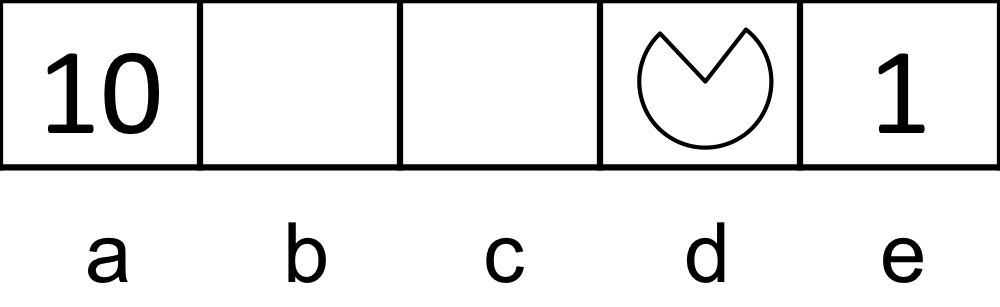
**Homework 5: MDPs**

**Name: *Ngoc Ha***

**Question 1: Solving MDPs**

0.0/6.0 points

Consider the gridworld MDP for which Left and Right actions are 100% successful. Specifically, the available actions in each state are to move to the neighboring grid squares. From state *a*, there is also an exit action available, which results in going to the terminal state and collecting a reward of 10. Similarly, in state *e*, the reward for the exit action is 1. Exit actions are successful 100% of the time.



Let the discount factor γ = 1. Fill in the following quantities.

V0(d) =



V1(d) =



V2(d) =



V3(d) =



V4(d) =



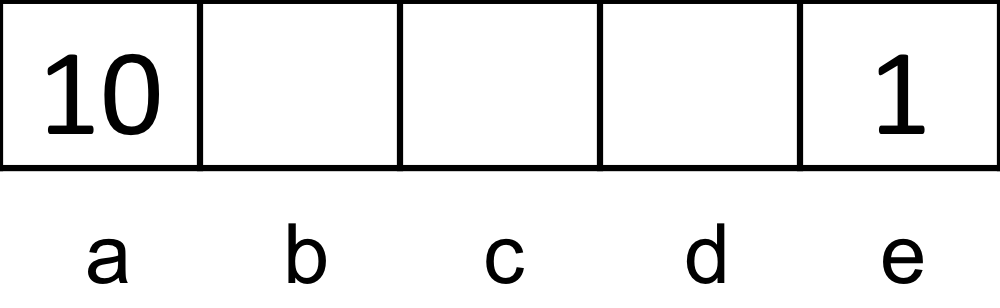
V5(d) =



**Question 2: Value Iteration Convergence Values**

0.0/5.0 points

Consider the gridworld where Left and Right actions are successful 100% of the time. Specifically, the available actions in each state are to move to the neighboring grid squares. From state *a*, there is also an exit action available, which results in going to the terminal state and collecting a reward of 10. Similarly, in state *e*, the reward for the exit action is 1. Exit actions are successful 100% of the time.



Let the discount factor γ = 0.2. Fill in the following quantities.

V\*(a) = V∞(a) =



V\*(b) = V∞(b) =



V\*(c) = V∞(c) =



V\*(d) = V∞(d) =



V\*(e) = V∞(e) =

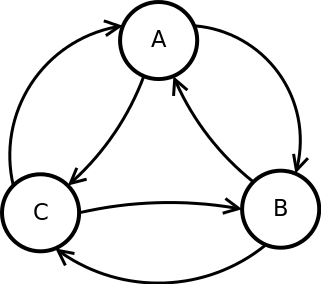
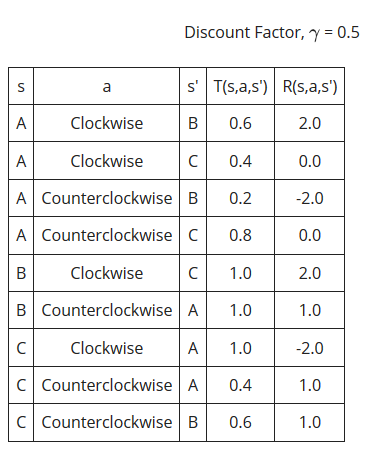


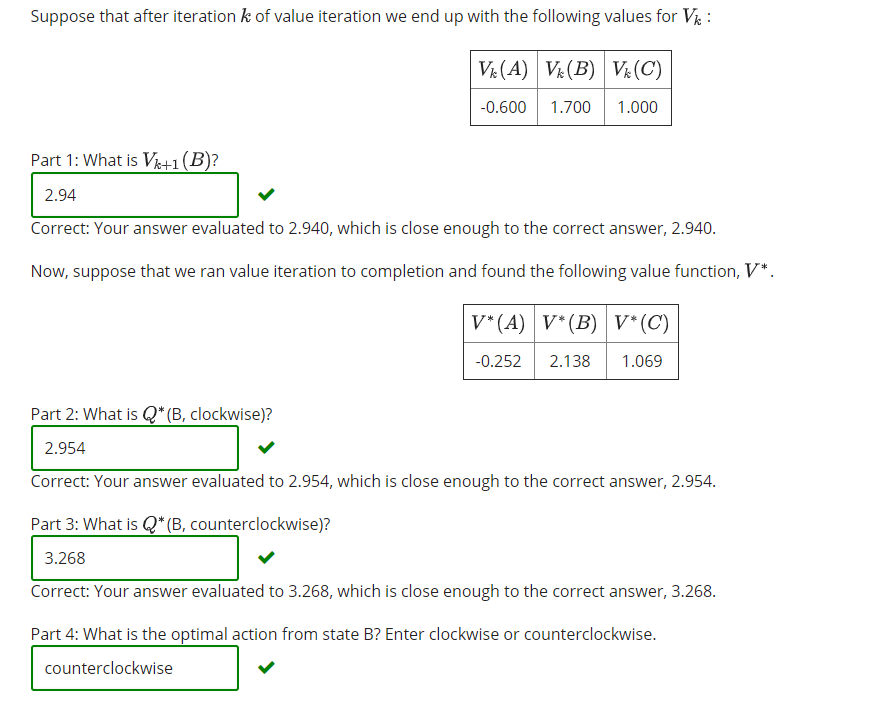
**Question 3: Value Iteration: Cycle**

0.0/16.0 points

*We recommend you work out the solutions to the following questions on a sheet of scratch paper, and then enter your results into the answer boxes.*   
Consider the following transition diagram, transition function and reward function for an MDP.

NOTE: This will change numbers with every reset. Do a screen print of your correct answers and paste them on this sheet.



**Question 4: Value Iteration Properties**

0.0/5.0 points

Which of the following are true about value iteration? We assume the MDP has a finite number of actions and states, and that the discount factor satisfies 0 < γ < 1.

Value iteration is guaranteed to converge.

Value iteration will converge to the same vector of values (V\*) no matter what values we use to initialize V.

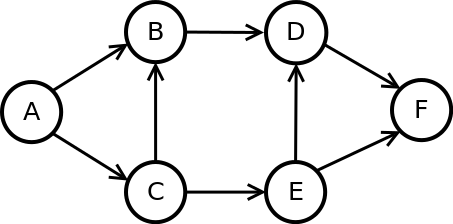
None of the above

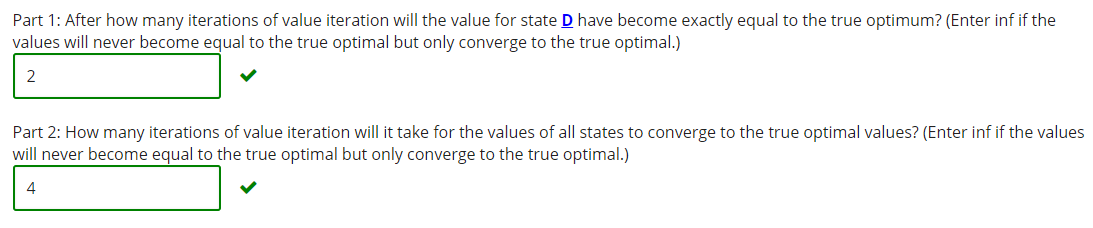
**Question 5.1: Value Iteration Convergence**

0.0/4.0 points

This is a randomized question. The variables that change with each reset of the question are colored **blue**. Since it changes, copy a screen print of your correct answer here.

We will consider a simple MDP that has six states, A, B, C, D, E, and F. Each state has a single action, *go*. An arrow from a state x to a state y indicates that it is possible to transition from state x to next state y when *go* is taken. If there are multiple arrows leaving a state x, transitioning to each of the next states is equally likely. The state F has no outgoing arrows: once you arrive in F, you stay in F for all future times. The reward is one for all transitions, with one exception: staying in F gets a reward of zero. Assume a discount factor = 0.5. We assume that we initialize the value of each state to 0. (Note: you should not need to explicitly run value iteration to solve this problem.)



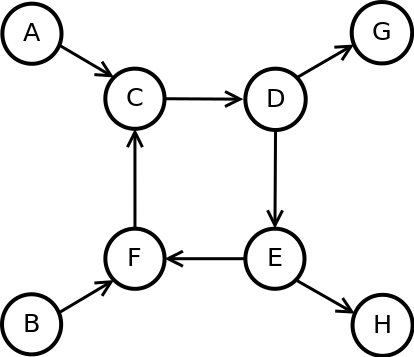


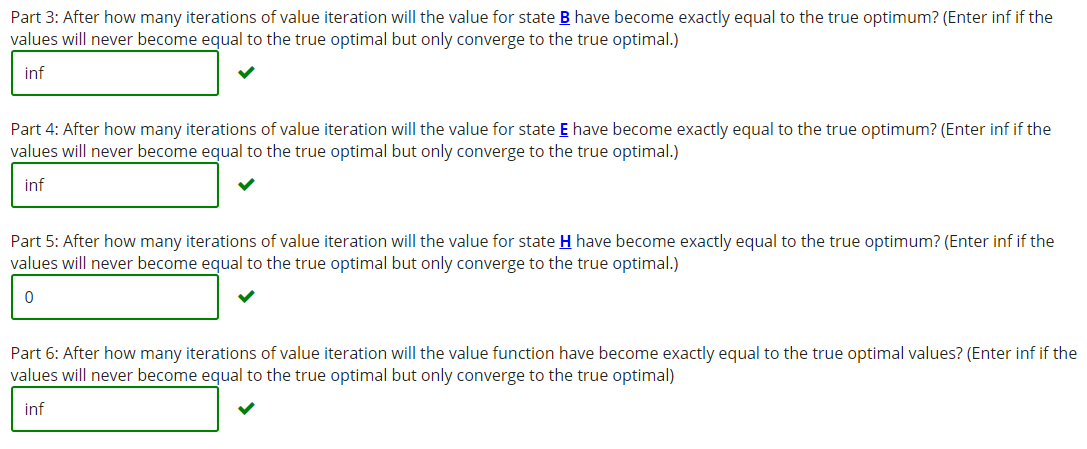
**Question 5.2: Value Iteration Convergence**

0.0/8.0 points

This is a randomized question. The variables that change with each reset of the question are colored **blue**. Since it changes, copy a screen print of your correct answer here.

We will consider a simple MDP that has eight states, A, B, C, D, E, F, G, and H . Each state has a single action, *go*. An arrow from a state x to a state y indicates that it is possible to transition from state x to next state y when is taken. If there are multiple arrows leaving a state x, transitioning to each of the next states is equally likely. The states G and H have no outgoing arrows: once you arrive in G or H, you stay in them for all future times. The reward is one for all transitions, with one exception: staying in G or H gets a reward of zero. Assume a discount factor = 0.5. We assume that we initialize the value of each state to 0. (Note: you should not need to explicitly run value iteration to solve this problem.)





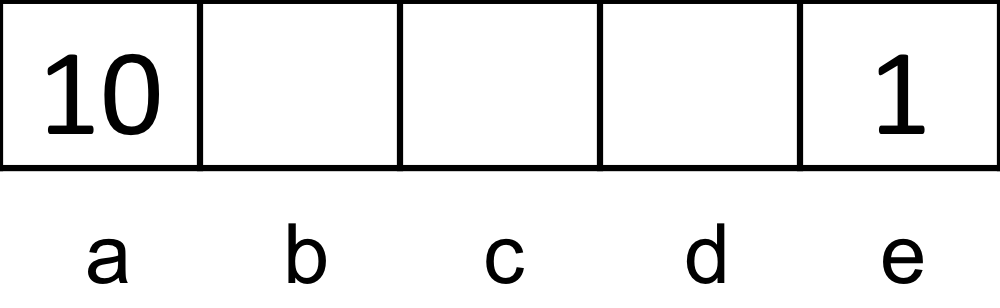
**Question 6: Policy Evaluation**

0.0/10.0 points

Consider the gridworld where Left and Right actions are successful 100% of the time.

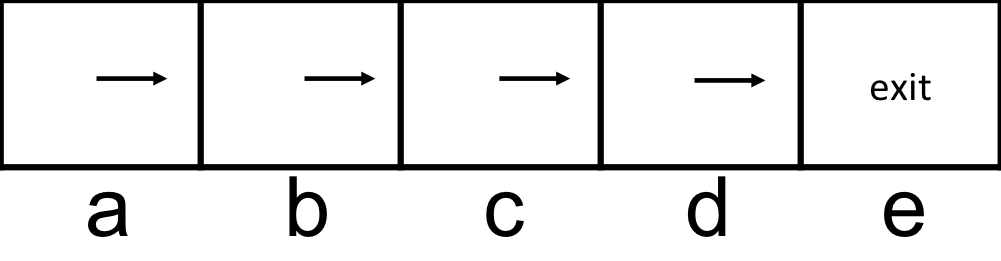
Specifically, the available actions in each state are to move to the neighboring grid squares. From state *a*, there is also an exit action available, which results in going to the terminal state and collecting a reward of 10. Similarly, in state *e*, the reward for the exit action is 1. Exit actions are successful 100% of the time.

The discount factor γ is 1.



**Part 1**

Consider the policy π1 shown below, and evaluate the following quantities for this policy.



Vπ1(a) =



Vπ1(b) =



Vπ1(c) =



Vπ1(d) =

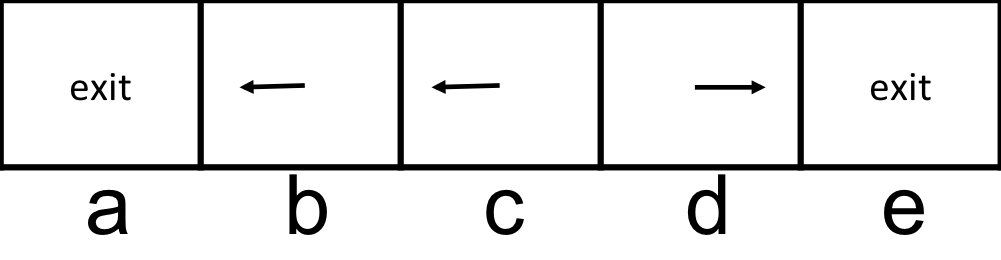


Vπ1(e) =



**Part 2**

Consider the policy π2 shown below, and evaluate the following quantities for this policy.



Vπ2(a) =



Vπ2(b) =



Vπ2(c) =



Vπ2(d) =



Vπ2(e) =



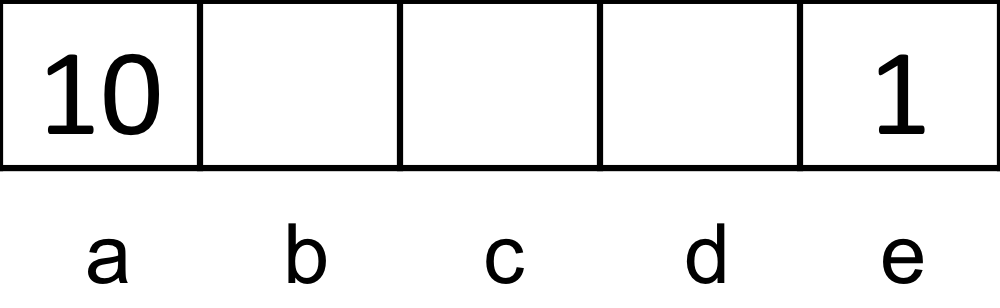
**Question 7: Policy Iteration**

0.0/5.0 points

Consider the gridworld where Left and Right actions are successful 100% of the time.

Specifically, the available actions in each state are to move to the neighboring grid squares. From state *a*, there is also an exit action available, which results in going to the terminal state and collecting a reward of 10. Similarly, in state *e*, the reward for the exit action is 1. Exit actions are successful 100% of the time.

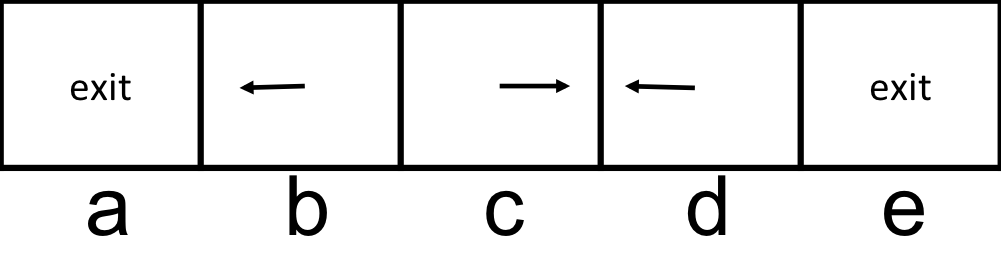
The discount factor (γ) is 0.9.



We will execute one round of policy iteration.

**Part 1: Policy Evaluation**

Consider the policy πi shown below, and evaluate the following quantities for this policy.



Vπi(a) =



Vπi(a) =



Vπi(a) =



Vπi(a) =



Vπi(a) =



**problem**

0.0/5.0 points

**Part 2: Policy Improvement**

Perform a policy improvement step. The current policy's values are the ones from Part 1 (so make sure you first correctly answer Part 1 before moving on to Part 2).

πi+1(a) =

Exit

Right

πi+1(b) =

Left

Right

πi+1(c) =

Left

Right

πi+1(d) =

Left

Right

πi+1(e) =

Left

Exit

**Question 8: Policy Iteration: Cycle**

0.0/16.0 points

*We recommend you work out the solutions to the following questions on a sheet of scratch paper, and then enter your results into the answer boxes.*   
Consider the following transition diagram, transition function and reward function for an MDP.

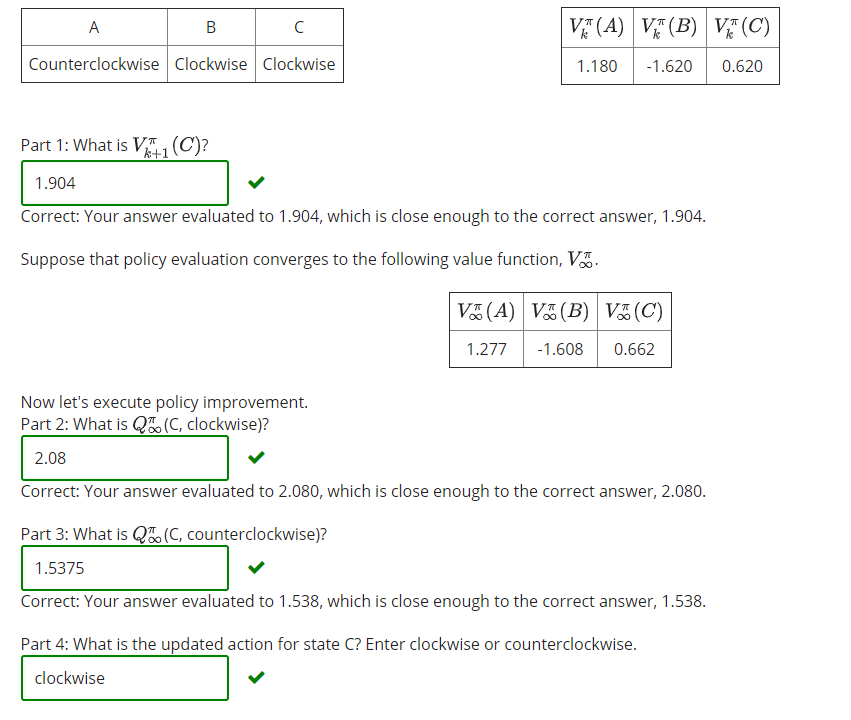
NOTE: This will change numbers with every reset. Do a screen print of your correct answers and paste them on this sheet.

|  |  |
| --- | --- |
| https://prod-edge-static.edx-cdn.org/assets/courseware/v1/63f3ae1c3bb7713dcb655adf46d1c124/c4x/BerkeleyX/CS188x-8/asset/hw4_value_iteration.png |  |
| Discount Factor γ = 0.5   |  |  |  |  |  | | --- | --- | --- | --- | --- | | s | a | s' | T(s,a,s') | R(s,a,s') | | A | Clockwise | B | 0.8 | 0.0 | | A | Clockwise | C | 0.2 | 0.0 | | A | Counterclockwise | B | 0.4 | -1.0 | | A | Counterclockwise | C | 0.6 | -2.0 | | B | Clockwise | A | 0.4 | 1.0 | | B | Clockwise | C | 0.6 | -2.0 | | B | Counterclockwise | A | 1.0 | 2.0 | | C | Clockwise | A | 0.6 | 0.0 | | C | Clockwise | B | 0.4 | -2.0 | | C | Counterclockwise | A | 0.4 | -1.0 | | C | Counterclockwise | B | 0.6 | -1.0 | | |

Suppose we are doing policy evaluation, by following the policy given by the left-hand side table below. Our current estimates (at the end of some iteration of policy evaluation) of the value of states when following the current policy is given in the right-hand side table.

|  |  |
| --- | --- |
|  |  |

|  |
| --- |
|  |



**Question 9: Wrong Discount Factor**

0.0/5.0 points

Bob notices value iteration converges more quickly with smaller γ and rather than using the true discount factor γ, he decides to use a discount factor of αγ with 0 < α < 1when running value iteration. Mark each of the following that are guaranteed to be true:

While Bob will not find the optimal value function, he could simply rescale the values he finds by (1 – γ)/(1 – α) to find the optimal value function.

If the MDP's transition model is deterministic and the MDP has zero rewards everywhere, except for a single transition at the goal with a positive reward, then Bob will still find the optimal policy.

If the MDP's transition model is deterministic, then Bob will still find the optimal policy.

Bob's policy will tend to more heavily favor short-term rewards over long-term rewards compared to the optimal policy.

None of the above.

**Question 10.1: MDP Properties**

0.0/5.0 points

Which of the following statements are true for an MDP?

If the only difference between two MDPs is the value of the discount factor then they must have the same optimal policy.

For an infinite horizon MDP with a finite number of states and actions and with a discount factor γ that satisfies 0 < γ < 1, value iteration is guaranteed to converge.

When running value iteration, if the policy (the greedy policy with respect to the values) has converged, the values must have converged as well.

None of the above

**Question 10.2: MDP Properties Continued**

0.0/5.0 points

Which of the following statements are true for an MDP?

If one is using value iteration and the values have converged, the policy must have converged as well.

Expectimax will generally run in the same amount of time as value iteration on a given MDP.

For an infinite horizon MDP with a finite number of states and actions and with a discount factor γ that satisfies 0 < γ < 1, policy iteration is guaranteed to converge.

None of the above

### Question 11: Policies

0.0/5.0 points

John, James, Alvin and Michael all get to act in an MDP (S, A, T, γ, R, s0).

* John runs value iteration until he finds V\* which satisfies ∀s ∈S:V\*(s) = maxα∈AΣs’T(s, a, s’)(R(s, a, s’) + γV\*(s’)) and acts according to πJohn = argmax α∈AΣs’T(s, a, s’)(R(s, a, s’) + γV\*(s’)) .
* James acts according to an arbitrary policy πJames.
* Alvin takes James's policy πJame and runs one round of policy iteration to find his policy πAlvin.
* Michael takes John's policy and runs one round of policy iteration to find his policy πMichael.

*Note: One round of policy iteration = performing policy evaluation followed by performing policy improvement.* Mark all of the following that are guaranteed to be true:

It is guaranteed that ∀s ∈S:VπJames(s) ≥ VπAlvin(s)

It is guaranteed that ∀s ∈S:VπMichael(s) ≥ VπAlvin(s)

It is guaranteed that ∀s ∈S:VπMichael(s) ≥ VπJohn(s)

It is guaranteed that ∀s ∈S:VπJames(s) ≥ VπJohn(s)

None of the above.