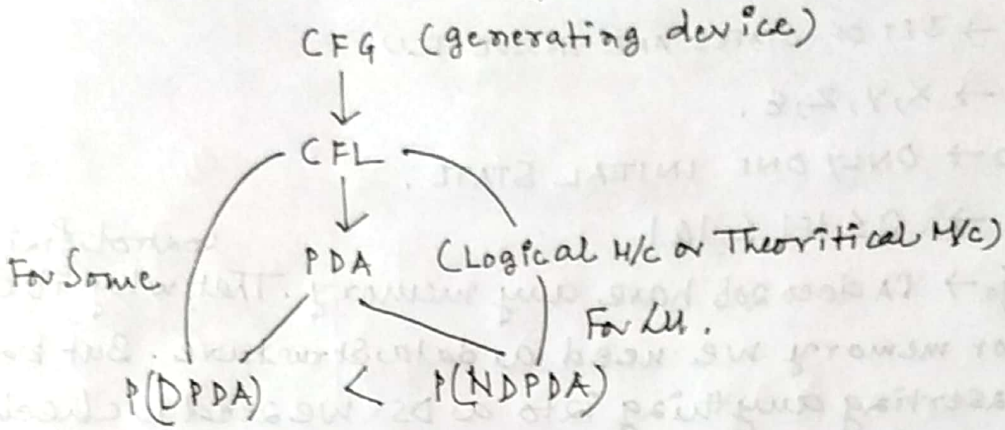
PUSH DOWN AUTOMATA (PDA)

According Chomsky Hierarchy,



There are two representation of ~~PDA~~ CFL, 1. CFG which generates it. 2. PDA which accepts it.

For all CFL we can design a NDPDA. But for some CFL we can design a DPDA.

DEFINITION OF PDA:

$PDA = \{Q, \Sigma, q_0, F, Z_0, \Gamma, \delta\}$

Q = Finite Set of States.

Σ = Input Alphabet

q_0 = Initial State.

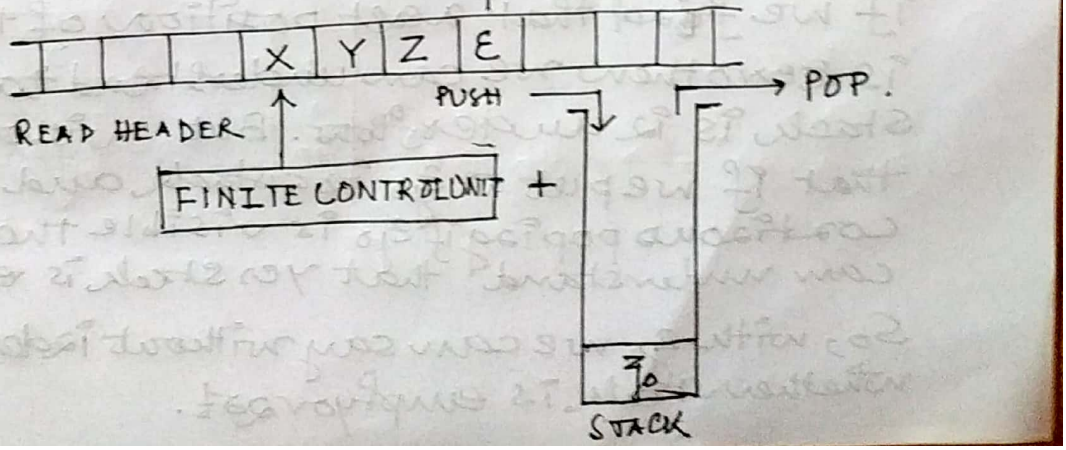
F = Set of final State

Z_0 = Bottom or initial Stack Symbol.

Γ = Stack Alphabet

δ = Transition Function

1. A PDA is nothing but a FA + STACK.
2. INPUT TAPE → IS DIVIDED INTO CELLS. EACH CELLS HAS CAPABILITY OF HOLDING SYMBOL AT A TIME.
3. WHEN WE ENCOUNTERED ε THEN WE CAN UNDERSTAND THAT IT HAS COMPLETED.
4. READ HEADER SCAN ONE SYMBOL AT A TIME AND AFTER SCANNING ONE SYMBOL IT MOVES TO THE NEXT SYMBOL.
5. FINITE CONTROL UNIT ARE THOSE WHICH CAN CHANGE TO STATES.



A PDA IS DEFINED BY 7 TUPLES.

1. $Q \rightarrow$ SET OF STATES ARE INSIDE FCU.
2. $\Sigma \rightarrow$ X, Y, Z, ϵ.
3. $q_0 \rightarrow$ ONLY ONE INITIAL STATE.
4. $F \rightarrow$ $0 \leq |F| \leq |Q|$
5. $Z_0 \rightarrow$ FA does not have any memory. That's why it can't control infinite computation. For memory we need a data structure. But before inserting anything into a DS we need to check overflow. Just like that before deleting any data we need to check underflow. Before checking overflow & underflow we need to do indexing. So, before deleting / inserting any data we need to check underflow / overflow also we need to check at which place they need to inserted or deleted.

But the advantage of stack is it is a zero address data structure. From stack we can push or pop only top element only and from one end only. So, in stack we just need to tell whether insertion need to be done or deletion need to be done. But we do not need to tell from where it need to be done.

Size of the stack is infinite. So, in stack $\exists$ no overflow. So, multiple push operations are allowed. So, we do not need to check overflow in our theoritical stack.

For checking underflow we normally do indexing in a stack such as last element position is suppose 1. So, after popping if we find that next position of the stack is zero then we can understand that the stack is in underflow. But without doing that if we put a Z_0 in a stack and after continuous popping if Z_0 is visible then we can understand that yes stack is empty.

So, with Z_0 we can say without indexing whether stack is empty or not.

$\Gamma \rightarrow$ (tow) Stack does2ot coumt aythi2g.
Suppose, we 2eed to coumt 10 2eros then, then for every value we 2eeda 2epresentive. So, we 2eed to push 10 zeroes i2to the Stack & we 2eed to pop 10 2eroes which back coem-ti2g.

Suppose we 2eed to i2sert 5.
Then 5 can be represented i2 various 20. Systeam as:

Decimal $\longrightarrow$ 5
Bi2ary $\longrightarrow$ 101
U2ary $\longrightarrow$ 11111

So, Stack follows unuary 20. system.
92 stack if we 2eed to push 2ero then we can push zero or any 2epresentati veof zero i2to the Stack.

$\Sigma \subseteq \Gamma$ & $\varepsilon_0 \in \Gamma$

So, Stack symbols are those symbols which we can pushed i2to the stack i2cludi2g, tape symbols & aditi2al symbols.

$\delta \rightarrow$ (Small Delta)

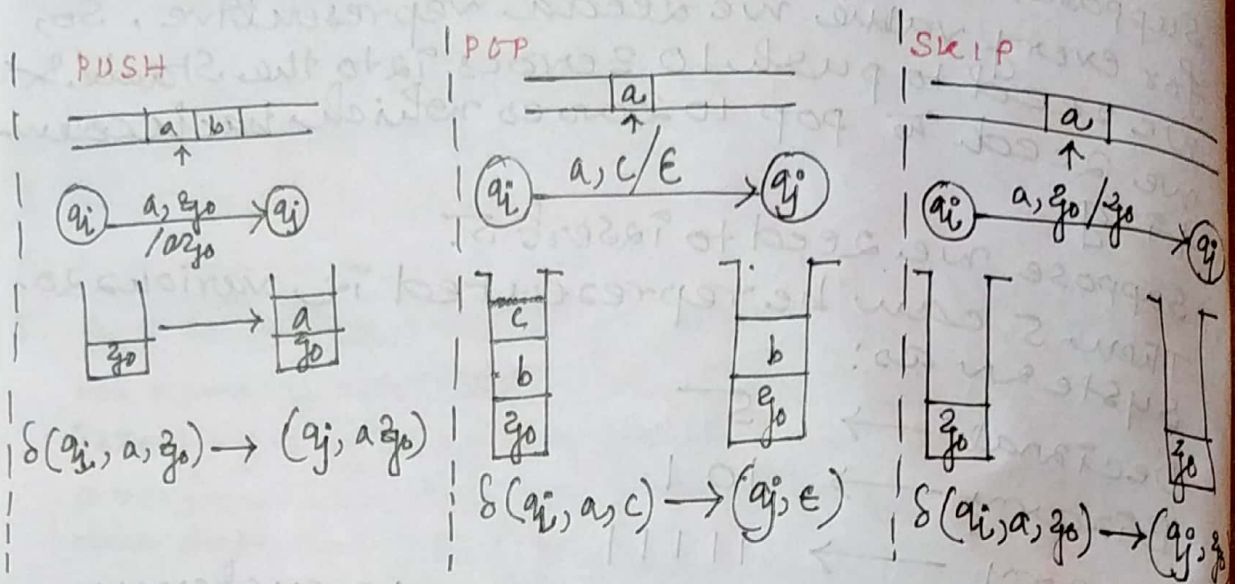
DPDA: $\delta : Q \times \Sigma \times \Gamma \longrightarrow Q \times \Gamma^*$ (we can write only δ and it is sufficient but after taki2g any transition stack allow us to ~~transit~~ take any ~~state~~ of the followi2g state:

a. 92sert (PUSH)
b. Delete (POP)
c. Skip.

DPDA: $Q \times (\Sigma \cup \in) \times \Gamma \rightarrow Q \times \Gamma^*$

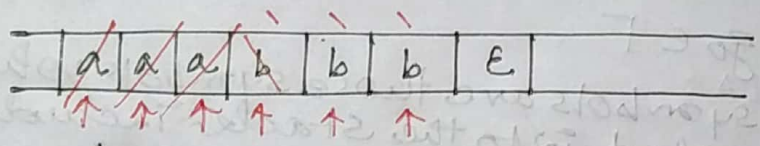
NDPDA: $Q \times \Sigma \times \Gamma \longrightarrow 2^{Q \times \Gamma^*}$

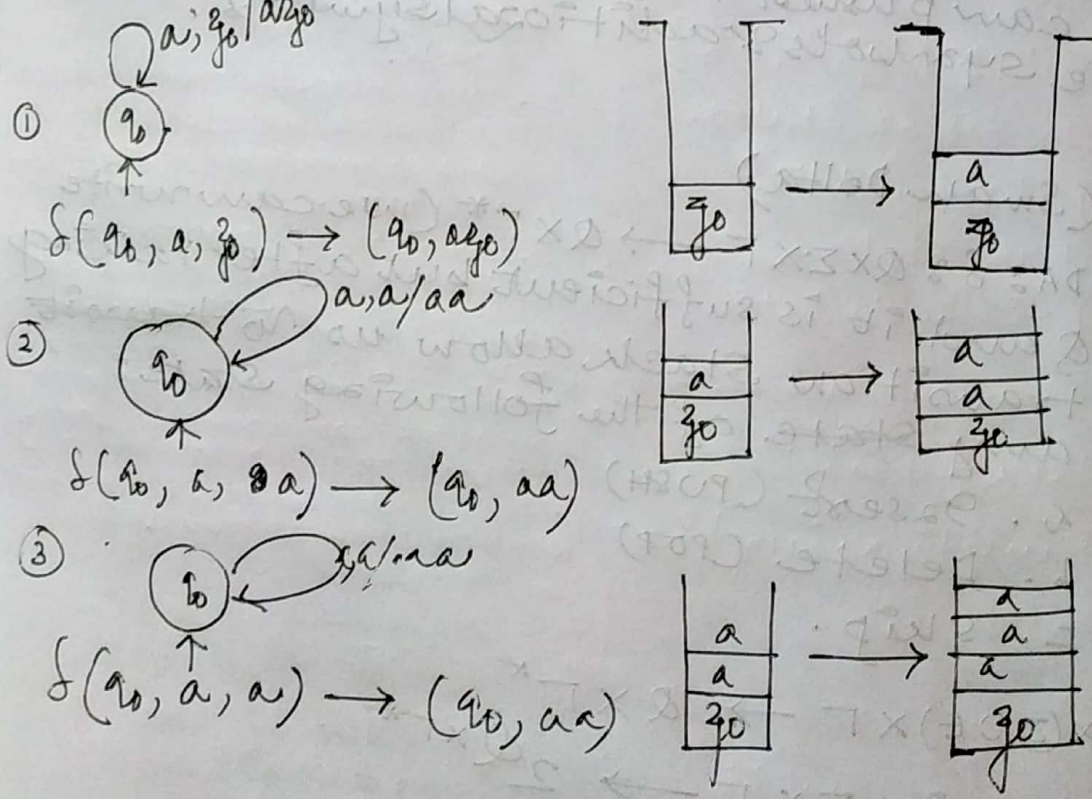
BASIC OPERATIONS ON STACK:



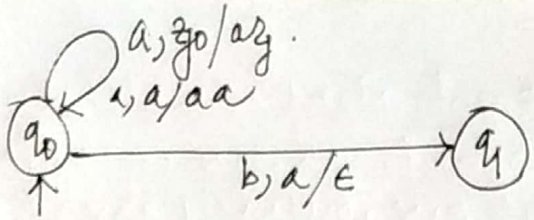
Q: $L = \{a^2 b^2 \mid 2 \geq 1\}$ — DESIGN THE PDA FOR THE FOLLOWING L.

ANSWER:

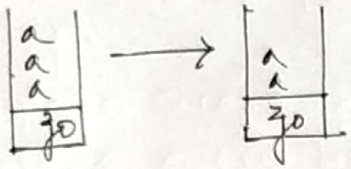




④

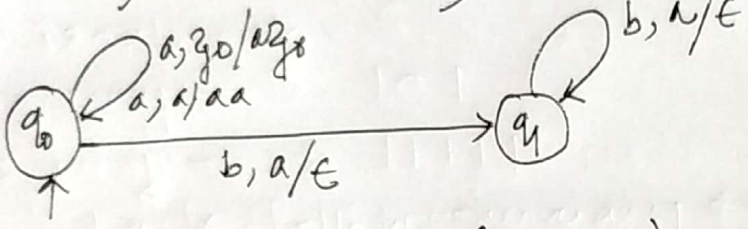


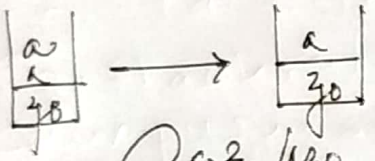
We need to change the state to remember the order.



$\delta(q_0, b, a) \longrightarrow \{(q_1, \epsilon)\}$

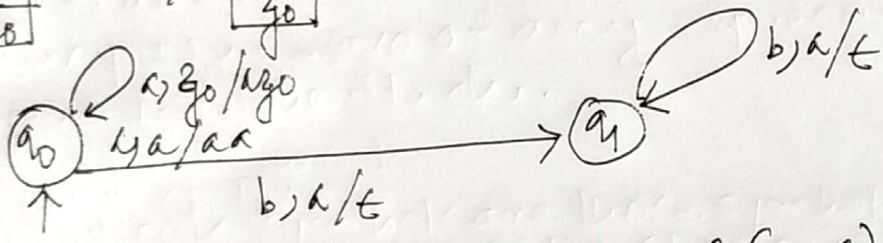
⑤

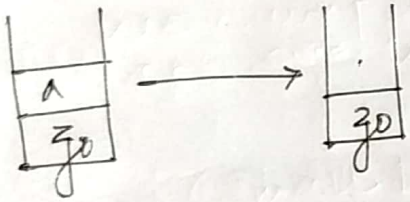




$\delta(q_1, b, a) \longrightarrow \{(q_1, \epsilon)\}$

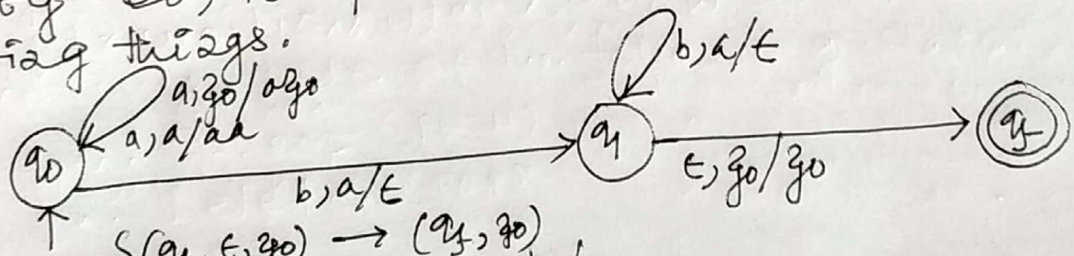
⑥





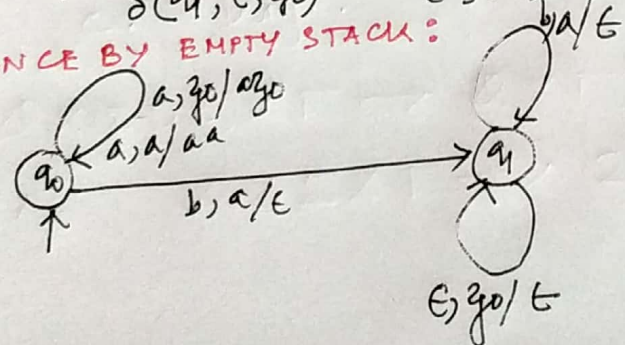
$\delta(q_1, b, a) \longrightarrow \{(q_1, \epsilon)\}$

Just to complete the i/p string at the i/p tape is not enough. We need to ensure that the stack is also empty. So, to represent it we need to do the following things.



$\delta(q_1, \epsilon, z_0) \longrightarrow (q_f, z_0)$

ACCEPTANCE BY EMPTY STACK:



$\delta(q_1, \epsilon, z_0) \longrightarrow (q_1, \epsilon)$