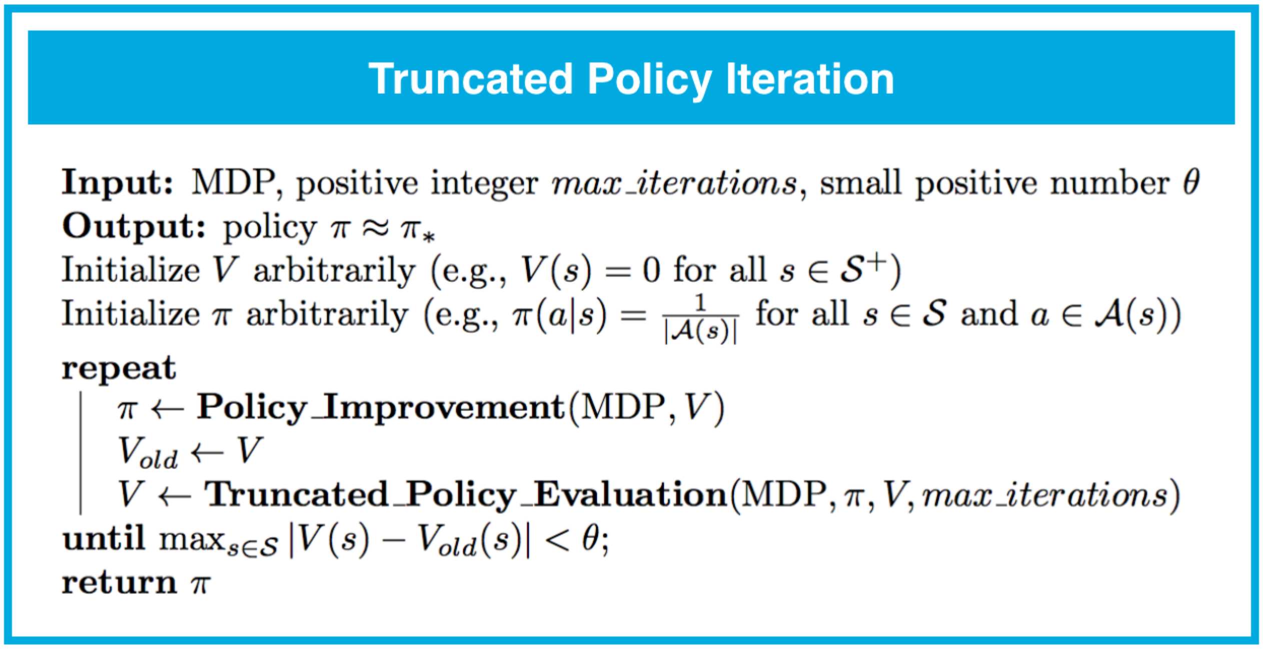
The pseudocode can be found below.



Truncated Policy Iteration

```
Input: MDP, positive integer $max\_iterations$, small positive number $\theta$
Output: policy $\pi \approx \pi_*$
Initialize $V$ arbitrarily (e.g., $V(s) = 0$ for all $s \in \mathcal{S}^+$)
Initialize $\pi$ arbitrarily (e.g., $\pi(a|s) = \frac{1}{|\mathcal{A}(s)|}$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$)
repeat
    $\pi \leftarrow$ Policy_Improvement(MDP, $V$)
    $V_{old} \leftarrow V$
    $V \leftarrow$ Truncated_Policy_Evaluation(MDP, $\pi$, $V$, $max\_iterations$)
until $\max_{s \in \mathcal{S}} |V(s) - V_{old}(s)| < \theta$;
return $\pi$
```

You may also notice that the stopping criterion for truncated policy iteration differs from that of policy iteration. In policy iteration, we terminated the loop when the policy was unchanged after a single policy improvement step. In truncated policy iteration, we stop the loop only when the value function estimate has converged.

You are strongly encouraged to try out both stopping criteria, to build your intuition. However, we note that checking for an unchanged policy is unlikely to work if the hyperparameter `max_iterations` is set too small. (To see this, consider the case that `max_iterations` is set to a small value. Then even if the algorithm is far from convergence to the optimal value function v_* or optimal policy π_*, you can imagine that updates to the value function estimate V may be too small to result in any updates to its corresponding policy.)

Please use the next concept to complete **Part 5: Truncated Policy Iteration** of `Dynamic_Programming.ipynb`. Remember to save your work!