Part 1 – Question 1

To determine convergence for the First-Visit Monte Carlo algorithm, I monitored the maximum absolute change in the value function between consecutive episodes. Whenever $max|V_t(s) - V_{t-1}(s)|$ dropped below ε = 0.001 for 50 consecutive episodes, I considered the algorithm converged. This method reflects stabilization of value estimates and avoids falsely detecting convergence due to temporary low-variance updates.

Part 2 – Question 2

Part 1 and Part 2 did not converge in exactly the same number of episodes. First-Visit updates each state once per episode, while Every-Visit updates each time the state appears. Every-Visit collects more return samples and may converge faster or show more variability early on, producing different convergence lengths.

Part 3 – Question 3

Monte Carlo Learning (On-Policy-Plus) is more similar to SARSA. It uses ε-greedy action selection and evaluates returns under the same policy that generates actions. Unlike Q-Learning, which is off-policy and evaluates using max actions, On-Policy-Plus follows the behavior policy, making it on-policy like SARSA.

Part 4 – Question 5

Convergence for Q-Learning was determined using the maximum change in Q-values between consecutive episodes: $max|Q_t - Q_{t-1}|$. If this remained below ε = 0.001 for 50 episodes, convergence was declared. This aligns with standard TD convergence criteria and ensures Q-values have stabilized.

Part 4 – Question 6

The optimal path extracted from the final Q matrix was 7 → 6 → 2 → 1. This is an optimal shortest path to the terminal state. Variations earlier in training arise from ε-greedy exploration, but the stabilized Q-values produce a consistent optimal route.

Part 5 – Question 7

SARSA convergence was determined using the same Q-value stabilization test as Q-Learning: $max|Q_t - Q_{t-1}| < 0.001$ for 50 consecutive episodes. SARSA is on-policy and more conservative, so this tolerance ensures values genuinely stabilize despite early exploration noise.

Part 5 – Question 8

The SARSA-derived greedy path was 7 → 3 → 2 → 1. This path reaches the terminal but is not the shortest path. Because SARSA is on-policy and learns from exploratory actions, it may converge to adequate, but not optimal, paths.

Part 5 – Question 9

Parts 4 and 5 did not converge in the same number of episodes. Q-Learning updates toward the maximum possible action value, making it more aggressive and sometimes faster. SARSA updates toward the value of the action actually taken, which is more conservative and may converge more slowly.

Part 6 – Question 10

Convergence for the Decaying ε-Greedy Q-Learning algorithm used the same Q-value threshold method: $max|Q_t - Q_{t-1}| < 0.001$ for 50 consecutive episodes. As ε decays, exploration decreases and Q-values stabilize more rapidly.

Part 6 – Question 11

The optimal path found was 7 → 6 → 5 → 1. This is a valid shortest path. Because ε decays over time, the agent increasingly exploits better actions, reinforcing the best route consistently.

Part 6 – Question 12

Parts 4, 5, and 6 did not converge in the same number of episodes. Q-Learning converges aggressively but noisily, SARSA more conservatively, and decaying ε-greedy converges the fastest because exploration gradually decreases, enabling stable exploitation.

Part 7 – Question 13

The cumulative average rewards are essentially identical across all three algorithms (Q-Learning, SARSA, and Decaying ε-Greedy) because the reward structure gives +100 for reaching the terminal state and 0 for other transitions. Since all three algorithms successfully reach the terminal state in every episode, they all achieve the same total reward of 100 per episode, resulting in identical cumulative averages. Any minor differences would only appear if episodes timed out without reaching the goal, but with proper convergence, all three methods achieve the same performance in this reward structure.

Part 7 – Question 14

The cumulative reward plots differ from the lecture examples because the assignment uses a reward structure with +100 only at the terminal state and 0 for all steps. The lecture examples included positive/negative step rewards, producing varying average returns. With only terminal rewards, all algorithms converge to a flat line at around +100.