Task 1.1 – Optimal Policy π*

We draw arrows in each cell pointing toward the optimal direction that maximizes expected reward.

Since the environment is deterministic, the agent should always take the shortest safe path to the goal (3,4) and avoid the danger (3,2) and the wall (2,2).

(3,1) → (3,2) □ ← DANGER

(3,3) → (3,4) □ ← GOAL

(2,3) ↓

(2,4) ↓

(1,1) → or ↓

(1,2) →

(1,3) →

(1,4) ↓

Resulting policy (π*(s)) in arrows:

↓	×	→	GOAL
↓	WALL	↓	↑
→	→	→	↑

****

Task 1.2 – Compute Optimal Value Function (γ = 0.9, H = 100)

- $V(4, 3)$

(3,4) — Goal State = 1 (terminal reward)

- $V(3, 3)$

$(3,3) \rightarrow (3,4) = \gamma \cdot V*(3,4) = 0.9 \cdot 1 = 0.9$

- $V(2, 3)$

$(2,3) \rightarrow (3,3) \rightarrow (3,4) = \gamma \cdot V*(3,3) = 0.9 \cdot 0.9 = 0.81$

- $V(3, 1)$

$(3,1)\rightarrow(2,1)\rightarrow(1,1)\rightarrow(1,2)\rightarrow(1,3)\rightarrow(1,4)\rightarrow(2,4)\rightarrow(3,4) = V*(3,1) = \gamma 7 \cdot 1 = 0.9(7) \approx 0.4782969$

- $V(1, 1) =$

$(1,1) \rightarrow (1,2) \rightarrow (1,3) \rightarrow (1,4) \rightarrow (2,4) \rightarrow (3,4) = V*(1,1) = \gamma 5 \cdot 1 = 0.9(5) \approx 0.59049$

****

Task 1.3 – Value Estimates with Probabilistic Transitions (P = 0.8, γ = 0.7)

- $V*(4, 3)$ = Invalid (Grid has only 3 rows)

- $V*(3, 3)$

$(3,3) \rightarrow (3,4)$ with $P = 0.8 \approx 0.8 \cdot 0.7 \cdot 1 = 0.56$

****

Task 2.1: Explain the exploration vs. exploitation problem in RL

Exploration: Trying new actions to discover their rewards.

Exploitation: Choosing the best-known action to maximize reward.

Balance: The agent must explore enough to learn good strategies but exploit known rewards to perform well.

Task 2.2: Explain the credit-assignment problem in RL

In RL, it's difficult to determine which action was responsible for a reward, especially when rewards are delayed.

The credit assignment problem is about figuring out how to assign reward value to previous actions.

Task 2.3: What is the Markov property?

A system has the Markov property if the next state depends only on the current state and action, not on the sequence of previous states.

Task 2.4: Motivate whether or not chess satisfies the Markov property

Yes and No:

Yes in standard RL modeling: the full board state contains all necessary information for decision-making.

No from a rules perspective: some actions (e.g., castling rights, en passant) depend on history, violating the strict Markov property.

Task 2.5: What is the difference between Q-learning and Deep Q-learning?

Aspect	**Q-learning**	**Deep Q-learning (DQN)**
State Representation	Uses a table (Q-table) for all states/actions	Uses a neural network to estimate Q-values
Best For	Small/simple environments	Large/complex environments (like Atari)
Scalability	Does not scale well to many states	Scales well with deep learning
Input Type	Works with discrete states	Works with images, sensors, raw data, etc.
Memory Usage	Needs memory for all state-action pairs	Learns with compact model parameters

DQN extends Q-learning to complex tasks using deep learning, making it suitable for Atari games.