Implementation: MC Prediction (Action Values)

The pseudocode for (first-visit) MC prediction (for the action values) can be found below. (*Feel free to implement either the first-visit or every-visit MC method. In the game of Blackjack, both the first-visit and every-visit methods return identical results.*)

First-Visit MC Prediction (for Action Values)

Input: policy π, positive integer $num\_episodes$
Output: value function Q ($\approx q_\pi$ if $num\_episodes$ is large enough)
Initialize $N(s,a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
Initialize $returns\_sum(s,a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
for $i \leftarrow 1$ ***to*** $num\_episodes$ **do**
    Generate an episode $S_0, A_0, R_1, \ldots, S_T$ using π
    for $t \leftarrow 0$ ***to*** $T-1$ **do**
        if (S_t, A_t) *is a first visit (with return* G_t*)* **then**
            $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$
            $returns\_sum(S_t, A_t) \leftarrow returns\_sum(S_t, A_t) + G_t$
    end
end
$Q(s,a) \leftarrow returns\_sum(s,a)/N(s,a)$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$
return Q

Both the first-visit and every-visit methods are **guaranteed to converge** to the true value function, as the number of visits to each state-action pair approaches infinity. (*So, in other words, as long as the agent gets enough experience with each state-action pair, the value function estimate will be pretty close to the true value.*)

We won't use MC prediction to estimate the action-values corresponding to a deterministic policy; this is because many state-action pairs will *never* be visited (since a deterministic policy always chooses the *same* action from each state). Instead, so that convergence is guaranteed, we will only estimate action-value functions corresponding to policies where each action has a nonzero probability of being selected from each state.

If you'd like to reference the pseudocode while working on the notebook, you are encouraged to open **this sheet** in a new window.

Feel free to check your solution by looking at the corresponding section in `Monte_Carlo_Solution.ipynb` .