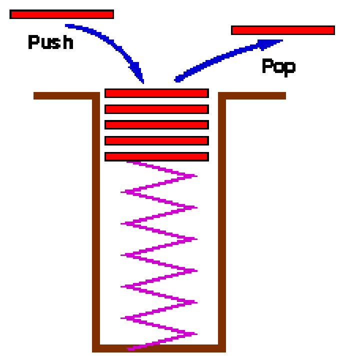
Pushdown Automata (PDA)

- Main Idea: Add a stack to an NFA
  - Stack provides potentially unlimited memory to an otherwise finite memory machine (finite memory = finite no. of states)
  - PDA = NFA + 

  - Stack is LIFO ("Last In, First Out")
  - Two operations:
    - "Push" symbol onto top of stack
    - "Pop" symbol from top of stack

6 Components of a PDA = $(Q, \Sigma, \Gamma, \delta, q_0, F)$

- Q = set of states
- Σ = input alphabet
- **Γ = stack alphabet**
- q_0 = start state
- $F \subseteq Q$ = set of accept states
- **Transition function δ: $Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \rightarrow Pow(Q \times \Gamma_\varepsilon)$**
  - **(current state, next input symbol, popped symbol) $\rightarrow$ {set of (next state, pushed symbol)}**
  - **Input/popped/pushed symbol can be ε**

New components!

When does a PDA accept a string?

- A PDA M accepts string $w = w_1 w_2 \ldots w_m$ if and only if there exists <u>at least one accepting computational path</u> i.e. a sequence of states $r_0, r_1, \ldots, r_m$ and strings $s_0, s_1, \ldots, s_m$ (denoting stack contents) such that:

1. $r_0 = q_0$ and $s_0 = \varepsilon$ *(M starts in q_0 with empty stack)*
2. $(r_{i+1}, b) \in \delta(r_i, w_{i+1}, a)$ *(States follow transition rules)*
3. $s_i = at$ and $s_{i+1} = bt$ for some $a, b \in \Gamma_\varepsilon$ and $t \in \Gamma^*$
   (M pops "a" from top of stack and pushes "b" onto stack)
4. $r_m \in F$ *(Last state in the sequence is an accept state)*

On-Board Examples

- PDA for $L = \{w\#w^R | \ w \in \{0,1\}^*\}$ (# acts as a "delimiter")
  - E.g. 0#0, 1#1, 10#01, 01#10, 1011#1101 $\in$ L
  - L is a CFL (what is a CFG for it?)
  - Recognizing L using a PDA:
    - Push each symbol of w onto stack
    - On reaching # (middle of the input), pop the stack – this yields symbols in w^R – and compare to rest of input
- PDA for $L_1 = \{ww^R | \ w \in \{0,1\}^*\}$
  - Set of all even length palindromes over {0,1}
- Recognizing L_1 using a PDA:
  - <u>Problem</u>: Don't know the middle of input string
  - <u>Solution</u>: Use nondeterminism (ε-transition) to guess!