

CSE 415 Autumn 2026 Assignment 4

Written Exercises I: Search, Games, MDPs and RL

Due Friday, November 6, at 11:59 PM. Answers are accepted without penalty until Sunday, November 8, at 11:59 PM. **Submit your answers in PrairieLearn**, in the assessment “A4: Written Exercises I”. This PDF shows the same questions, so that you can print it and work on paper; the boxes are for your own work, and nothing is collected from this handout.

In PrairieLearn, click **Save** after you enter each answer. You can change a saved answer as often as you like until the deadline. Your answers are **not graded until after the deadline**, so PrairieLearn will not tell you whether an answer is right; check your own work. In a “select all that apply” part, check at least one box: if one is left with no box checked, PrairieLearn marks the save invalid, and that whole question will score 0. Do not click “Finish assessment” until you are done with all twelve exercises: after that you cannot change your answers.

Do all **twelve** exercises; they total **70 points**. Each exercise starts on a new page. Every student gets the same instances, so that you can discuss them with TAs in office hours. If any corrections are made to this assignment, they will be announced on Ed.

This is an *individual-work* assignment. You may discuss the methods with others, but work out your own answers. You may use generative AI tools to help you learn the methods; the answers you submit must be your own work, and you should be able to do problems like these without AI on the midterm.

Version of October 24, 2026. Any updates will be announced on Ed.

1 State Spaces

(5 points)

Consider the Towers of Hanoi with **4 disks** and **3 pegs**. The disks are numbered 1 (smallest) to 4 (largest). A state lists the disks on each peg from bottom to top; for example **3 1 | -- | 2** means disks 3 and 1 are on peg 1 (1 on top), peg 2 is empty, and disk 2 is on peg 3.

As in Lecture 03 (slide 14), operator **Move- i - j** moves the top disk of peg i onto peg j . Its precondition: peg i is not empty, and peg j is either empty or has a top disk larger than the one being moved.

- (a) (1 point) How many states does this problem have? (Count every arrangement in which no disk is on top of a smaller one.)

For parts (b)–(d), the current state s is: Peg 1: 3 1, Peg 2: 2, Peg 3: 4 (that is, **3 1 | 2 | 4**).

- (b) (1 point) Which operators are applicable in s (their preconditions are true)? Choose from: **Move-1-2, Move-1-3, Move-2-1, Move-2-3, Move-3-1, Move-3-2**.

- (c) (1 point) Which state results from applying **Move-1-3** to s ? Choose one:

- (i) **3 | 2 1 | 4**
- (ii) **1 | 2 | 4 3**
- (iii) **3 | 2 | 4 1**
- (iv) **3 1 | - | 4 2**

- (d) (1 point) Starting from s , the following six operators are applied in order:

1. **Move-1-3**, 2. **Move-2-1**, 3. **Move-3-2**, 4. **Move-2-1**, 5. **Move-3-1**, 6. **Move-1-3**

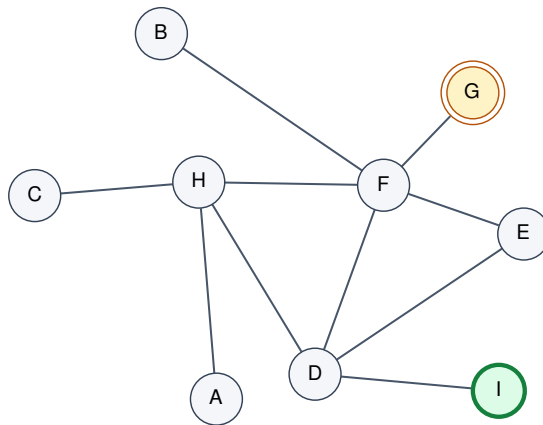
Which is the first move whose precondition is false? Give its number (1–6), or **0** if all six moves are legal.

- (e) (1 point) Now consider the Farmer-Fox-Chicken-and-Grain problem of Worksheet 03. A state is written as the initials of what is on the *left* bank (F = farmer, f = fox, c = chicken, g = grain); for example **Ffg** means the chicken is on the right bank. A state is illegal if the fox is left alone with the chicken, or the chicken is left alone with the grain. Which of these states are **illegal**? **cg, Ffc, fg, (empty), Ff, Fcg**

2 Blind Search

(6 points)

Below is a problem-space graph with 9 states. Every edge can be traveled in either direction. The start state is **I** (green, thick border) and the goal state is **G** (double circle).



State	Successors
A	H
B	F
C	H
D	E, F, H, I
E	D, F
F	B, D, E, G, H
G	F
H	A, C, D, F
I	D

Rules for every part. Use the iterative formulations from Lecture 04 (slide 14 for depth-first search, slide 15 for breadth-first search), with these details:

- Successors are generated in **alphabetical order**. In DFS they are inserted at the front of OPEN in that order, so the alphabetically first successor is selected next.
- A state is *selected* when it is removed from OPEN in step 3. The goal test happens then, so the search stops when the goal is selected. A selection order **includes the start and the goal**.

(a) (1 point) Breadth-first search: in what order are states selected?

(b) (1 point) During that breadth-first search, what is the largest number of states on OPEN at any one time? (Count OPEN after each step 5, and also at the start.)

(c) (1 point) Depth-first search: in what order are states selected?

(d) (1 point) When DFS selects the goal, follow the back pointers to the start. How many *edges* does that path have? (A state's back pointer is the state whose expansion most recently put it on OPEN.)

(e) (2 points) Iterative deepening DFS runs depth-limited DFS with limits 0, 1, 2, ... until the goal is selected. In each iteration:

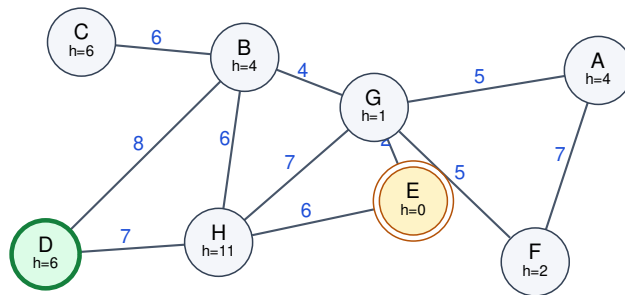
- The search is a tree search: there is no CLOSED list, but a successor is skipped if it already appears on the path from the start to the state being expanded.
- A state at depth equal to the limit is selected (and goal-tested) but not expanded.

With which depth limit is the goal first selected? And how many selections are made in total, counting every iteration (including the last one, up to and including the goal)? Show the selection order of each iteration.

3 Uniform-Cost Search and A*

(7 points)

In the weighted graph below, every edge can be traveled in either direction, and its number is its cost. The start state is **D** (green, thick border) and the goal is **E** (double circle). The table gives the value of a heuristic h at each state, an estimate of its cost to the goal.



State	h
A	4
B	4
C	6
D	6
E	0
F	2
G	1
H	11

Rules for every part. Use the algorithm of Lecture 06, slide 9. For uniform-cost search (UCS) the priority of a state is g , its cost from the start along the best path found so far. For A* the priority is $f = g + h$.

- When two or more states on OPEN share the lowest priority, select the one that is **first alphabetically**.
- A state is *selected* when it is removed from OPEN (step 3). Both searches stop as soon as the goal is selected. A selection order includes the start and the goal.
- A path is given by following back pointers from the goal; write it from the start to the goal.

(a) (3 points) Uniform-cost search (ignore h): give the selection order, the path found, and its cost.

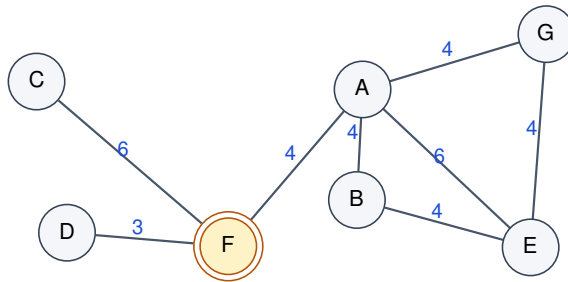
- (b) (3 points) A* with the heuristic shown: give the selection order, the path found, and its cost.

- (c) (1 point) Is the path that A* found a lowest-cost path from D to E? Explain briefly.

4 Properties of Heuristics

(8 points)

In the weighted graph below, every edge can be traveled in either direction, and its number is its cost. The goal is $\gamma = \mathbf{F}$ (double circle). The table gives three heuristics h_1 , h_2 and h_3 .



s	$h_1(s)$	$h_2(s)$	$h_3(s)$
A	4	2	2
B	6	4	4
C	8	4	3
D	2	1	1
E	10	9	6
F	0	0	0
G	8	7	4

Definitions (Lectures 05–07). $d(s, \gamma)$ is the cost of a cheapest path from s to the goal.

- h is *admissible* if $h(s) \leq d(s, \gamma)$ for every state s .
- h is *consistent* if $h(s_i) - h(s_j) \leq d(s_i, s_j)$ for every edge (s_i, s_j) , in both directions, where $d(s_i, s_j)$ is the edge's cost, and $h(\gamma) = 0$.
- For admissible h_i and h_j , h_i *dominates* h_j if $h_i(s) \geq h_j(s)$ for every state s .

(a) (2 points) Find $d(s, \gamma)$ for every state s other than the goal.

(b) (1 point) At which states is h_1 **not** admissible (that is, $h_1(s) > d(s, \gamma)$)?

- (c) (1 point) h_2 is admissible. Is it consistent? List every state s_i that has an edge (s_i, s_j) with $h_2(s_i) - h_2(s_j) > d(s_i, s_j)$ (or that is the goal with $h_2(\gamma) \neq 0$).

- (d) (1 point) h_2 and h_3 are both admissible. Which statement is true?

- (i) h_2 dominates h_3
- (ii) h_3 dominates h_2
- (iii) Neither dominates the other

- (e) (1 point) Is $h(s) = \max(h_1(s), h_3(s))$ admissible? Explain briefly.

- (f) (2 points) h_3 is consistent. Suppose we change only the value $h_3(B)$, keeping every other value of h_3 as in the table. Which values of $h_3(B)$ keep h_3 consistent? (Values must also be ≥ 0 .) Give the smallest and the largest, and show the condition each edge at B gives.

5 Eight-Puzzle Heuristics

(6 points)

An eight-puzzle state s and the goal state γ used in Lecture 07 (the gray square is the blank):

State s			Goal γ		
2	5	1		1	2
3		6	3	4	5
7	4	8	6	7	8

As in lecture, $h_{\text{hamming}}(s)$ is the number of tiles out of place, and $h_{\text{manhattan}}(s)$ is the sum of the Manhattan distances (rows plus columns) of the 8 tiles from their goal squares. The blank is not a tile.

(a) (1 point) $h_{\text{hamming}}(s) = ?$

(b) (1 point) $h_{\text{manhattan}}(s) = ?$ Show the distance of each tile.

(c) (1 point) In s , the blank moves **left** (so tile 3 slides right). What is $h_{\text{manhattan}}$ of the new state?

(d) (1 point) Consider this relaxed version of the puzzle: A tile can be lifted off the board and put down on any empty square, or set aside on the table and put on any empty square later; each lift or put-down costs $\frac{1}{2}$. The exact cost of solving the relaxed puzzle from s is which heuristic?

- (i) h_{hamming}
- (ii) $h_{\text{manhattan}}$
- (iii) $h_{\text{euclidean}}$
- (iv) h_{zero}

- (e) (1 point) Here is another relaxation of the eight puzzle. In the relaxed puzzle P' , the blank may swap places with *any* tile, anywhere on the board, not just an adjacent one; each swap costs 1. Let $h_{\text{swap}}(s)$ be the cost of an optimal solution of P' from s . (Hint: if the blank is not on its goal square, swapping in the tile that belongs there is always part of an optimal solution. If the blank is on its goal square but some tiles are still out of place, the blank must first swap with one of them.) Give $h_{\text{swap}}(s)$ and $h_{\text{swap}}(s_2)$ for this state s_2 :

State s_2

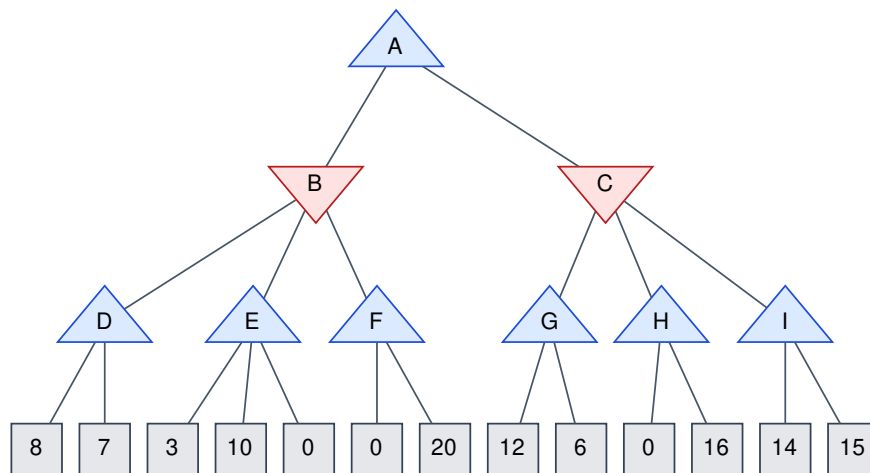
	4	2
3	1	5
6	8	7

- (f) (1 point) Which statements are true? Select all that apply. (Your answers to (b) and (e) may help.)
- (i) h_{swap} is admissible, because every legal move of the eight puzzle (the blank swaps with an adjacent tile) is also a legal move of P' with the same cost, so an optimal solution of P' costs at most $d(s, \gamma)$.
 - (ii) h_{swap} is not admissible, because $h_{\text{swap}}(s)$ can be larger than $h_{\text{hamming}}(s)$.
 - (iii) h_{swap} dominates h_{hamming} .
 - (iv) $h_{\text{manhattan}}$ dominates h_{swap} , because a tile that slides one square at a time travels at least as far as a tile that swaps directly into place.
 - (v) h_{swap} dominates $h_{\text{manhattan}}$, because one swap in P' can move a tile any distance.
 - (vi) Neither h_{swap} nor $h_{\text{manhattan}}$ dominates the other.

6 Minimax

(4 points)

In the game tree below, **Max** is to move at the root A. Up-pointing triangles are Max nodes, down-pointing triangles are Min nodes, and the squares are leaves holding static values.



(a) (2 points) Use minimax to find the backed-up value of each child of the root.

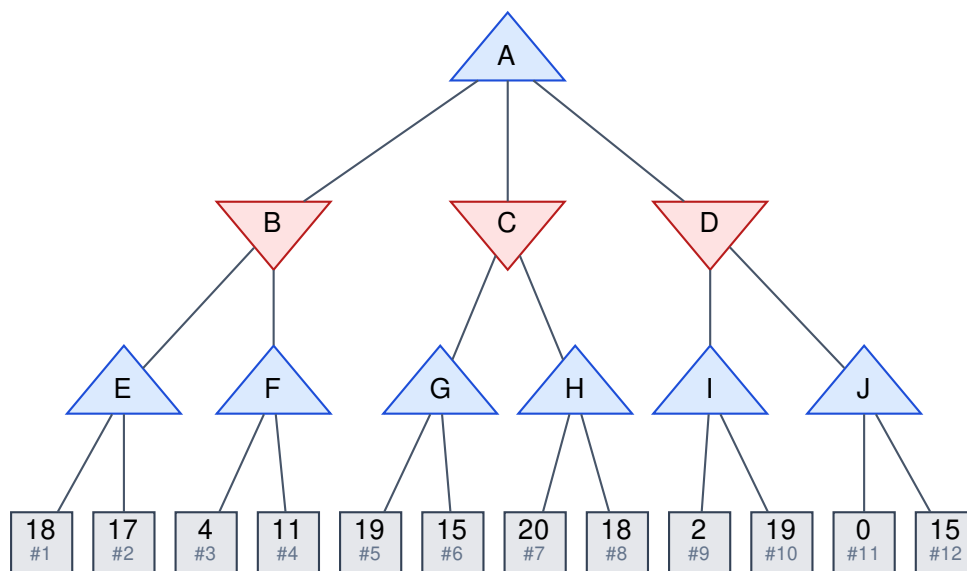
(b) (1 point) What is the value of the root A?

(c) (1 point) Which move should Max make at A?

7 Alpha-Beta Pruning

(6 points)

In the game tree below, **Max** is to move at the root A. Up-pointing triangles are Max nodes, down-pointing triangles are Min nodes, and the squares are leaves; each leaf shows its static value and, below it, its number (#1, #2, ... from left to right).



Run minimax with alpha-beta pruning as in Lecture 09 (slide 6), visiting children **from left to right**. The root is called with $\alpha = -\infty$ and $\beta = +\infty$. α and β are passed down by value and updated locally; a Max node stops searching its children as soon as its value is $\geq \beta$, and a Min node as soon as its value is $\leq \alpha$.

- (a) (2 points) Which leaves are **never evaluated**? (Give their numbers.)

- (b) (2 points) Node H (a Max node): what α and β is it called with, and what value v does it return? (If its search is cut off, v is the value it has when it stops.)

(c) (1 point) What is the value of the root A?

(d) (1 point) Suppose the children of A could be searched in any order (each subtree stays as drawn, and the search within each subtree is still left to right). Which child should be searched **first** so that alpha-beta evaluates as few leaves as possible? How many leaves are evaluated then?

8 Zobrist Hashing

(4 points)

A game is played on a 3×3 board with White and Black pieces. Positions are hashed with Zobrist hashing (Lecture 10), using the table of 8-bit random values below: the entry for square R_iC_j (row i , column j , numbered from 0) and piece p is written R_iC_jp , so R1C2W is the value for a White piece in row 1, column 2. The hash $Z(s)$ of a position is the XOR of the entries for the occupied squares; the empty board has $Z = 00000000$. White pieces are drawn as open circles and Black pieces as filled circles.

Position s				square	W (White)	B (Black)
	C0	C1	C2			
R0				R0C0	00100010	10010001
R1				R0C1	11011000	11001101
R2				R0C2	11000011	00010000
				R1C0	01000001	00011110
				R1C1	01111110	11000010
				R1C2	01110011	01111000
				R2C0	10100110	01100001
				R2C1	11001001	00110101
				R2C2	00011000	01111100

Give each hash as 8 binary digits, including leading zeros, and show the XORs you use.

(a) (2 points) Compute $Z(s)$ from scratch.

(b) (1 point) In s , the White piece on R0C0 moves to the empty square R0C1, giving s' . Compute $Z(s')$ from $Z(s)$ with XORs.

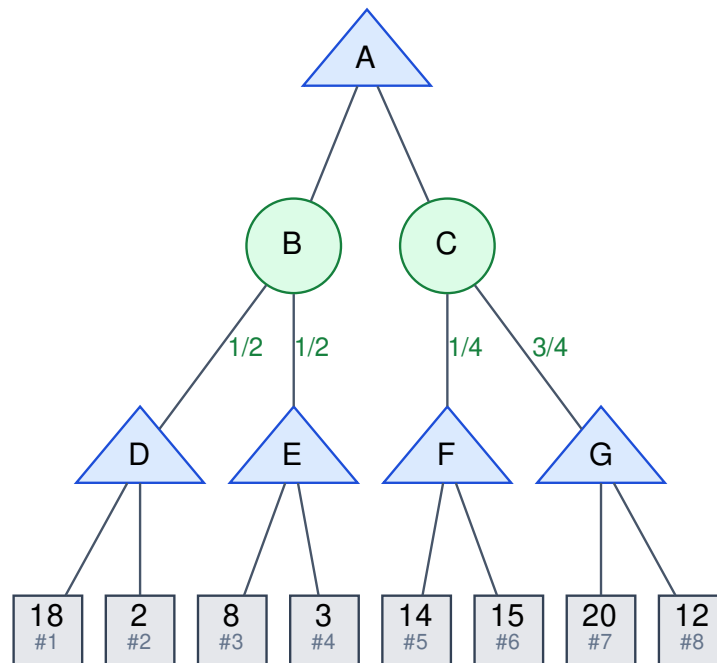
(c) (1 point) Then, in s' , the Black piece on R1C1 captures the White piece on R2C0 (the White piece is removed and the Black piece moves to R2C0), giving s'' . Compute $Z(s'')$ from $Z(s')$.



9 Expectimax

(6 points)

In the game tree below, Max is to move at the root A. Up-pointing triangles are Max nodes, circles are Chance nodes, and the squares are leaves (static value, and #number below). Each edge out of a Chance node is labeled with the probability of that outcome.



As in Lecture 11: at a Max node, $\text{value}(n) = \max$ of the values of its successors; at a Chance node, $\text{value}(n) = \sum P(s) \cdot \text{value}(s)$ over its successors s . Give values as exact decimals (for example 8.75).

(a) (2 points) The value of each Chance node below the root:

--

(b) (1 point) What is the value of the root A?

(c) (1 point) Which move should Max make at A?

(d) (2 points) Keep every other leaf fixed, and raise the value of leaf #1 (now 18). Above what value of leaf #1 does Max's best move change to the move to B? Show how you found it.

10 MDPs: Q-Values and Policies

(7 points)

A grid-world MDP, as in Lecture 13. The gray square is a wall. The agent's actions are North, East, South and West. An action moves the agent in the intended direction with probability 0.8, and in each of the two perpendicular directions with probability 0.1; if a wall or the edge of the grid is in the way, the agent stays where it is.

- The exit states are G (exit +10), J (exit -10). In an exit state the only action is *Exit*, which pays the exit reward and ends the episode (it goes to a terminal state whose value is 0).
- Every other transition pays the living reward $R(s, a, s') = -1$.
- The discount is $\gamma = 0.9$.

Each cell below shows the state's name and its optimal value $V^*(s)$, rounded to 2 decimals. Use **these rounded values** in your calculations.

A 3.29	B 5.11	C 7.13	D 5.25
E 3.51	F 5.76	G exit +10 10.00	H 7.13
I 0.69	J exit -10 -10.00		K 5.04

- (a) (3 points) Find $Q^*(F, a) = \sum_{s'} T(F, a, s') [R(F, a, s') + \gamma V^*(s')]$ for each action (to within 0.01). Show the terms.

- (b) (2 points) Extract the policy from V^* : $\pi^*(s) = \arg \max_a \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^*(s')]$, at the states A, B, C, E, H, I, K. Remember that the best action does not always point at the neighbor with the highest value.

- (c) (2 points) Here are the optimal values for the same MDP with a smaller discount, $\gamma = 0.3$ (rounded to 2 decimals). Compare the optimal policies for $\gamma = 0.9$ and $\gamma = 0.3$: at which of the states A, B, C, E, F, H, I, K is the optimal action different? Explain briefly why.

A -1.22	B -0.66	C 1.36	D -0.65
E -0.82	F 1.08	G exit +10 10.00	H 1.36
I -1.40	J exit -10 -10.00		K -0.72

11 Value Iteration

(6 points)

A grid-world MDP, as in Lecture 13. The gray square is a wall. The agent's actions are North, East, South and West. An action moves the agent in the intended direction with probability 0.8, and in each of the two perpendicular directions with probability 0.1; if a wall or the edge of the grid is in the way, the agent stays where it is.

- The exit states are E (exit +5), B (exit -5). In an exit state the only action is *Exit*, which pays the exit reward and ends the episode (it goes to a terminal state whose value is 0).
- Every other transition pays the living reward $R(s, a, s') = -1$.
- The discount is $\gamma = 0.9$.

A	B exit -5	C	D
E exit +5	F		G
H	I	J	K

Run value iteration (Lecture 14), starting from $V_0(s) = 0$ for every state:

$$V_{k+1}(s) = \max_a \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V_k(s')].$$

Every V_{k+1} is computed from V_k only.

- (2 points) Give $V_1(s)$ for every state.
- (4 points) Give $V_2(s)$ for every state, to within 0.01, and show the computation for each state whose V_2 is not simply the same for all its actions.

Put both in one table:



12 Q-Learning

(5 points)

A Q-learning agent has states A, B, C, D and actions *left* and *right*. Its current Q-values are:

s	$Q(s, \text{left})$	$Q(s, \text{right})$
A	3	3
B	7	7
C	0	-3
D	2	10

The learning rate is $\alpha = 0.5$ and the discount is $\gamma = 0.9$. For each observed transition (s, a, r, s') , the agent performs the Q-learning update of Lecture 15:

$$\begin{aligned} \text{sample} &= r + \gamma \max_{a'} Q(s', a') \\ Q(s, a) &\leftarrow (1 - \alpha) Q(s, a) + \alpha [\text{sample}] \end{aligned}$$

The agent observes these two transitions, **in this order**:

1. $(s = D, a = \text{right}, r = 0, s' = B)$
2. $(s = A, a = \text{left}, r = 2, s' = D)$

Give values to within 0.01, and show the sample and the update.

(a) (2 points) After transition 1, what is $Q(D, \text{right})$?

(b) (2 points) After transition 2, what is $Q(A, \text{left})$?

(c) (1 point) After both updates, which action is greedy (has the larger Q-value) in state A?