$L = \{ \text{WW} \mid \text{W} \in \{0, 1\}^* \}$

This empty transition can be thought of as a "guess" that we're in the middle of the string.

```
INPUT = 11
```

```
\[
\begin{array}{c}
\text{INPUT} \\
1
\end{array}
\]
```

```
\[
\begin{array}{c}
\text{ARE ANY OF THESE FINAL STATES, ACCEPTING STATES?} \\
\text{YES} \Rightarrow \text{ACCEPT 11}
\end{array}
\]
```
L = \{ \text{ww} \mid \text{w} \in \{0,1\}^* \}

CFG G: S \rightarrow OSO \mid 1S1 \mid \varepsilon

Q. Can we convert CFG G to a PDA?
(Could you write a program to recognize the strings of a language using only the stack data structure?) Recognizing the strings of a language is basically what a compiler does.

INPUT X = 0110

Parse X using G

```
Input | State | Stack
0     | 1     | S
1     | 1     | 0
0     | 0     | 0
$     | 1     |
```

"TOP DOWN" parsing

Process:
1. Push the start symbol onto the stack.
2. Use grammar rules to convert stack symbols to all possible strings.
3. When the next input symbol matches the top of the stack, pop and move to next input symbol.
4. Repeat 2 and 3 until end of input.
PDA

$\delta(0,0 \rightarrow \varepsilon, 0, \varepsilon \rightarrow S, q_1, q_2, q_3, q_4)\delta(0,1 \rightarrow \varepsilon, 0, \varepsilon \rightarrow S, q_1, q_2, q_3, q_4)\delta(\varepsilon, \varepsilon \rightarrow 050, 0, \varepsilon \rightarrow S, q_1, q_2, q_3, q_4)\delta(\varepsilon, \varepsilon \rightarrow \varepsilon, 0, \varepsilon \rightarrow S, q_1, q_2, q_3, q_4)\delta(\varepsilon, \varepsilon \rightarrow \varepsilon, 0, \varepsilon \rightarrow S, q_1, q_2, q_3, q_4)$

$\varepsilon, s \rightarrow 050$ is shorthand for 3 pushes onto the stack.

Push the last symbol first.