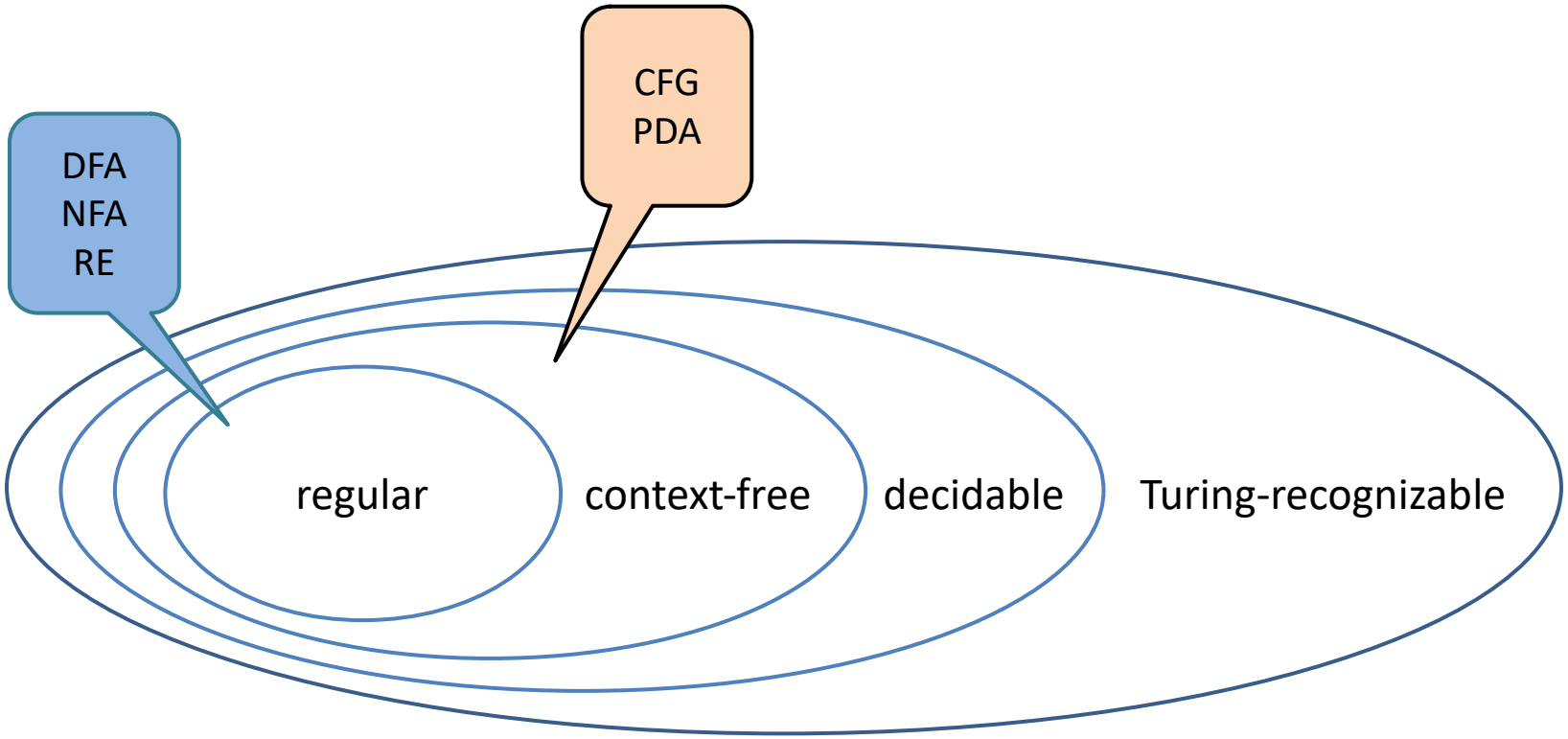
## Pushdown Automata

**Kun Suo**

Computer Science, Kennesaw State University

https://kevinsuo.github.io/

# Pushdown Automata (PDA)



# Pushdown Automata (PDA)

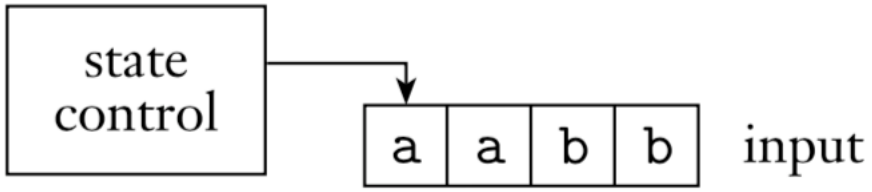
- Pushdown automatas are equivalent in power to context-free grammars (PDA=CFG)

- PDA can recognize some nonregular languages

# What does PDA look like?

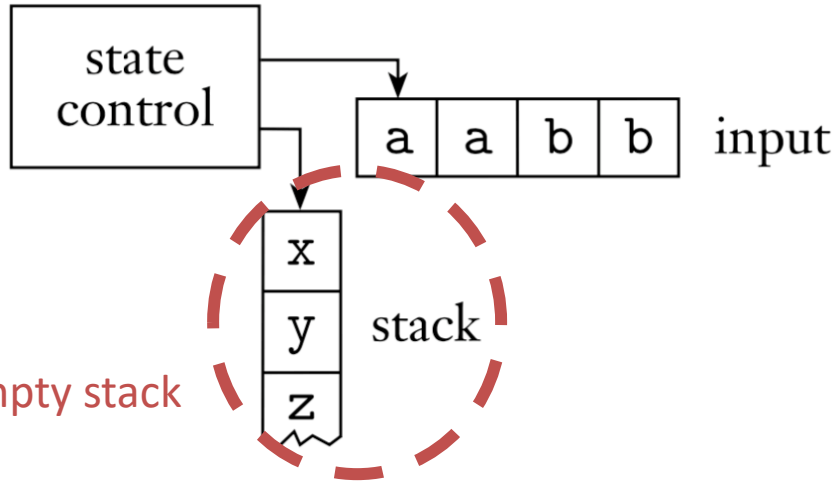
finite automaton

Memory = 1




pushdown automaton

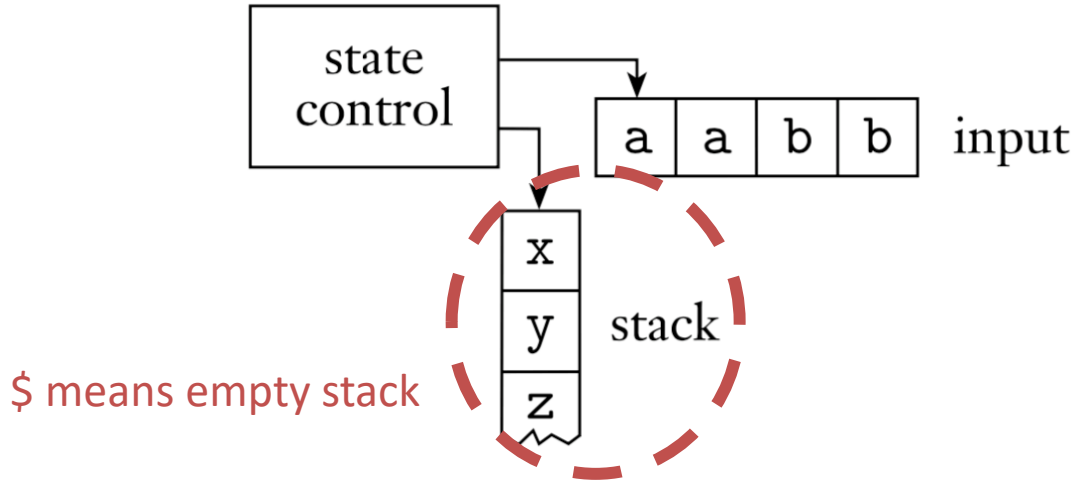
Memory = N




$ means empty stack

# What does PDA looks like?

pushdown automaton

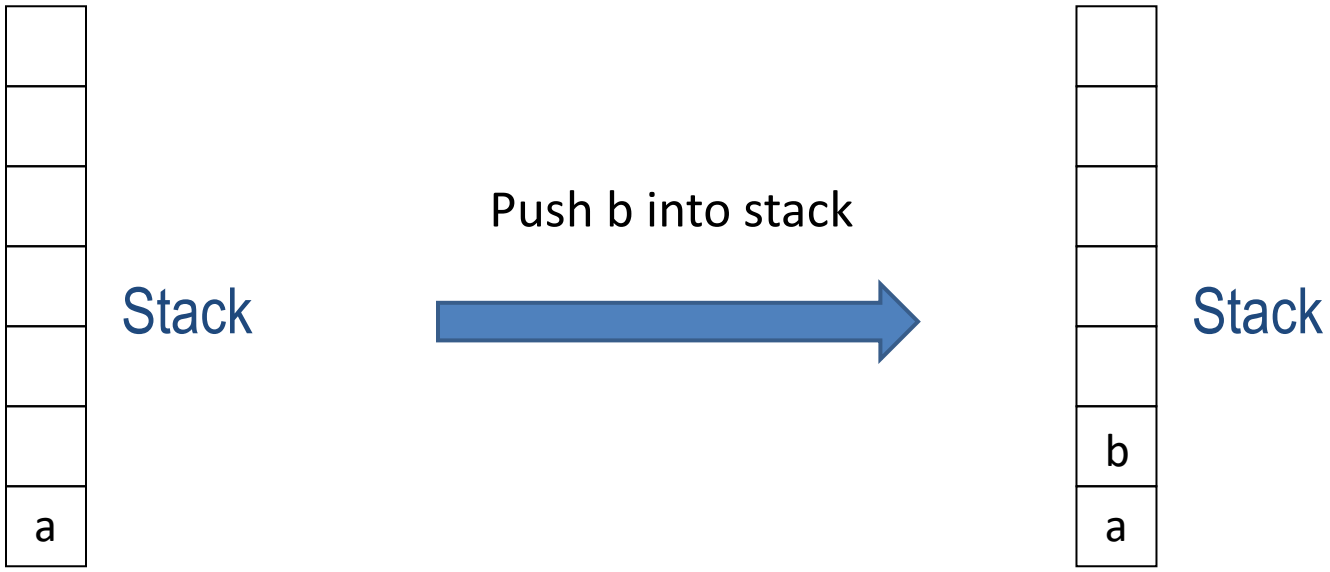



- Pushdown automata has more memories than finite automata

- PDA = finite automata + **A stack (unlimited size)**

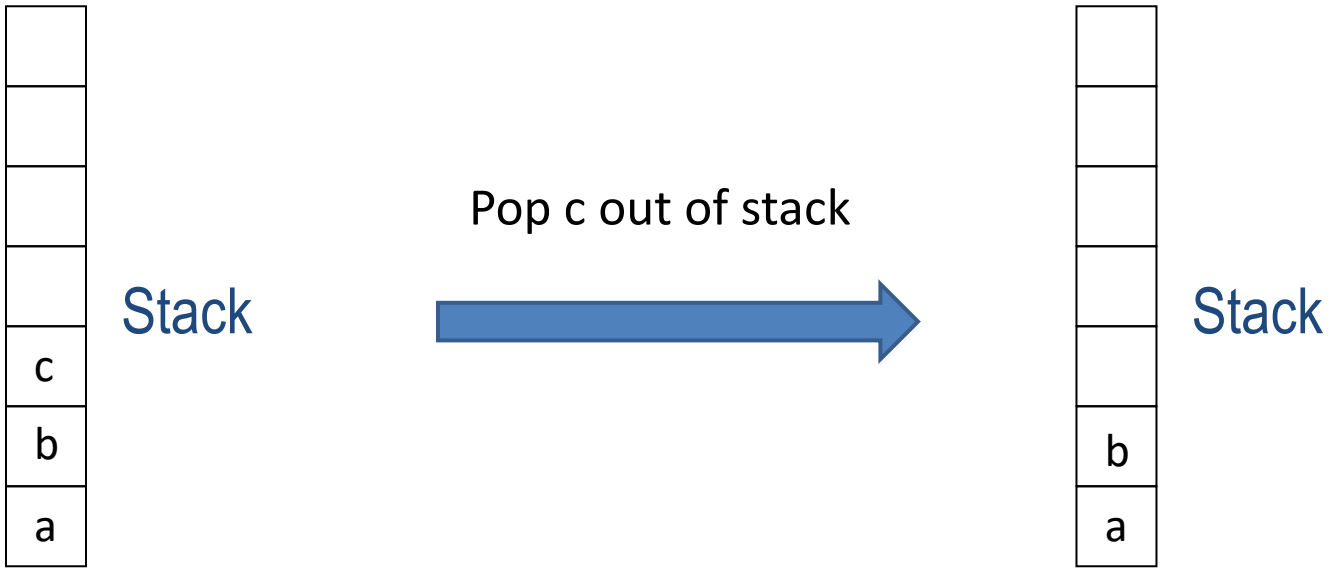
# Stack operation

- Push: add to the top of stack

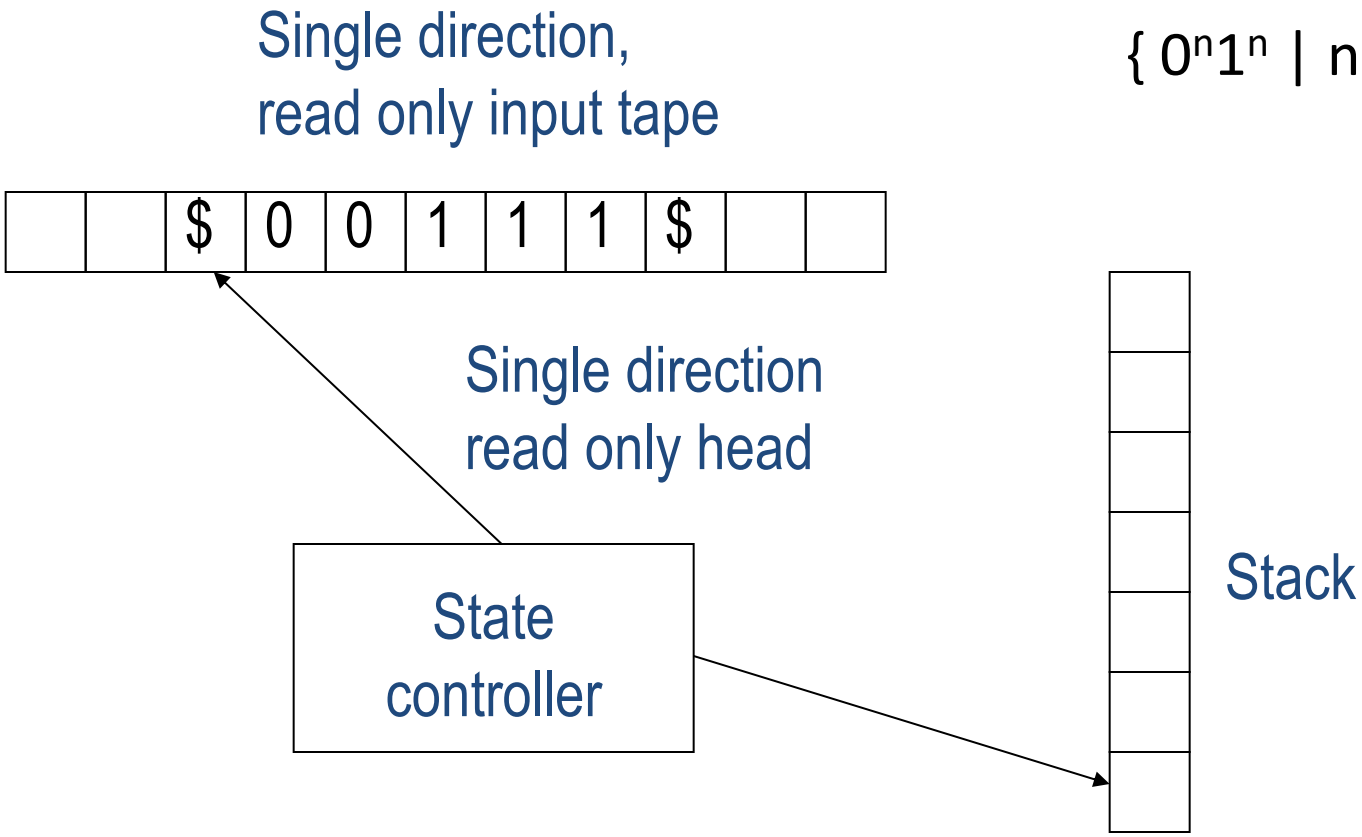


# Stack operation

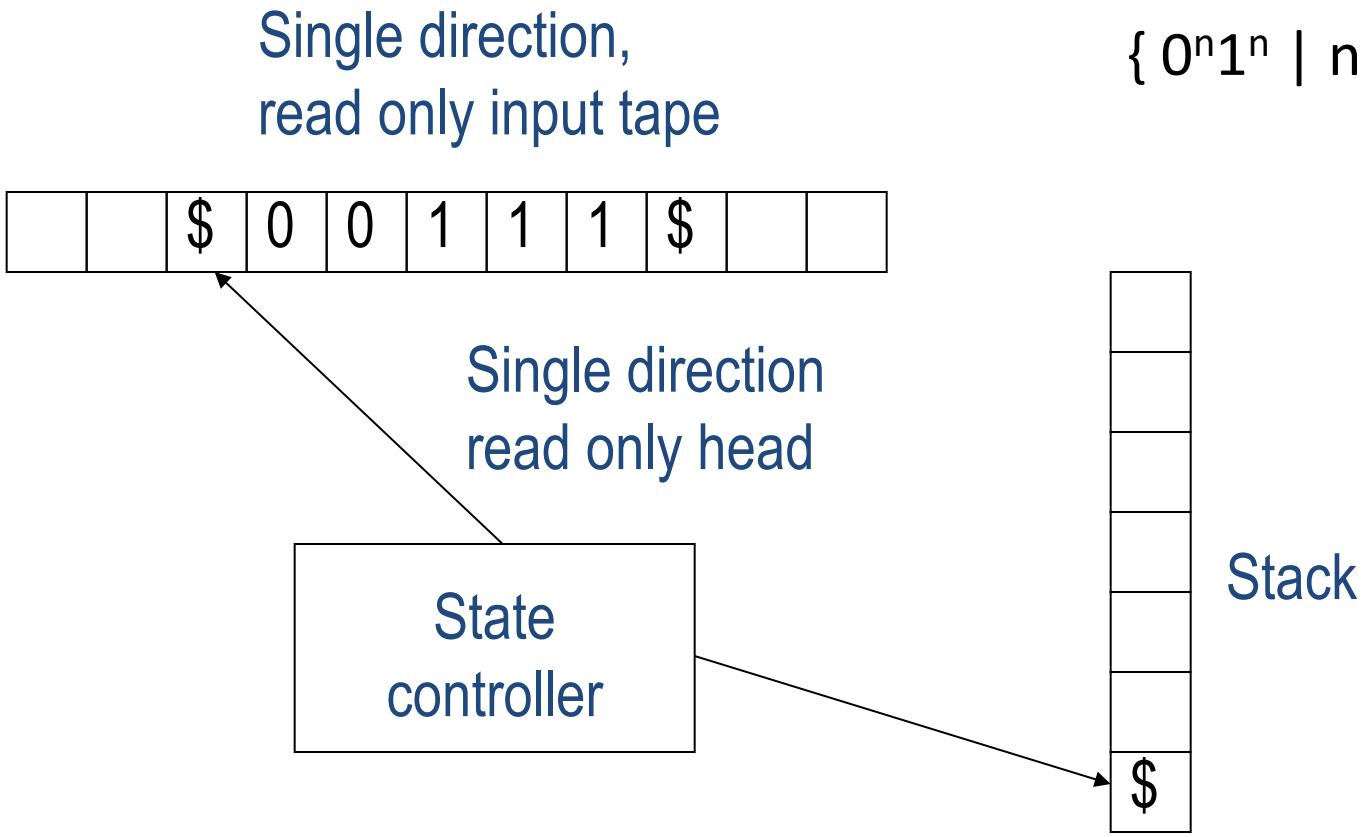
- Pop: remove from the top of stack



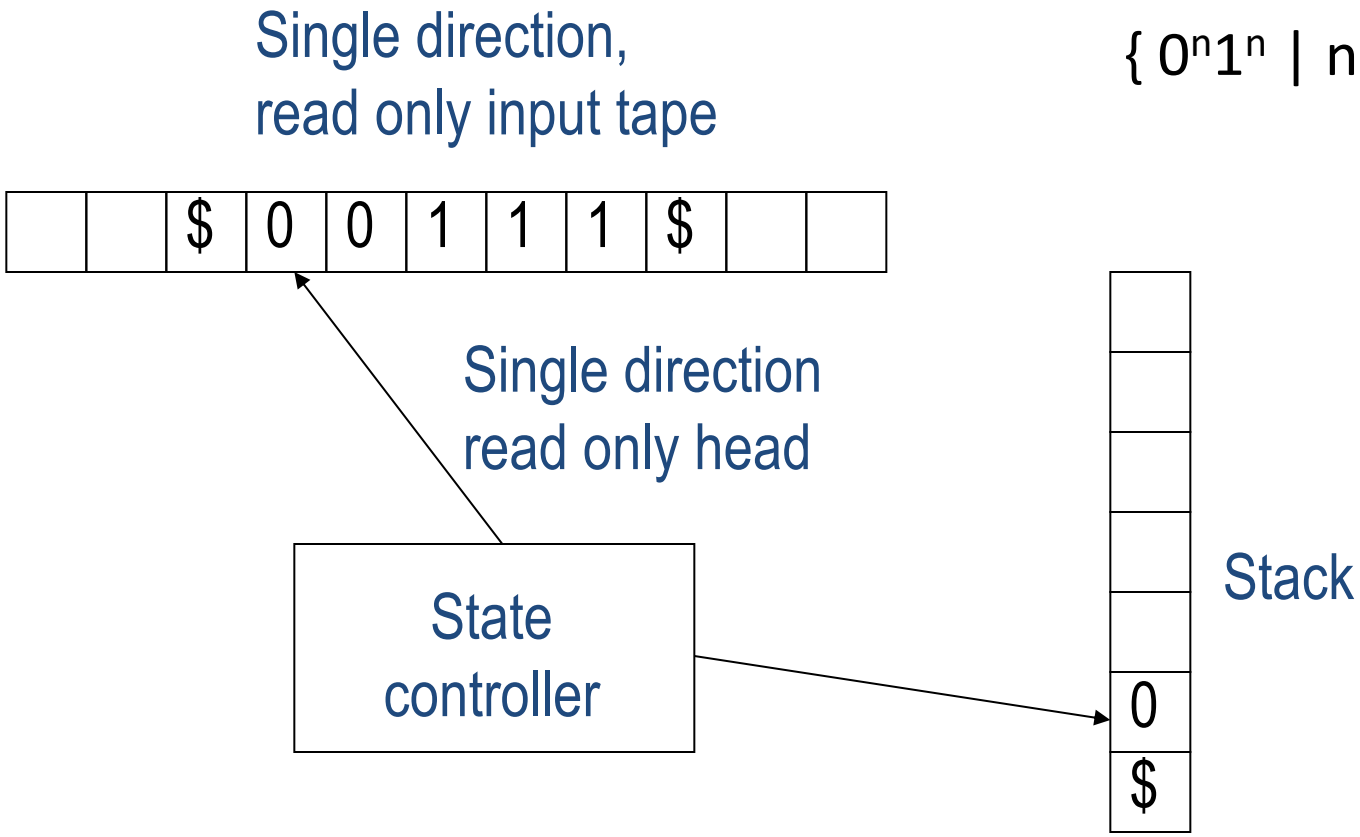
# To test whether 00111 is in $0^n1^n$



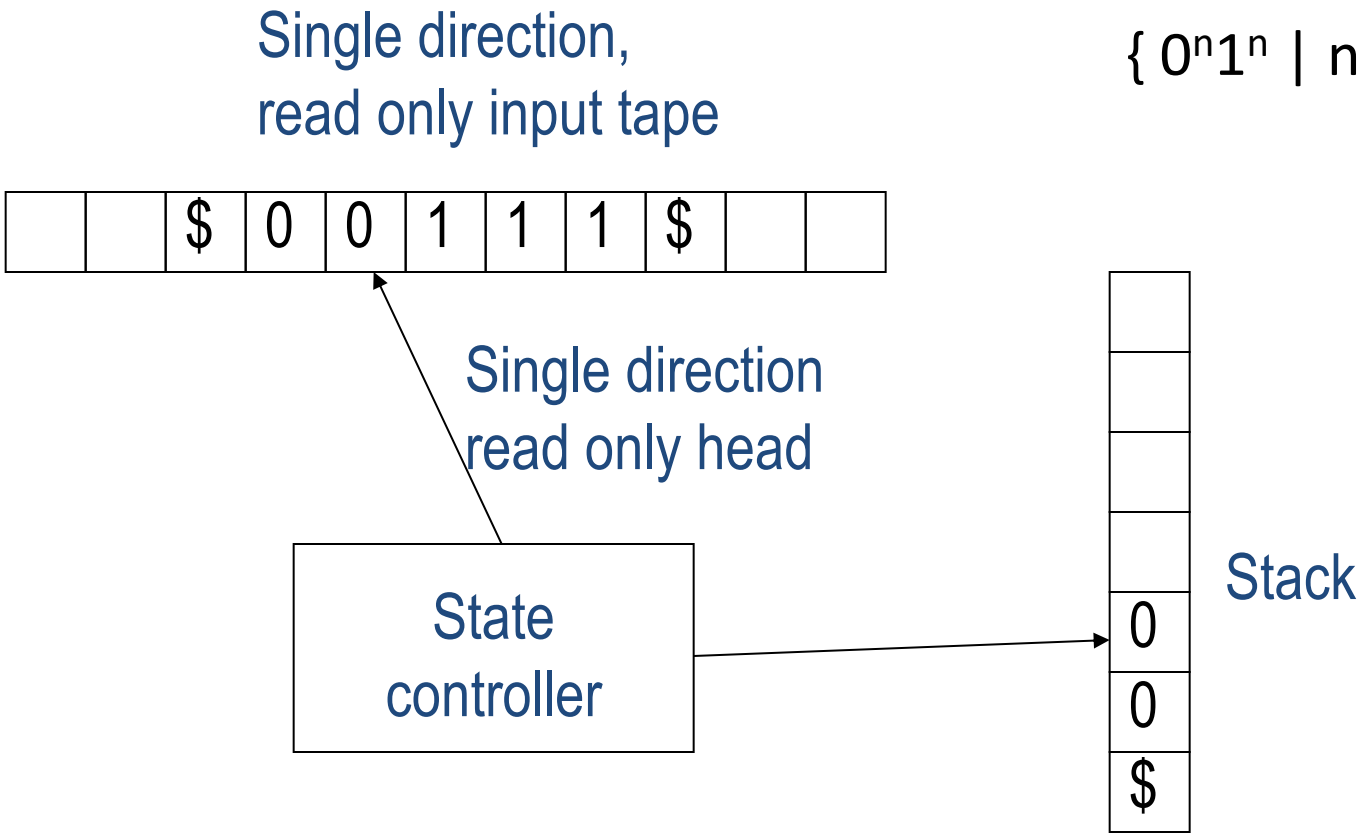
# To test whether 00111 is in $0^n1^n$



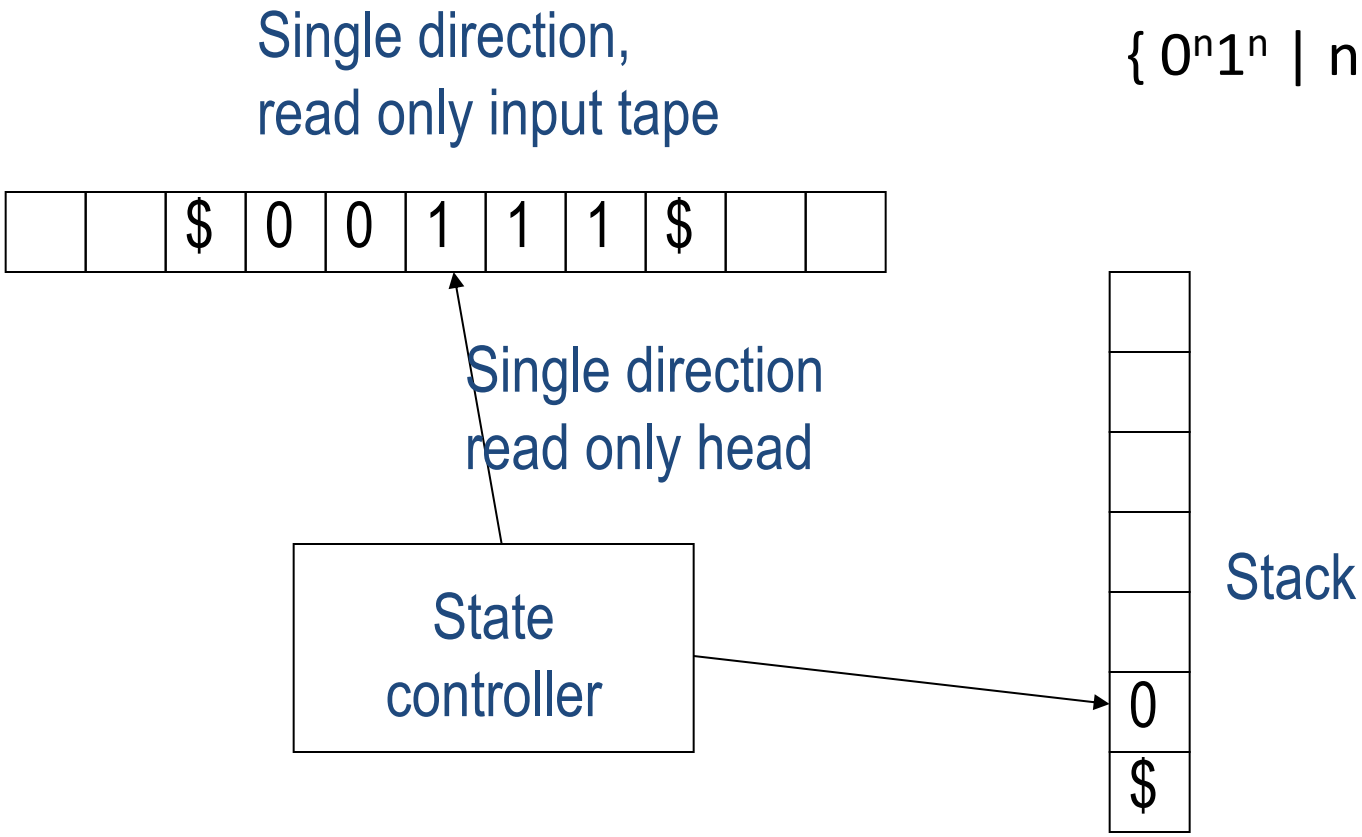
# To test whether 00111 is in $0^n1^n$



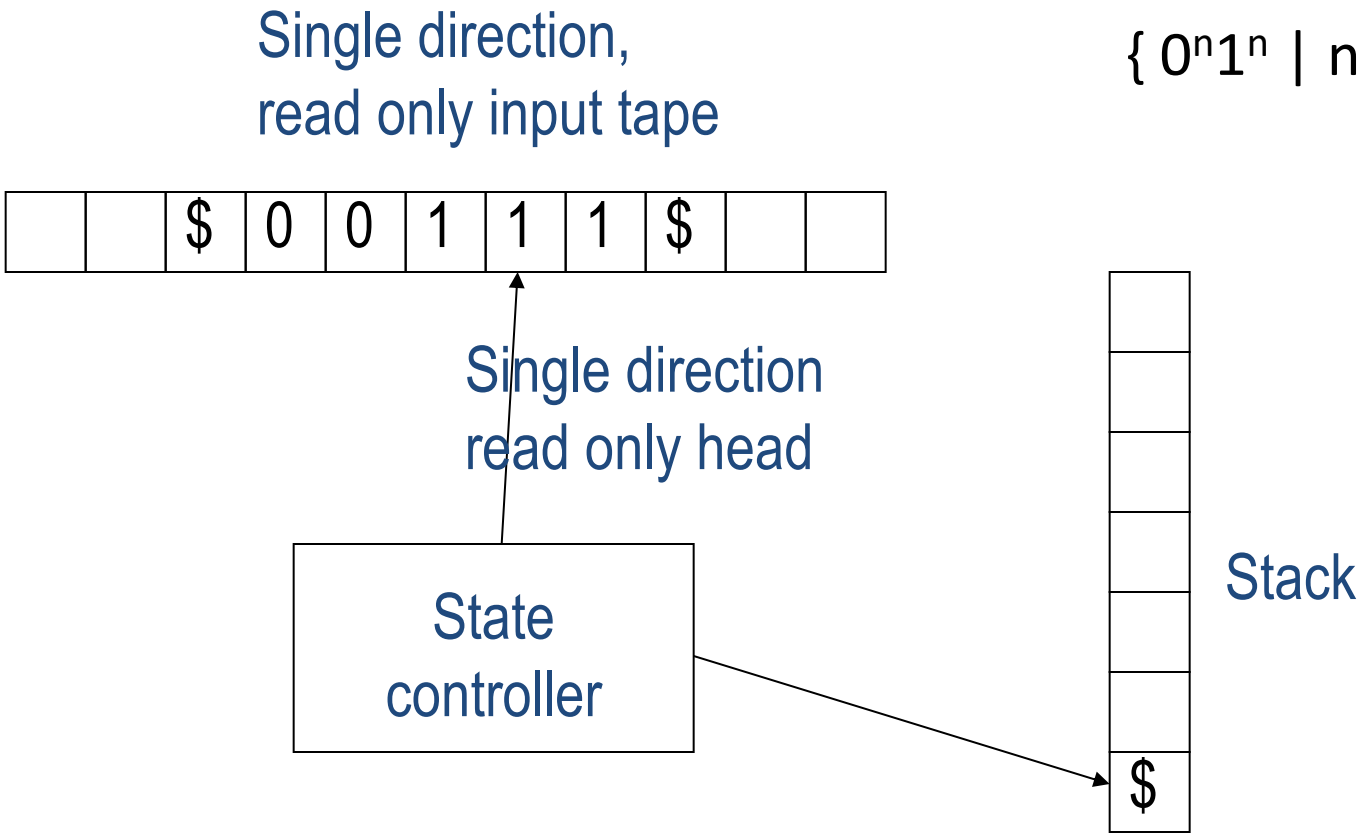
# To test whether 00111 is in $0^n1^n$



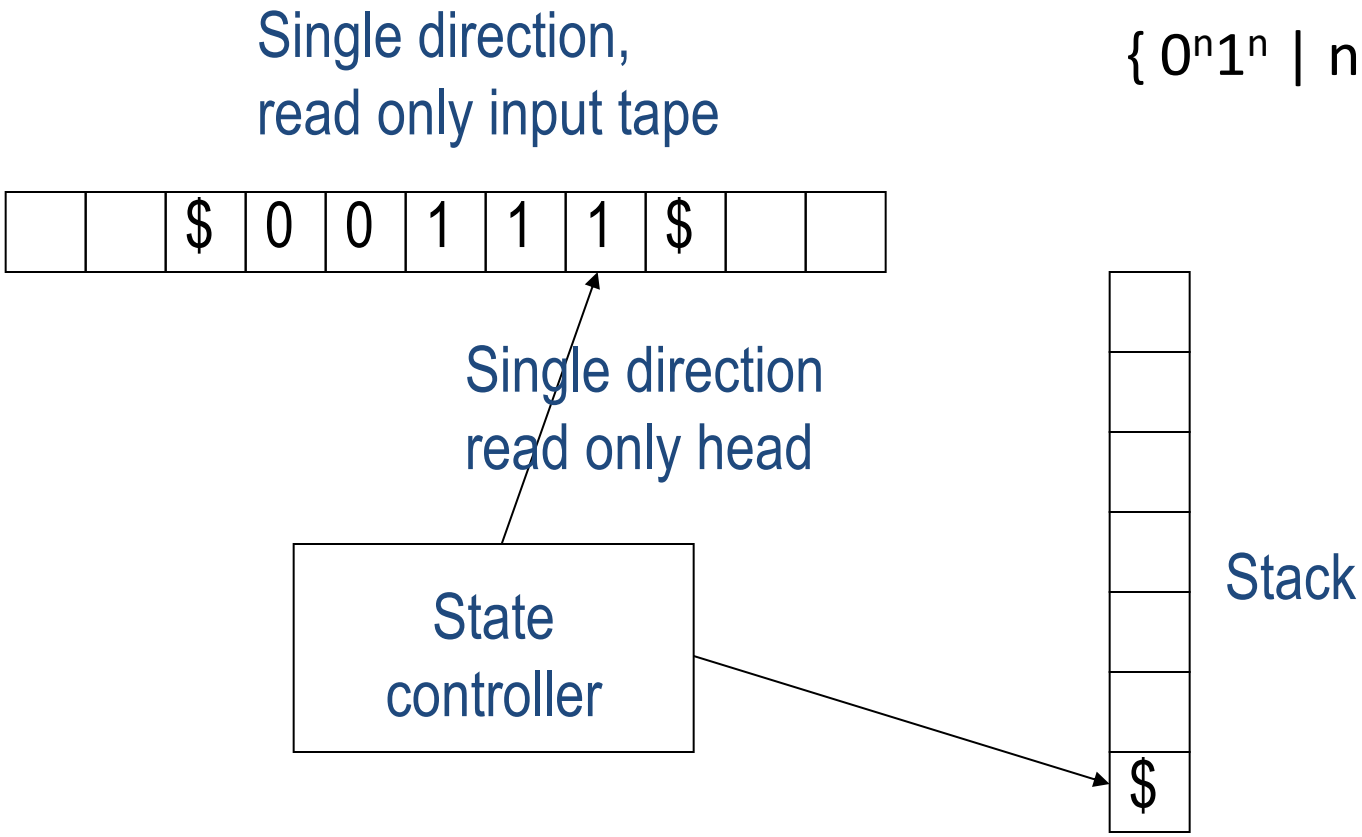
# To test whether 00111 is in $0^n1^n$



# To test whether 00111 is in $0^n1^n$

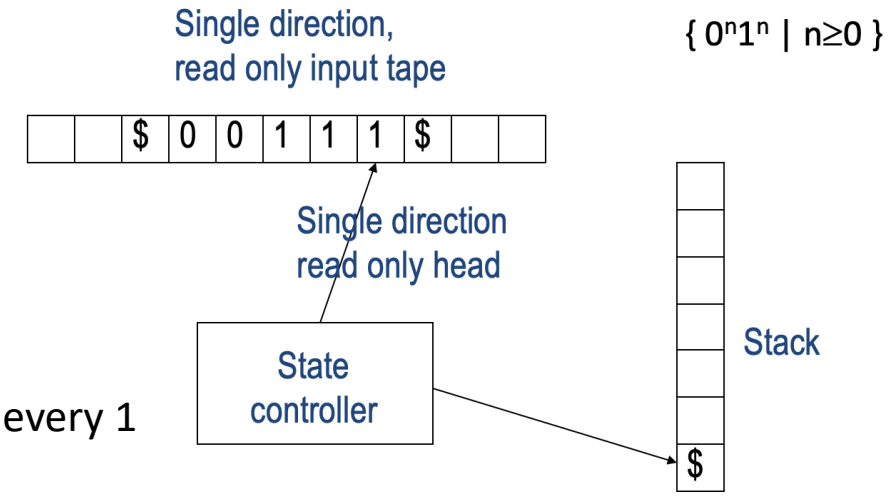


# To test whether 00111 is in $0^n1^n$



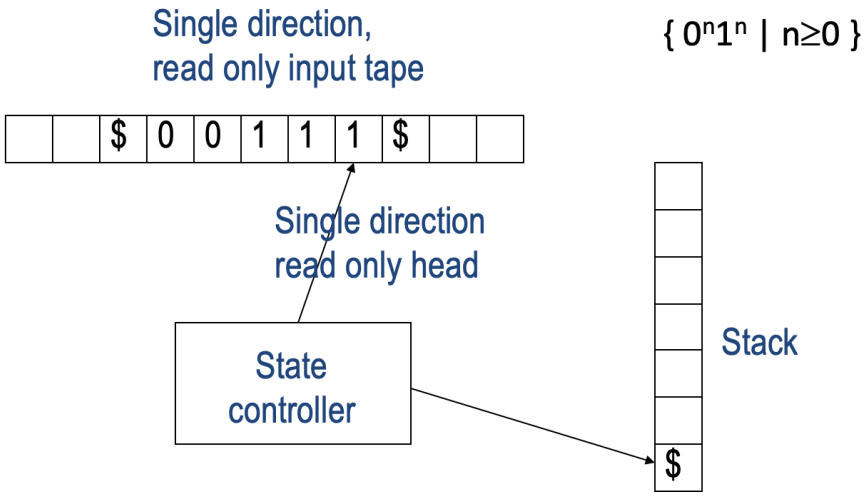
# Informal description for PDA to recognize some languages

- $A = \{0^n1^n \mid n \geq 0\}$
- Read symbols from input
  - Operation
    - For every 0s, push 0 into stack
    - When read 1s, pop one 0 from stack for every 1
  - Determine accept/reject:
    - When finish reading string and there is no 0s in stack, accept;
    - When there exist 0s after 1s, reject.
    - When tape is not finished while the stack is empty, reject;
    - When tape finished while the stack is non-empty, reject;



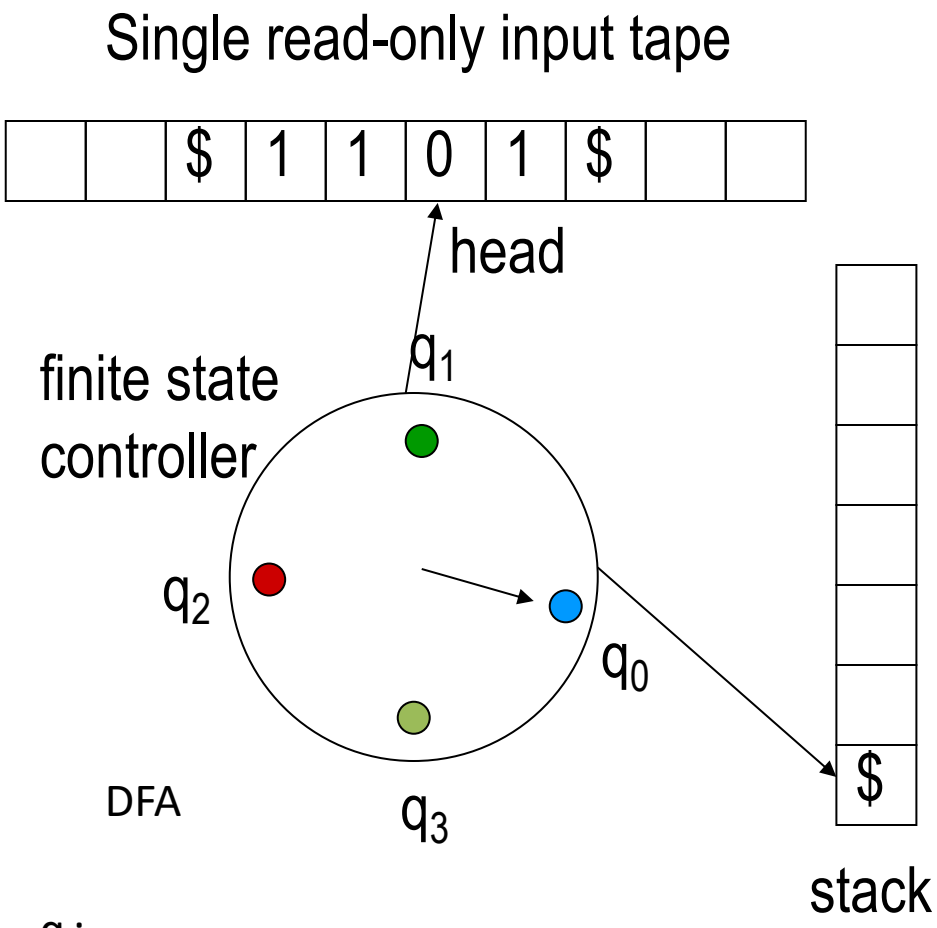
# Informal description for PDA to recognize some languages

- L = {w | w has some features}
- Read symbols from input
  - **STEP1:** If the language is regular, do not need to use stack; if not regul define operations on stack
  - **STEP2:** define operations:
    - When to push
    - When to pop
  - **STEP 3:** determine accept/reject:
    - Under which cases, accept
    - Under which cases, reject



# Question: Informal description

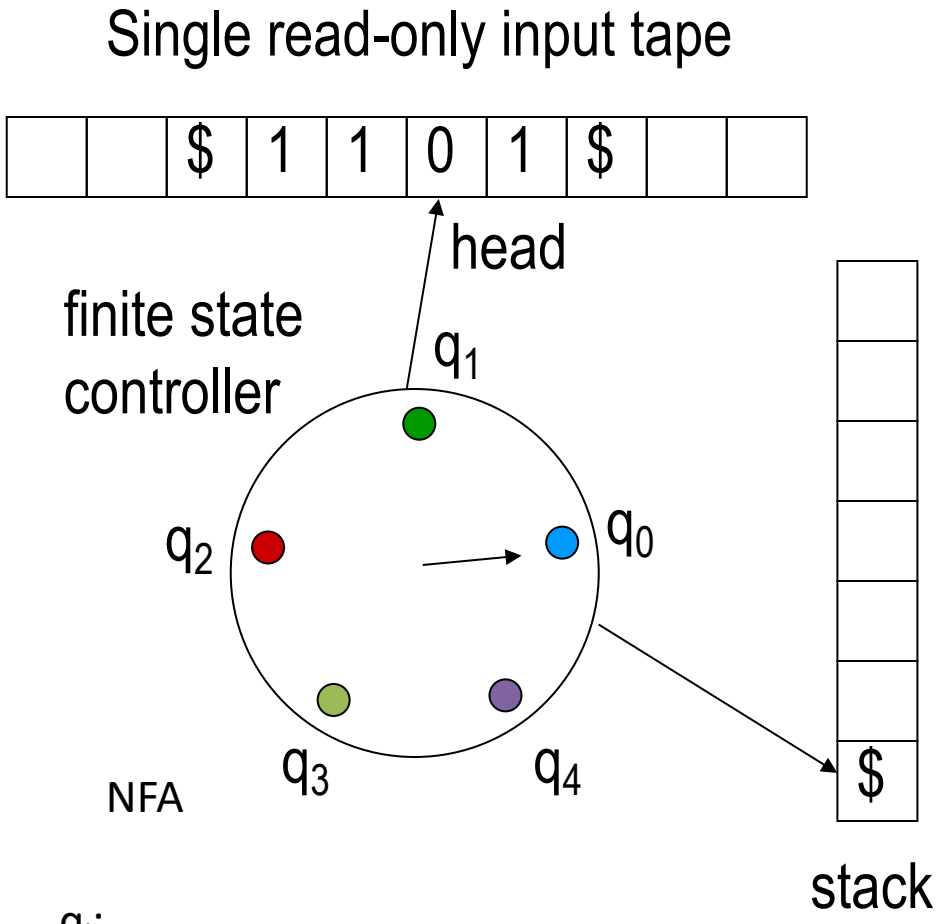
- L1={w| w has at least three 1s}
  - This set is regular, so the PDA doesn't even need to use its stack.
  - The PDA scans the string and uses its finite control to maintain a counter which counts up to 3. The PDA accepts the moment it sees three ones.




$q_i$:
No 1s; one 1; two 1s; three and more 1s;

# Question: Informal description

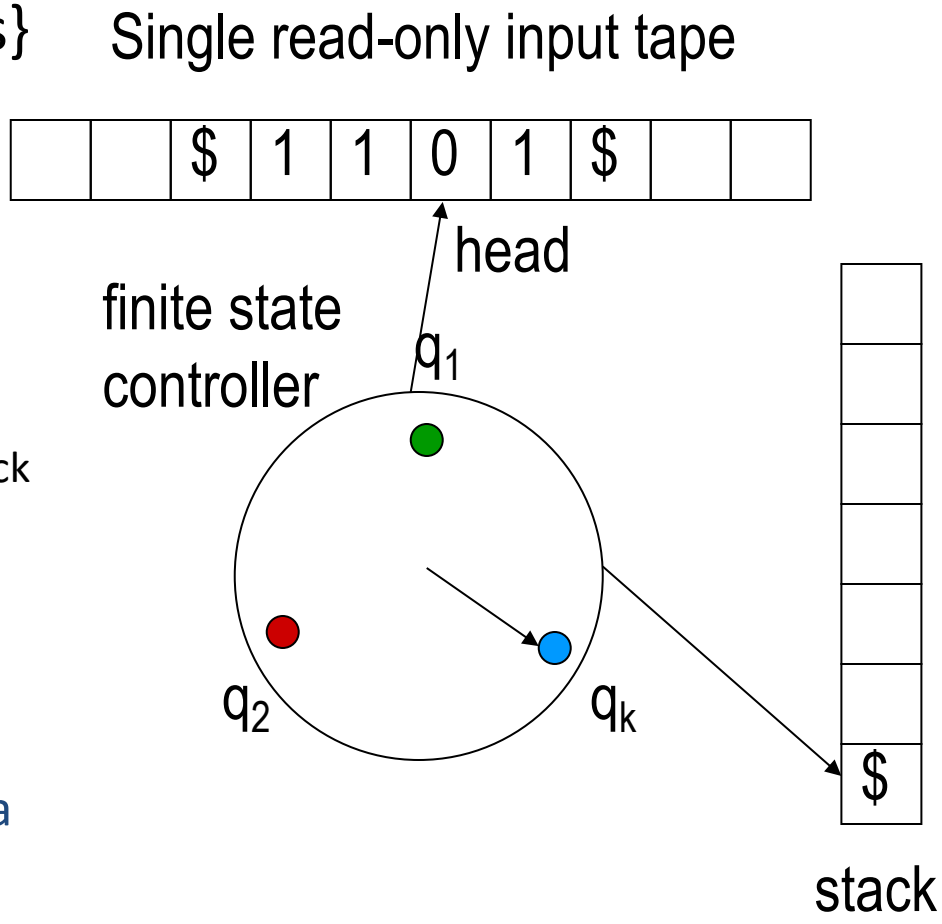
- L2={w| w starts and ends with the same symbol}
  - This set is regular, so the PDA doesn't even need to use its stack.
  - The PDA scans the string and keep track of the first and last symbol in its finite control. If they are the same, accepts.




$q_i$:
$\varepsilon$; 1***0; 0***1; 1***1; 0***0

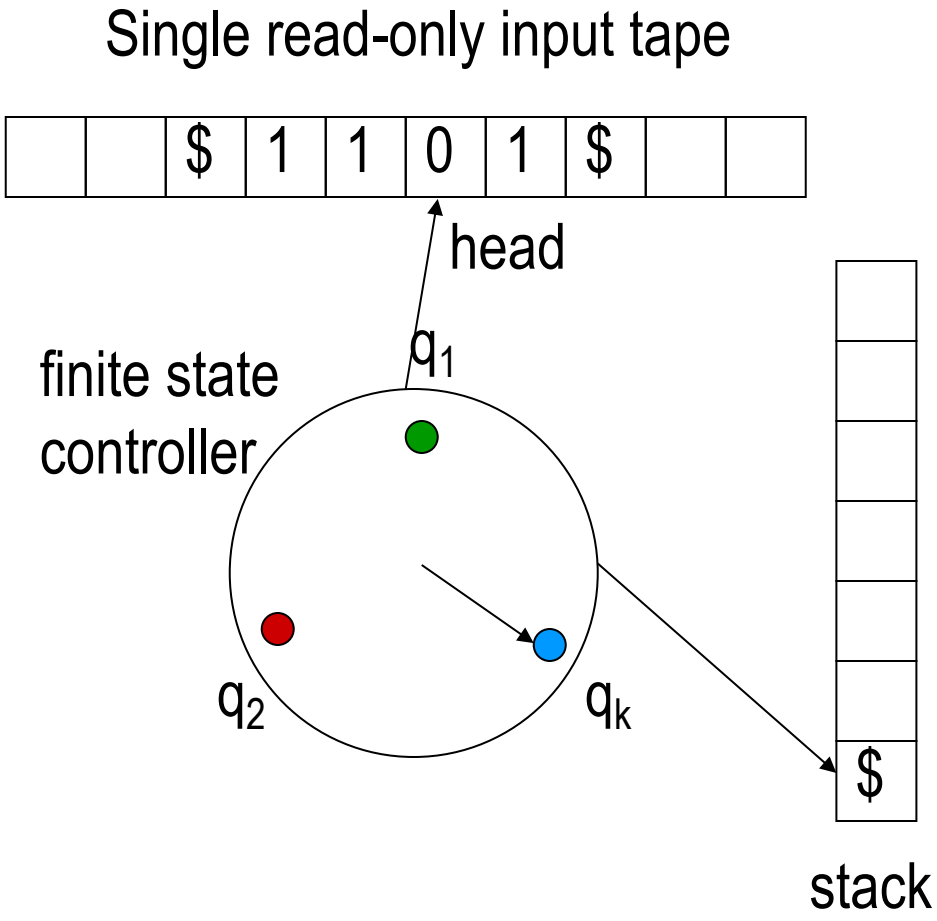
# Question: Informal description

- L3={w| w has more 1s than 0s}
  - This set is not regular.
  - The PDA scans across the input.
    - POP: If it sees a 1 and its top stack symbol is a 0, it pops the stack. Similarly, if it scans a 0 and its top stack symbol is a 1, it pops the stack.
    - PUSH: In all other cases, it pushes the input symbol onto the stack.
  - After it scans the input, if there is a 1 on top of the stack, it accepts. Otherwise it rejects.



# Question: Informal description

- L4=Ø
  - Just reject.



# Definition of PDA (non-deterministic)

- PDA $M=(Q,\Sigma,\Gamma,\delta,q_0,F)$, where

  1) Q: set of states

  2) $\Sigma$: input alphabet, $\Sigma_\varepsilon=\Sigma\cup\{\varepsilon\}$

  3) $\Gamma$: stack alphabet, $\Gamma_\varepsilon=\Gamma\cup\{\varepsilon\}$

  4) $\delta$: $Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon\rightarrow P(Q\times\Gamma_\varepsilon)$,

     transition function

  5) $q_0\in Q$: start state

  6) $F\subseteq Q$: accept state set

# PDA vs. NFA

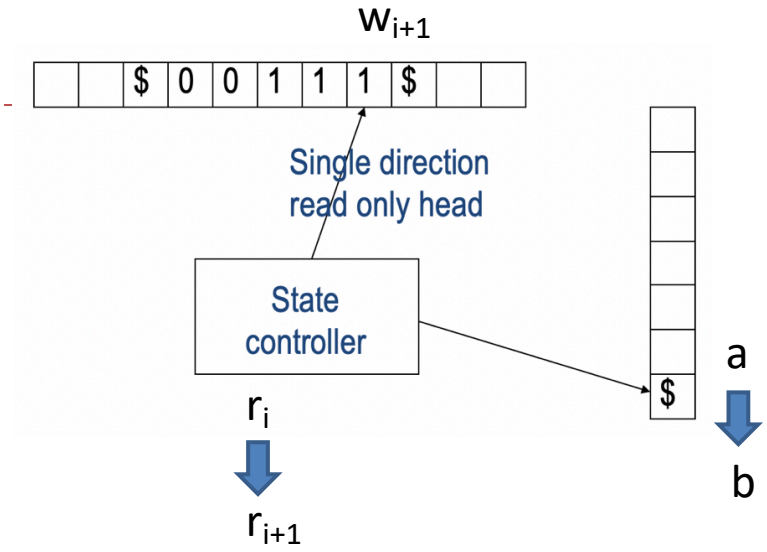
A ***pushdown automaton*** is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where $Q$, $\Sigma$, $\Gamma$, and $F$ are all finite sets, and

1. $Q$ is the set of states,
2. $\Sigma$ is the input alphabet,
3. $\Gamma$ is the stack alphabet,
4. $\delta: Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow \mathcal{P}(Q \times \Gamma_\varepsilon)$ is the transition function,
5. $q_0 \in Q$ is the start state, and
6. $F \subseteq Q$ is the set of accept states.

A ***nondeterministic finite automaton*** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. $Q$ is a finite set of states,
2. $\Sigma$ is a finite alphabet,
3. $\delta: Q \times \Sigma_\varepsilon \longrightarrow \mathcal{P}(Q)$ is the transition function,
4. $q_0 \in Q$ is the start state, and
5. $F \subseteq Q$ is the set of accept states.

# Computation on PDA




- M=($Q,\Sigma,\Gamma,\delta,q_0,F$);

  input $w=w_1w_2...w_m$,

  $w_i \in \Sigma_\varepsilon$

- Computation: (state, stack)

  $(r_0,s_0), (r_1,s_1) , \dots , (r_m ,s_m)$,

  Where $r_i \in Q$, $s_i \in \Gamma^*$, satisfying

  1) $(r_0,s_0)=(q_0,\varepsilon)$;

  At first, the first state is $q_0$ and stack is empty

  2) $(r_{i+1},b) \in \delta(r_i,w_{i+1},a)$;

  where $s_i=at$; $s_{i+1}=bt$,

  $a,b \in \Sigma_\varepsilon$,

  $t \in \Gamma^*$ (t are other elements in stack)

  After input $w_{i+1}$ , state changes from $r_i$ to $r_{i+1}$ and the top element in stack changes from a to b

# Computation on PDA

- Accept of computation:

  3) $r_m \in F$;

- M accepts w:

  M is at accept states after input of w

- The language that M accepts:

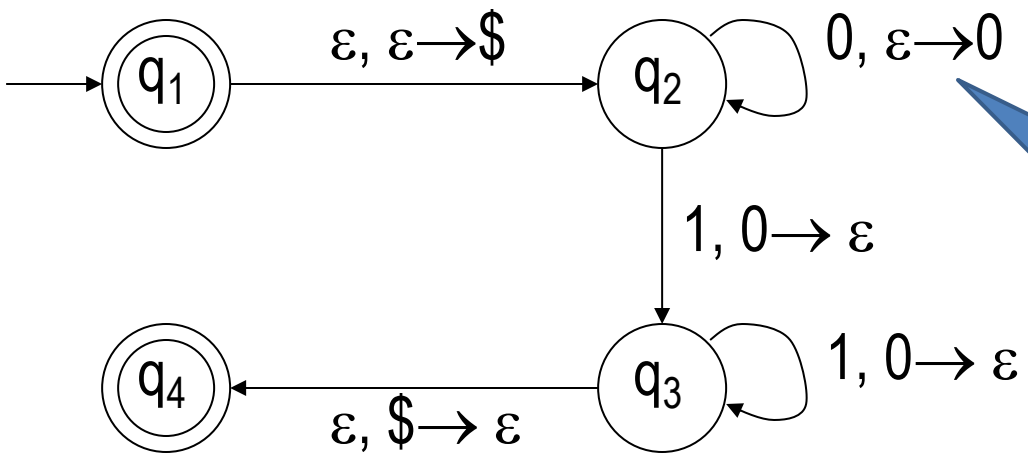
  L(M) = { x | M accepts x }

# PDA example

| $ | 000…0 | 111…1 | $ |
|---|---|---|---|
| $q_1$ | $q_2$ | $q_3$ | $q_4$ |

- $L = \{0^n1^n \mid n \geq 0\}$
- $M_1=(\{q_1,q_2,q_3,q_4\},\{0,1\},\{0,\$\},\delta,q_1,\{q_1,q_4\})$

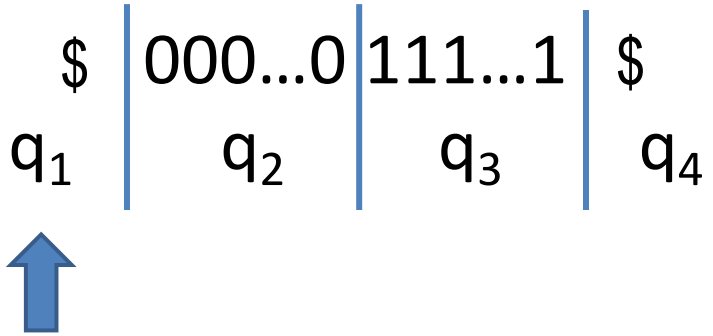
We only put 0 or $ into stack



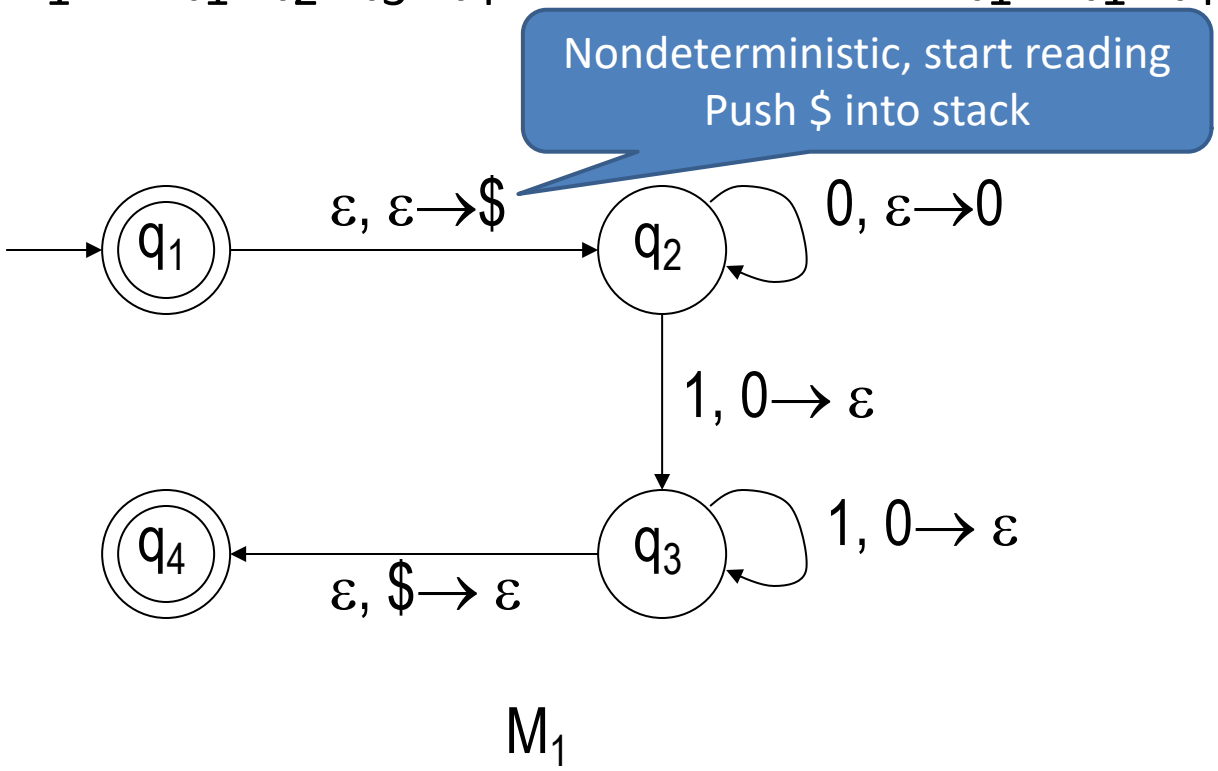

a,b->c
a: input
b->c: the top of stack changes

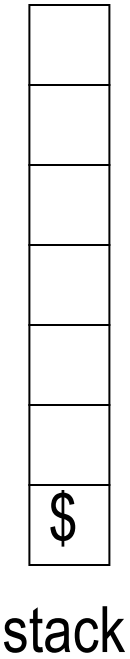
$M_1$

# PDA example

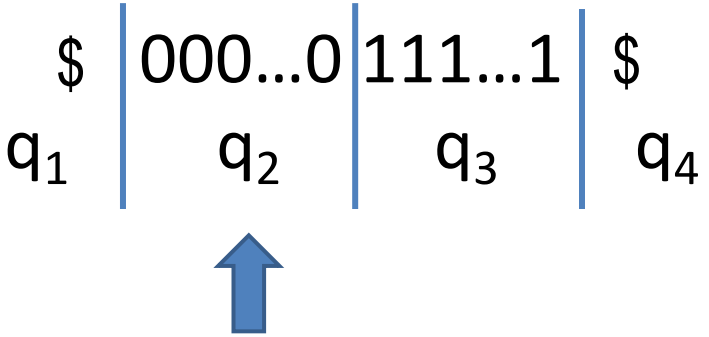



- $L = \{0^n1^n \mid n \geq 0\}$
- $M_1=(\{q_1,q_2,q_3,q_4\},\{0,1\},\{0,\$\},\delta,q_1,\{q_1,q_4\})$

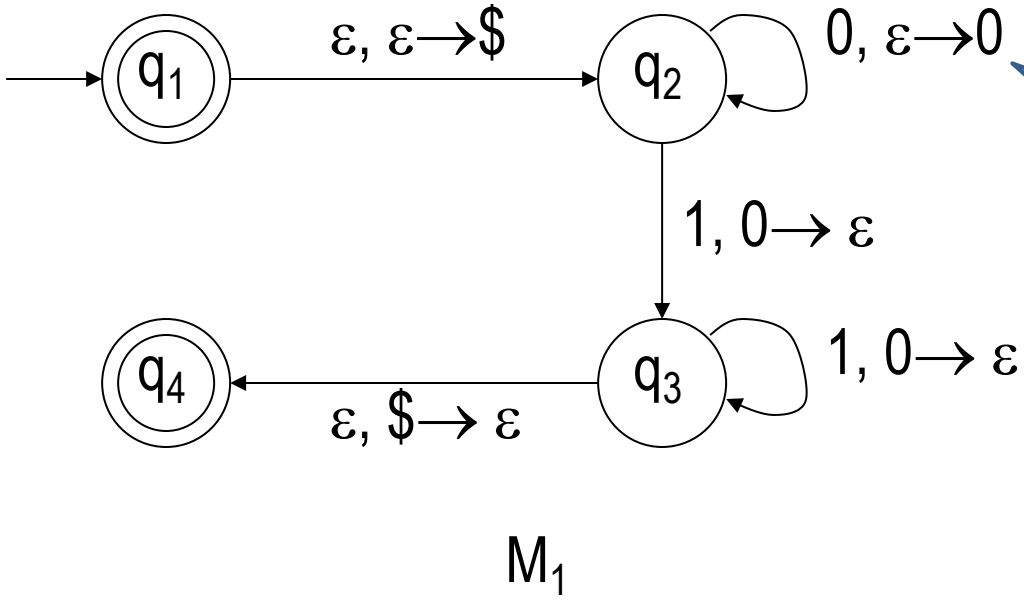


# PDA example

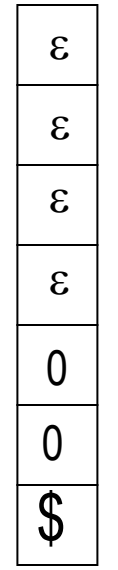



- $L = \{0^n1^n \mid n \geq 0\}$
- $M_1=(\{q_1,q_2,q_3,q_4\},\{0,1\},\{0,\$\},\delta,q_1,\{q_1,q_4\})$



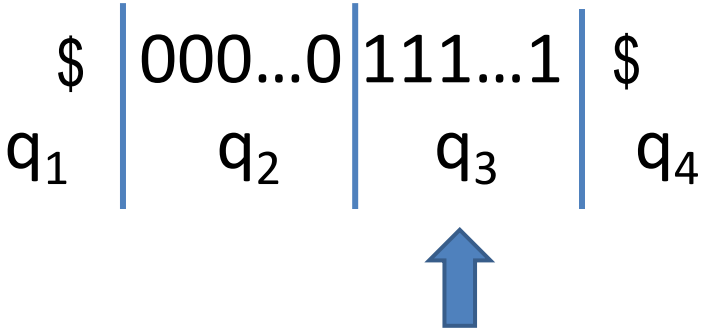

$M_1$

For input 0, push 0 into the stack

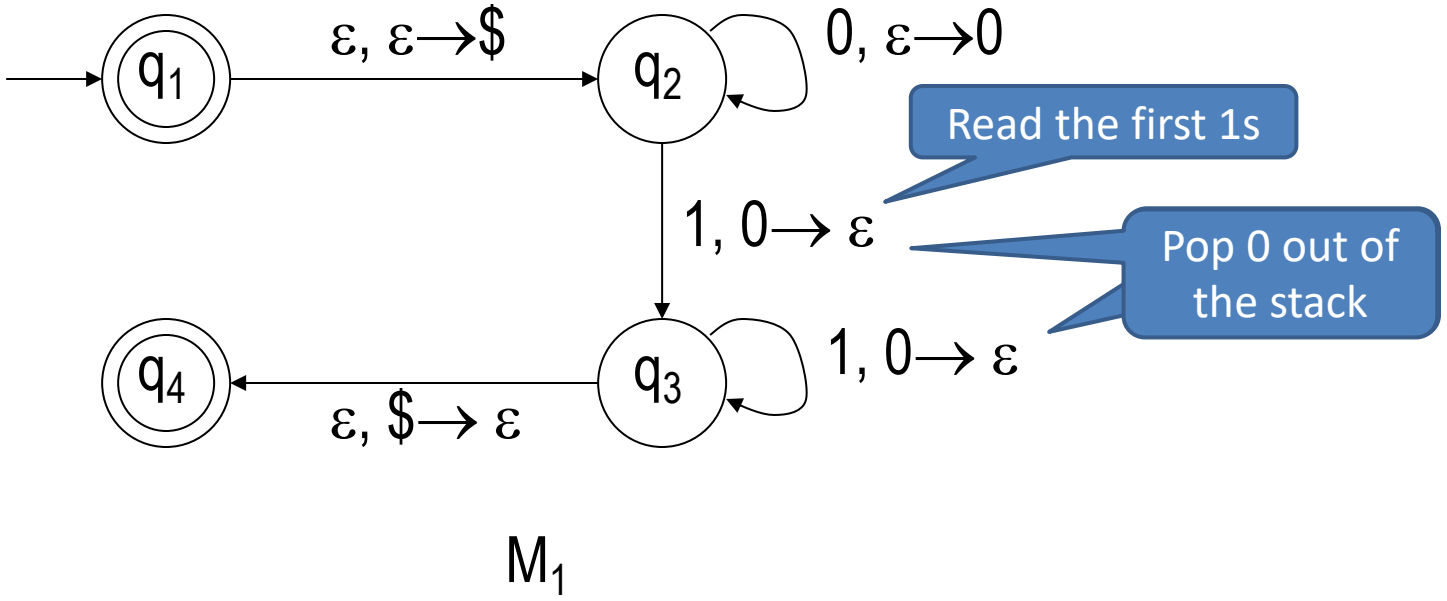



stack

# PDA example



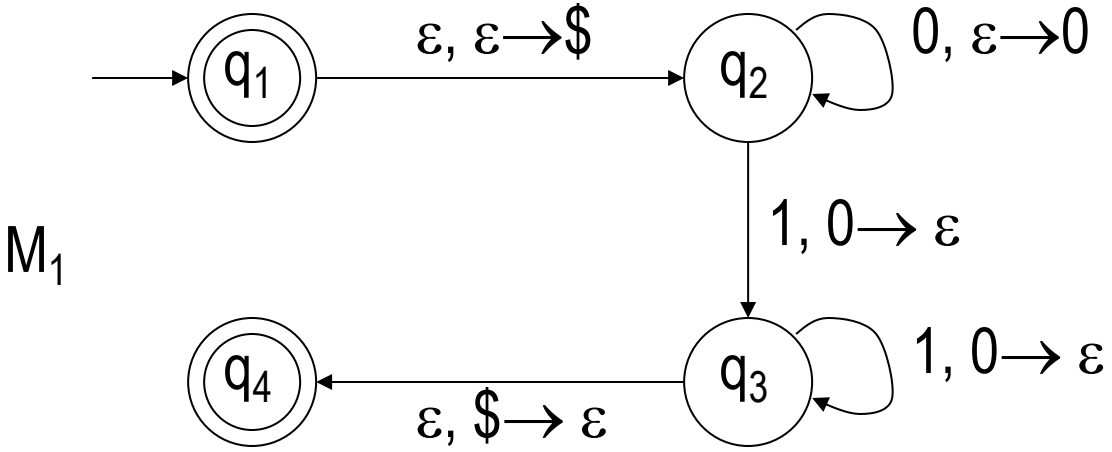

- $L = \{0^n1^n \mid n \geq 0\}$
- $M_1=(\{q_1,q_2,q_3,q_4\},\{0,1\},\{0,\$\},\delta,q_1,\{q_1,q_4\})$

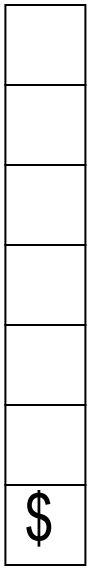


# PDA example

| $ | 000…0 | 111…1 | $ |
|---|---|---|---|
| $q_1$ | $q_2$ | $q_3$ | $q_4$ |

- $L = \{0^n1^n \mid n \geq 0\}$
- $M_1=(\{q_1,q_2,q_3,q_4\},\{0,1\},\{0,\$\},\delta,q_1,\{q_1,q_4\})$




If $ is popped out of stack, means stack is empty;
If input is ε, means string finishes. Accept

# PDA example

| $ | 000…0 | 111…1 | $ |
|---|---|---|---|
| $q_1$ | $q_2$ | $q_3$ | $q_4$ |

- $L = \{0^n1^n \mid n \geq 0\}$
- $M_1=(\{q_1,q_2,q_3,q_4\},\{0,1\},\{0,\$\},\delta,q_1,\{q_1,q_4\})$

If input is 0,
(q2, ε) changes to (q2, 0)
ε in stack change to 0 (PUSH 0)

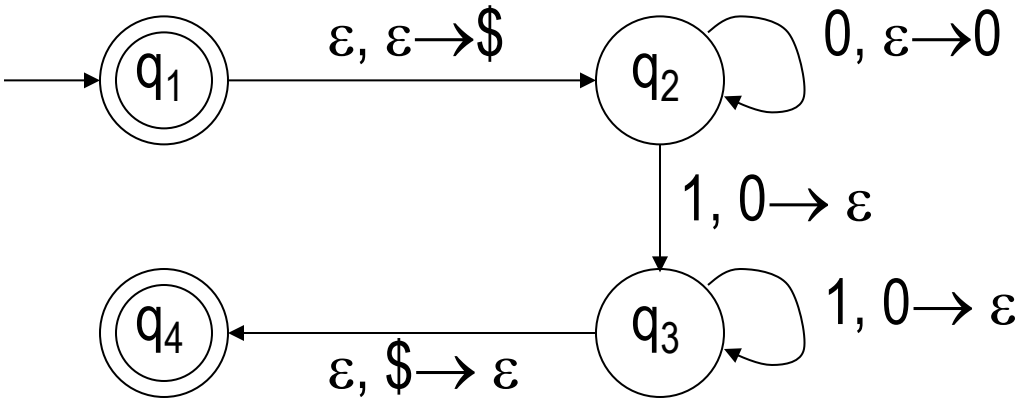
$\delta: Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon\rightarrow P(Q\times\Gamma_\varepsilon)$

$\delta$ table

| | Input | 0 | | | 1 | | | ε | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\Gamma_\varepsilon$ | stack | 0 | $ | ε | 0 | $ | ε | 0 | $ | ε |
| Q | state $q_1$ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | $\{(q_2,\$)\}$ |
| | state $q_2$ | ∅ | ∅ | $\{(q_2,0)\}$ | $\{(q_3,\varepsilon)\}$ | ∅ | ∅ | ∅ | ∅ | ∅ |
| | state $q_3$ | ∅ | ∅ | ∅ | $\{(q_3,\varepsilon)\}$ | ∅ | ∅ | ∅ | $\{(q_4,\varepsilon)\}$ | ∅ |
| | state $q_4$ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ |

# PDA example

δ graph




δ table

| Input | | 0 | | | 1 | | | ε | | |
|---|---|---|---|---|---|---|---|---|---|---|
| stack | | 0 | $ | ε | 0 | $ | ε | 0 | $ | ε |
| state | $q_1$ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | $\{(q_2,\$)\}$ |
| | $q_2$ | ∅ | ∅ | $\{(q_2,0)\}$ | $\{(q_3,\varepsilon)\}$ | ∅ | ∅ | ∅ | ∅ | ∅ |
| | $q_3$ | ∅ | ∅ | ∅ | $\{(q_3,\varepsilon)\}$ | ∅ | ∅ | ∅ | $\{(q_4,\varepsilon)\}$ | ∅ |
| | $q_4$ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ |

# Design PDA

| $ | aaa…a | bbb…b | ccc…c |
|---|---|---|---|
| $q_1$ | $q_2$ | $q_3$ | $q_4$ |

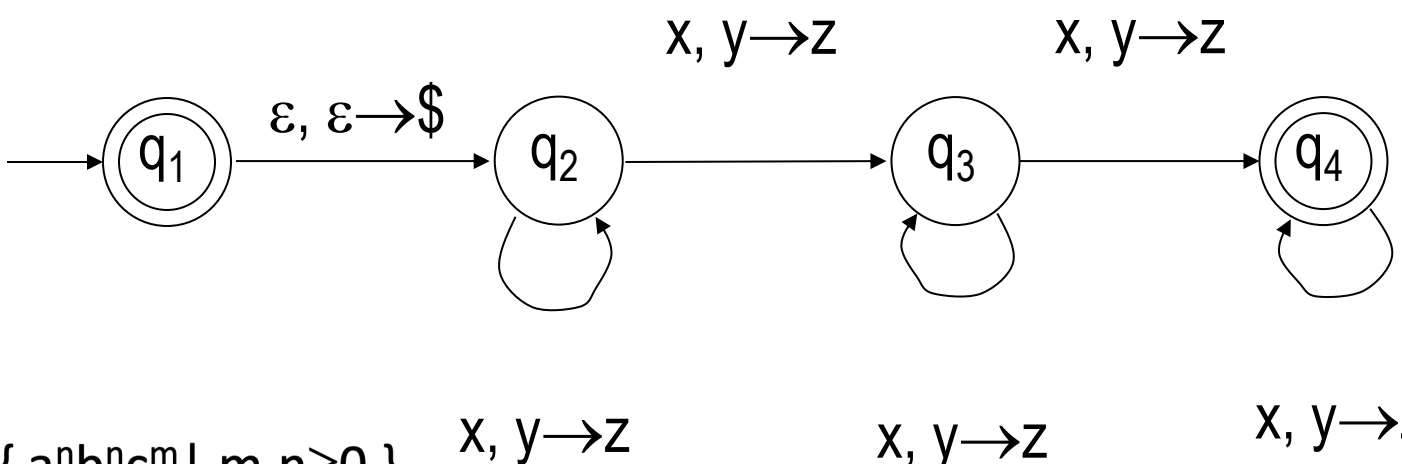
- $L(M_2)=\{ a^n b^n c^m \mid m,n \geq 0 \}$
  - Operation:
    - For an input a, and push a into stack
    - For an input b, pop one a from the stack
  - Determine accept/reject
    - If the stack is empty when finish reading b, then after reading all the cs, accept;
    - Otherwise, reject;

# Design PDA

| $ | aaa…a | bbb…b | ccc…c |
|---|---|---|---|
| $q_1$ | $q_2$ | $q_3$ | $q_4$ |

- $L(M_2)=\{ a^n b^n c^m \mid m,n \geq 0 \}$

x,y->z
x: input
y->z: the top of stack changes

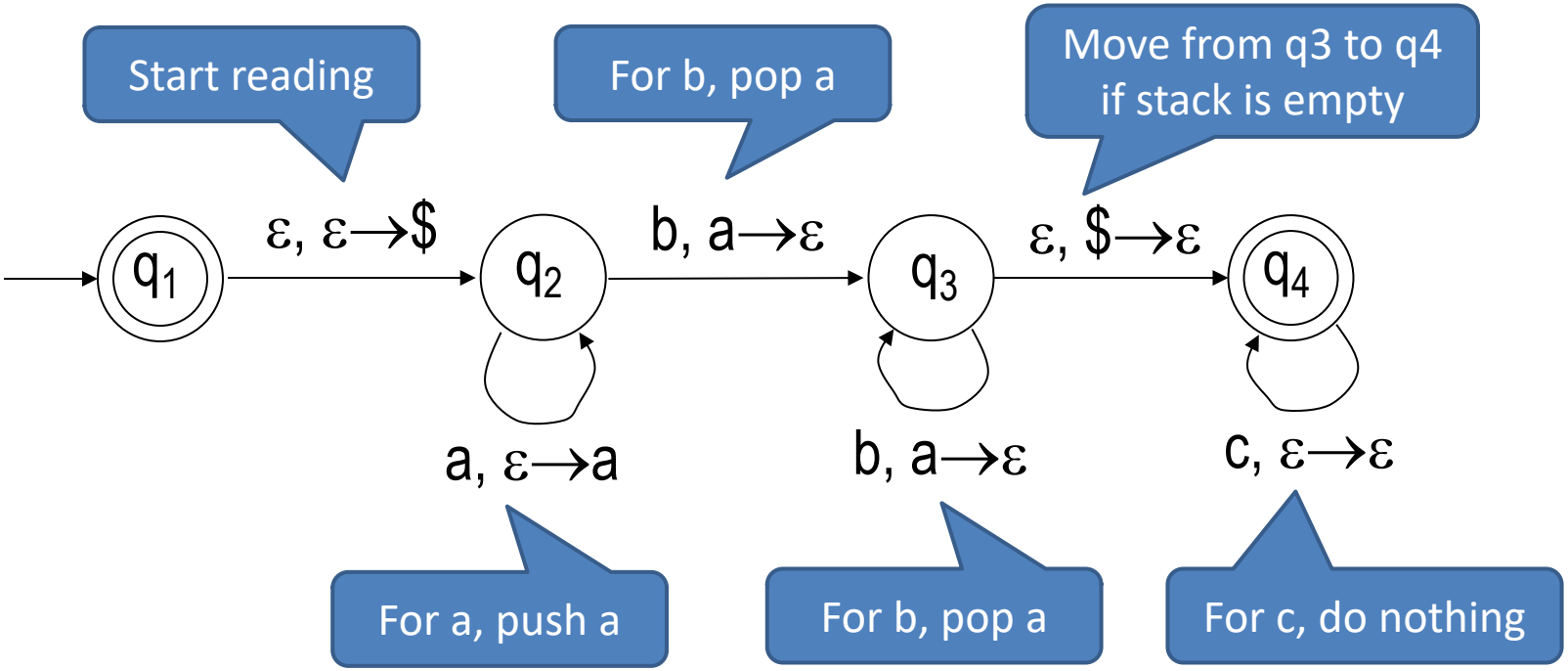



- $L(M_2)=\{ a^n b^n c^m \mid m,n \geq 0 \}$
  - Operation:
    - For an input a, and push a into stack
    - For an input b, pop one a from the stack
  - Determine accept/reject
    - If the stack is empty when finish reading b, then after reading all the cs, accept;
    - Otherwise, reject;

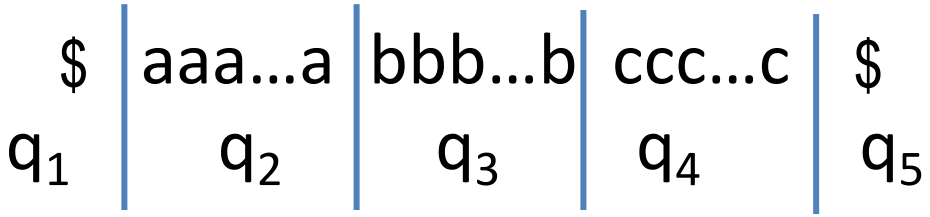
# Design PDA

| $ | aaa…a | bbb…b | ccc…c |
|---|---|---|---|
| $q_1$ | $q_2$ | $q_3$ | $q_4$ |

- $L(M_2)=\{ a^n b^n c^m \mid m,n \geq 0 \}$



# Design PDA

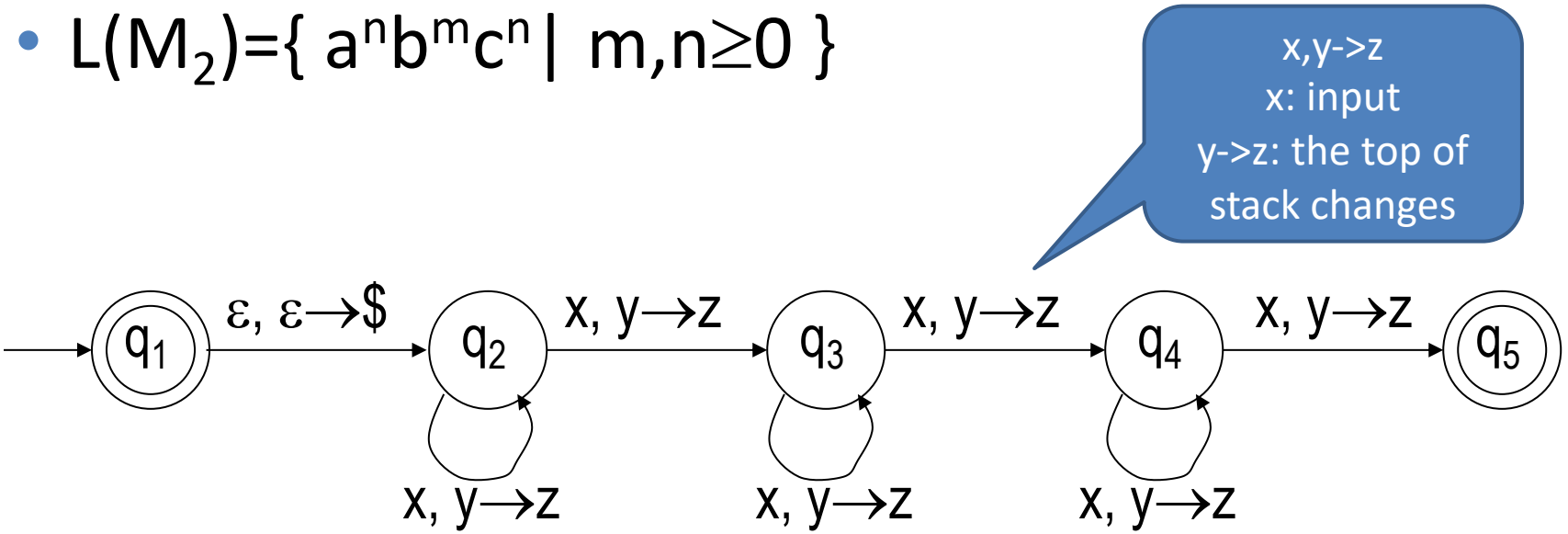



- $L(M_2)=\{ a^n b^m c^n \mid m,n \geq 0 \}$
  - Operation:
    - For an input a, and push a into stack
    - After reading some bs, every time, for an input c, pop one a from the stack
  - Determine accept/reject
    - If the stack is empty when input is done, accept;
    - Otherwise, reject.

# Design PDA

| $ | aaa…a | bbb…b | ccc…c | $ |
|---|---|---|---|---|
| $q_1$ | $q_2$ | $q_3$ | $q_4$ | $q_5$ |

- $L(M_2)=\{ a^n b^m c^n \mid m,n \geq 0 \}$

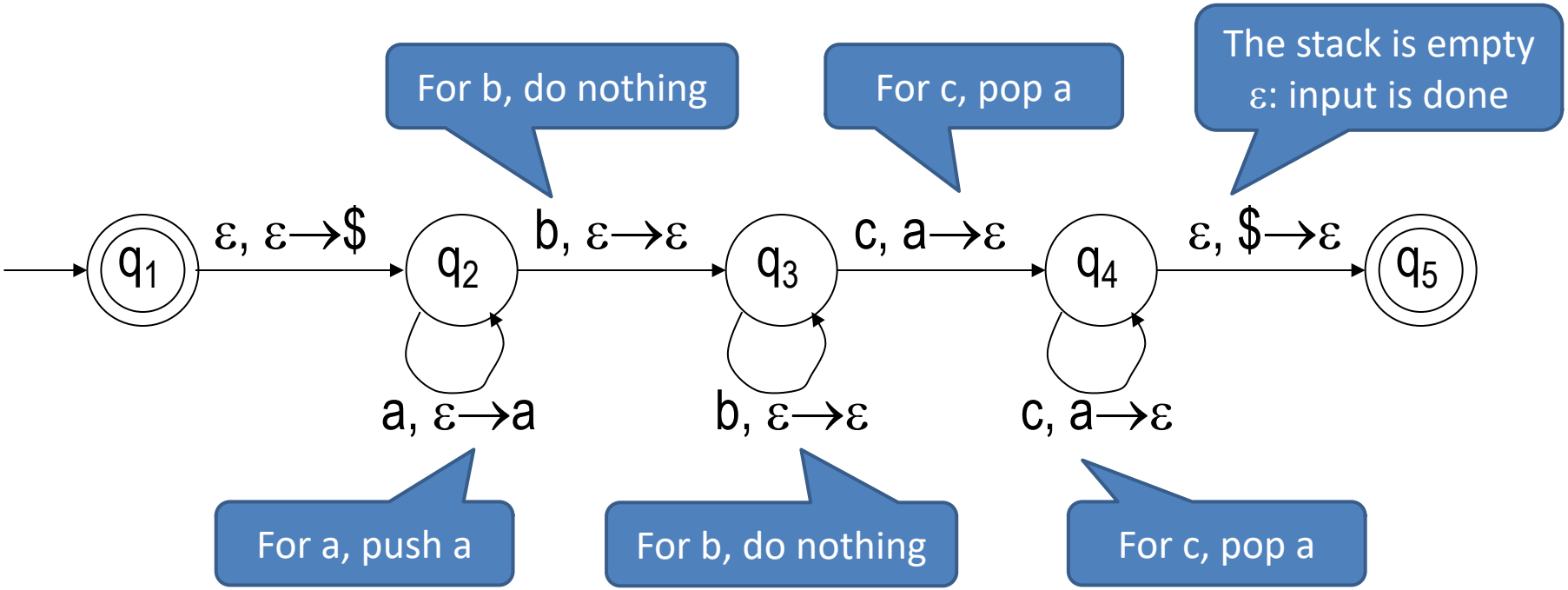



- $L(M_2)=\{ a^n b^m c^n \mid m,n \geq 0 \}$
  - Operation:
    - For an input a, and push a into stack
    - After reading some bs, every time, for an input c, pop one a from the stack
  - Determine accept/reject
    - If the stack is empty when input is done, accept;
    - Otherwise, reject.

# Design PDA

| $ | aaa…a | bbb…b | ccc…c | $ |
|---|---|---|---|---|
| $q_1$ | $q_2$ | $q_3$ | $q_4$ | $q_5$ |

- $L(M_2)=\{ a^n b^m c^n \mid m,n \geq 0 \}$

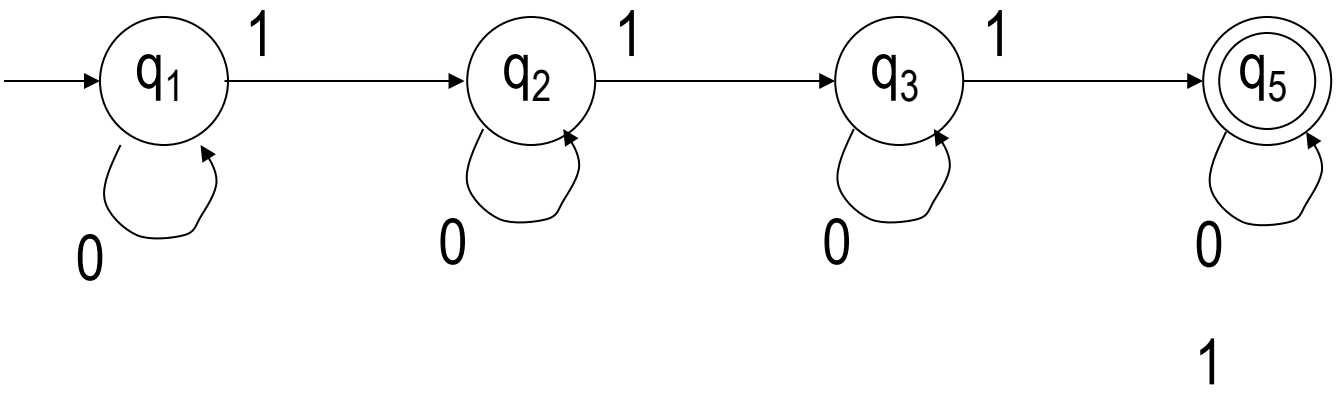


# Design PDA

- $L(M_3)=\{ w \mid$ w contains at least three 1s$\}$, input = {0, 1}
  - Input : 001101
    - Output : Accepted
  - Input : 100010
    - Output : Not Accepted
  - Regular language
    - Does not need the stack, $\varepsilon \rightarrow \varepsilon$

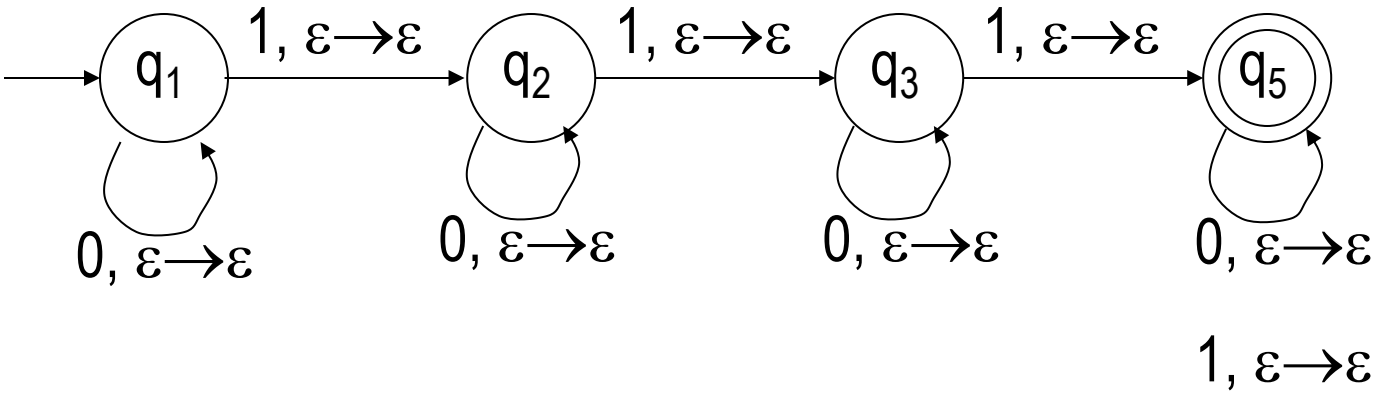
# Design PDA

- $L(M_3)$={ w| w contains at least three 1s}, input = {0, 1}



# Design PDA

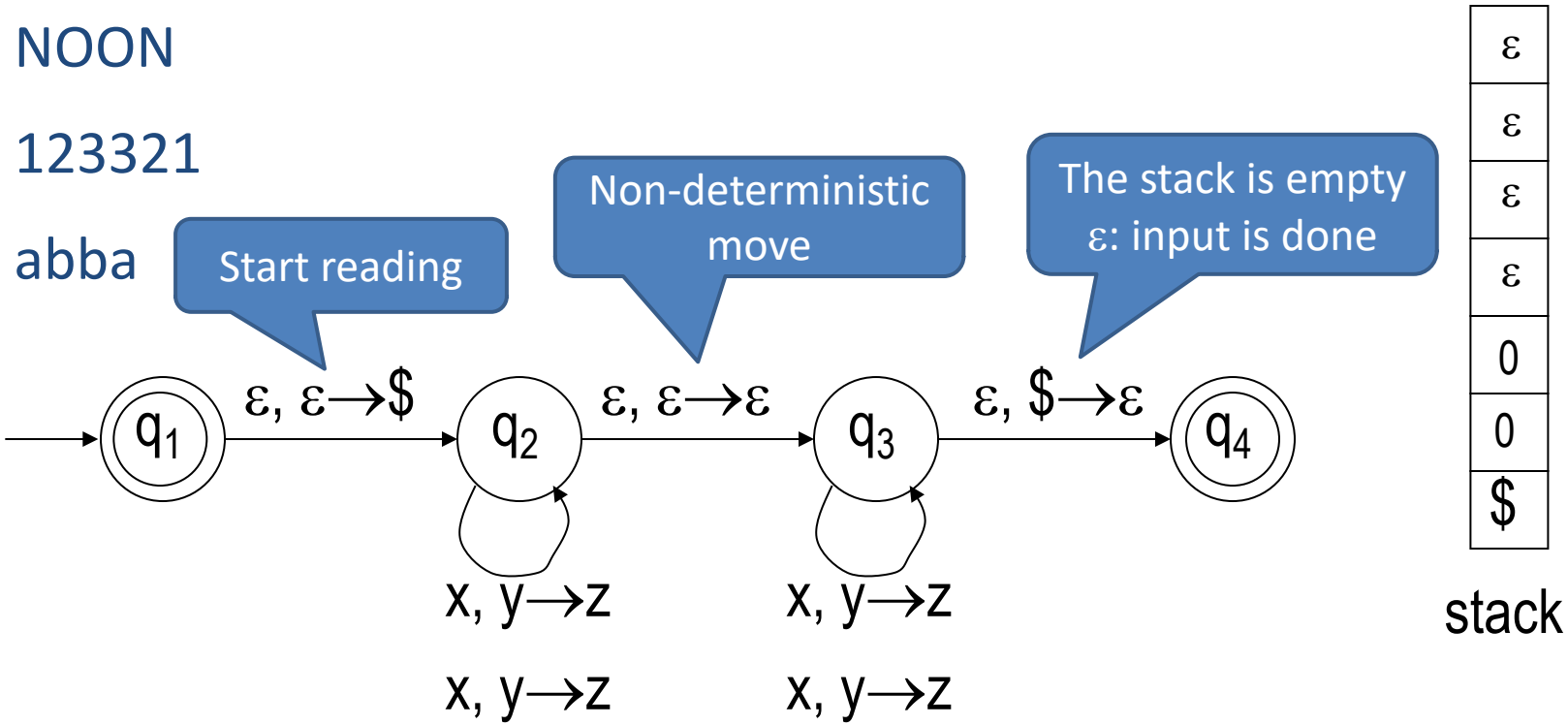
- $L(M_3)=\{ w \mid w$ contains at least three 1s$\}$, input = {0, 1}



# Design PDA: { $ww^R$ }

| $q_1$ | 111…0 | 0…111 | $q_4$ |
|---|---|---|---|
| | $q_2$ | $q_3$ | |

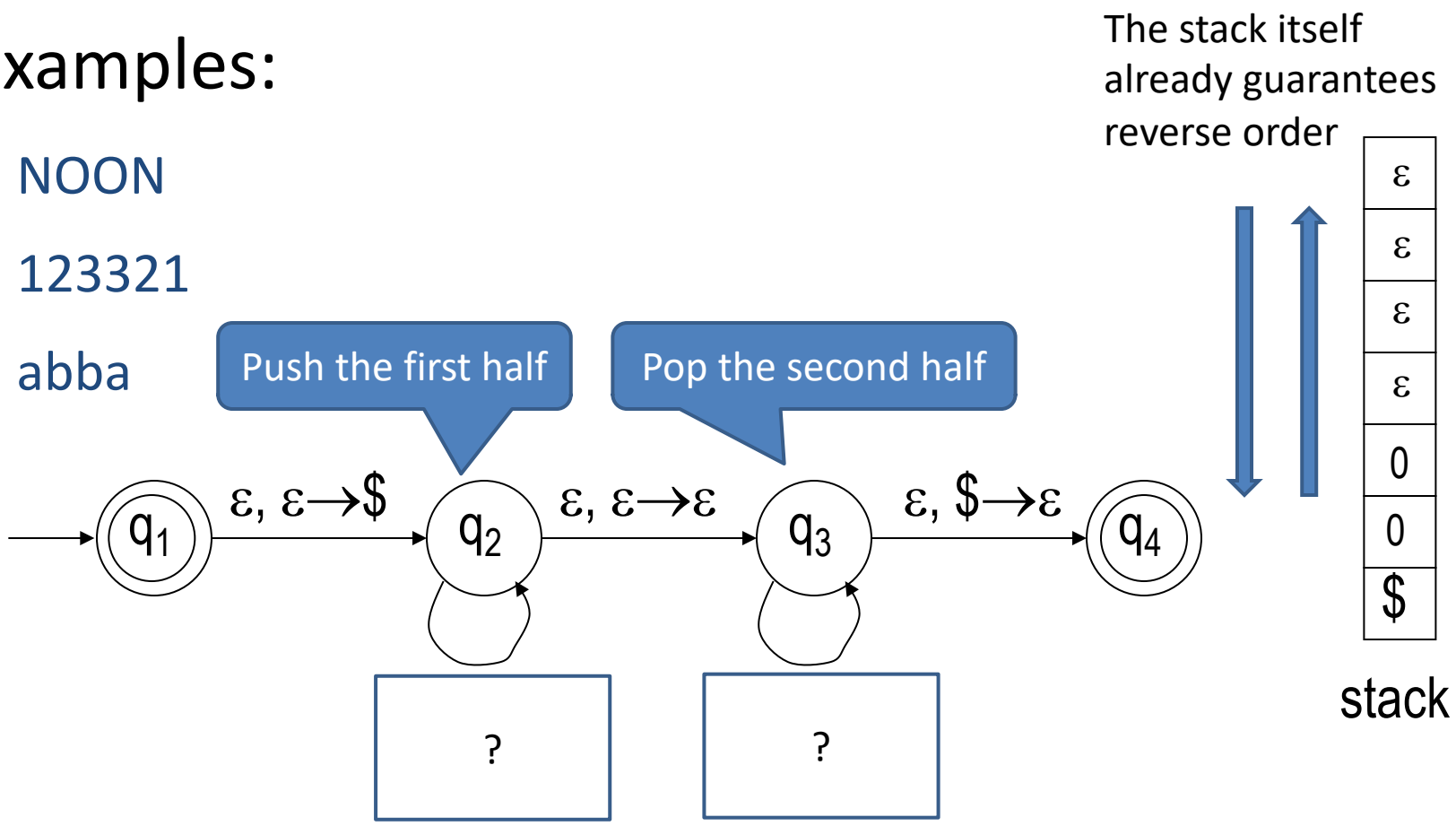
- Palindromes:
- Examples:
  - NOON
  - 123321
  - abba



# Design PDA: { $ww^R$ }

$q_1$ | abc…z $q_2$ | z…cba $q_3$ | $q_4$

- Palindromes:
- Examples:
  - NOON
  - 123321
  - abba



# Conclusion

- What is pushdown automata (PDA)?
- How to use PDA to recognize some CFL? Informal description
- Definition of PDA $M=(Q,\Sigma,\Gamma,\delta,q_0,F)$
- PDA examples, $\delta$: x, y$\rightarrow$z

PUSH z: x, $\varepsilon \rightarrow$ z

POP z. : x, z$\rightarrow \varepsilon$