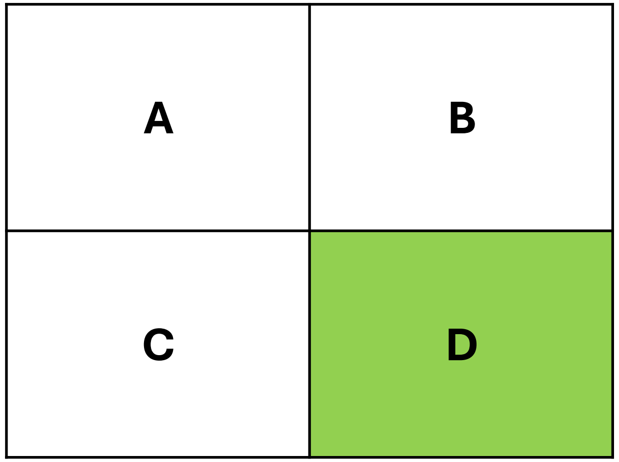
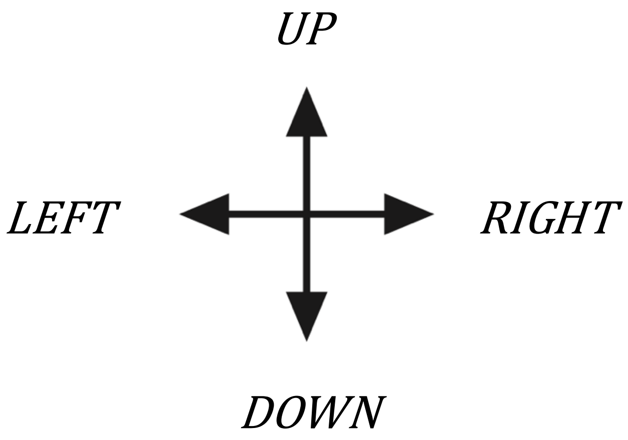
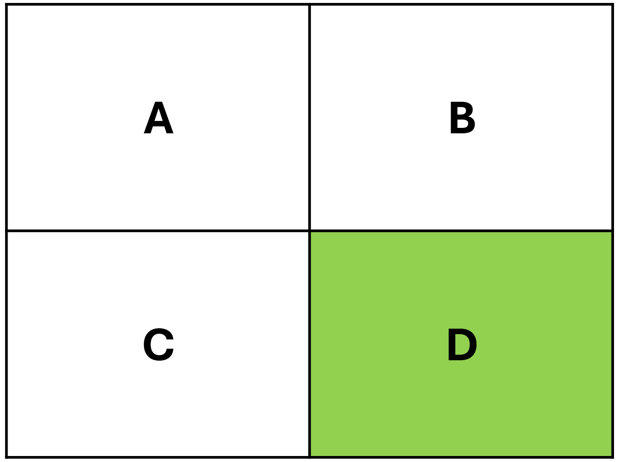
Example: 2x2 Gridworld

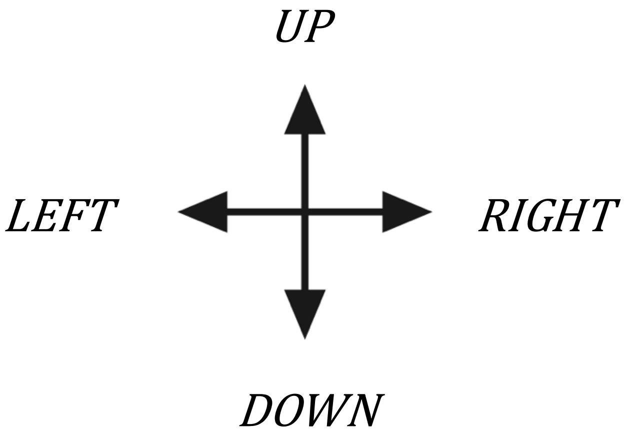




- **States** $\mathcal{S} = (A, B, C, D)$
- **Actions** $\mathcal{A} = (UP, DOWN, LEFT, RIGHT)$
- **Policy** $\mathcal{P}$ = From every state, choose each action with probability 0.25
- **Reward** $(\mathcal{R} = -1)$ $per\ step$
- Discount Factor $(\gamma = 1)$

Example: 2x2 Gridworld

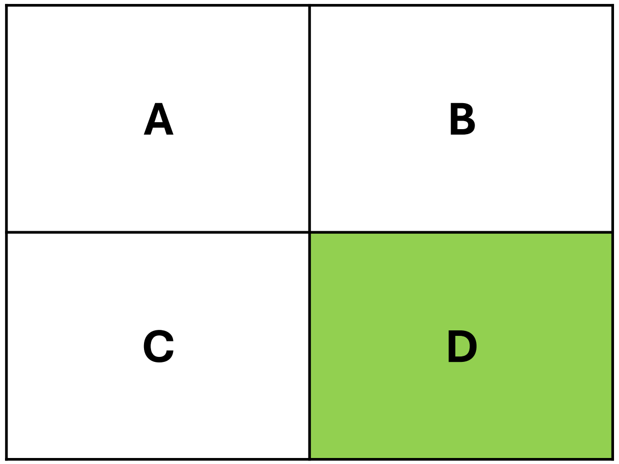


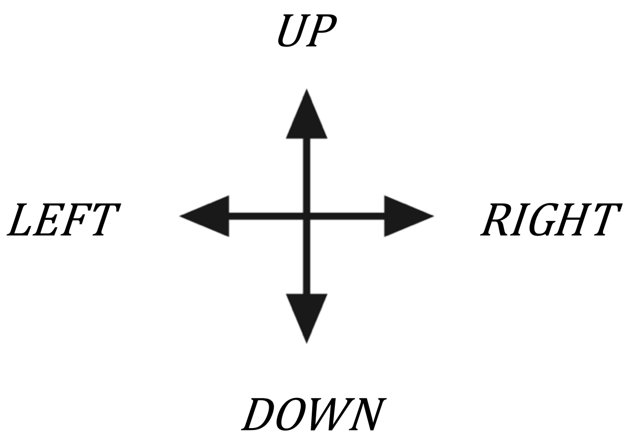


- Undiscounted MDP ($\gamma = 1$)
- Non-terminal states (A,B,C)
- Terminal State (D)
- Agent follows a uniform random policy

$$\pi(\text{up} \mid \cdot\,) = \pi(\text{left} \mid \cdot\,) = \pi(\text{down} \mid \cdot\,) = \pi(\text{right} \mid \cdot\,) = 0.25$$

Example: 2x2 Gridworld





Rules:

- From each state, actions move you in that direction if possible, otherwise you stay in the same square.
- Reward is -1 until the terminal state is reached.
- The goal is to reach state D which gives **0 reward** and ends the episode.

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0		
B	0		
C	0		

0	0
0	

$k = 0$

Initially, we set the value functions $v_k(s)$ of all states to **zero**

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0		
B	0		
C	0		

0	0
0	

$\boxed{k = 0}$

We then use the **Bellman equation** to update the value function $v_k(s)$ of all states at each k iteration.

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0		
B	0		
C	0		

0	**0**
0	

$k = 0$

$$v_{k+1}(s) = \sum_a \pi(a \mid s)[r(s,a) + \gamma v_k(s')]$$

$$v_{k+1}(A) = \frac{1}{4}[(-1 + v(A)) + (-1 + v(B)) + (-1 + v(C)) + (-1 + v(A))$$

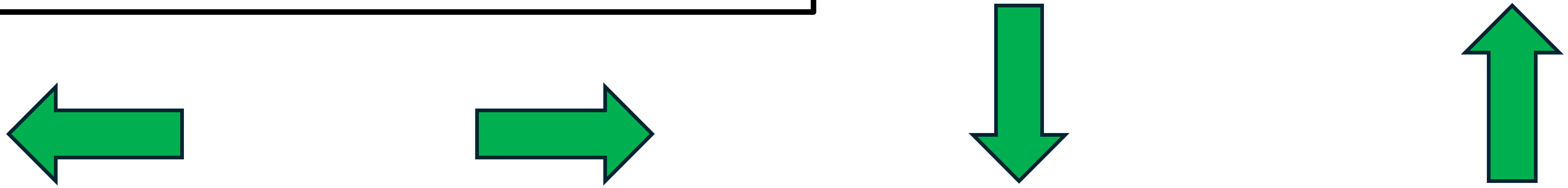
Using the Bellman equation, we get this

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0		
B	0		
C	0		

0	0
0	

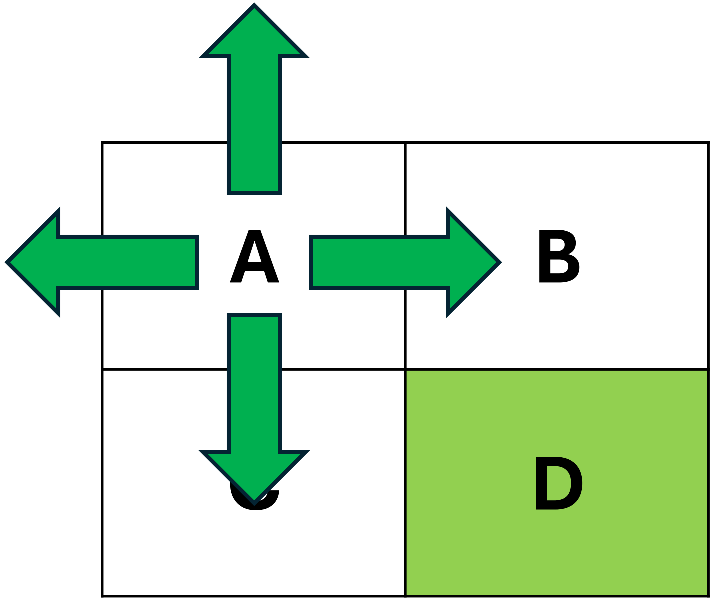
$\boxed{k = 0}$

$$\boxed{v_{k+1}(s) = \sum_a \pi(a \mid s)[r(s,a) + \gamma v_k(s')]}$$



$$v_{k+1}(A) = \frac{1}{4}[(-1 + v(A)) + (-1 + v(B)) + (-1 + v(C)) + (-1 + v(A))$$

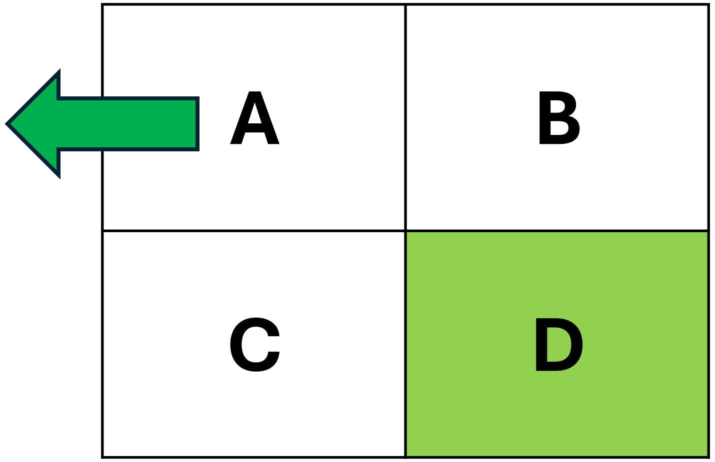
Let us this break down. To update the value function of **state A**, we get the sum of the immediate reward plus the value function of the next state for all possible actions from state A.



0	0
0	

$$v_{k+1}(A) = \frac{1}{4}[(-1 + v(A)) + (-1 + v(B)) + (-1 + v(C)) + (-1 + v(A))$$

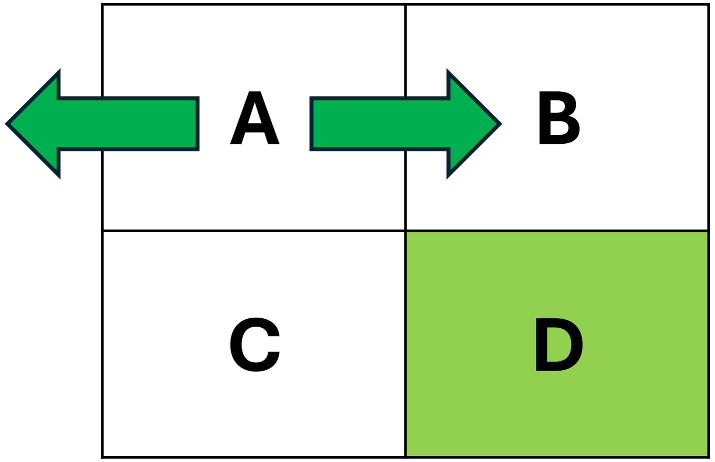
Let us this break down. To update the value function of **state A**, we get the sum of the immediate reward plus the value function of the next state for all possible actions from **state A**.



0	0
0	

$$v_{k+1}(A) = \frac{1}{4}[(\underbrace{-\mathbf{1} + \boldsymbol{v}(\boldsymbol{A})}) + (-1 + v(B)) + (-1 + v(C)) + (-1 + v(A))$$

First term: When we go **LEFT**, we stay in **state A** and we will get a reward of -1 plus the value function of **state A**.

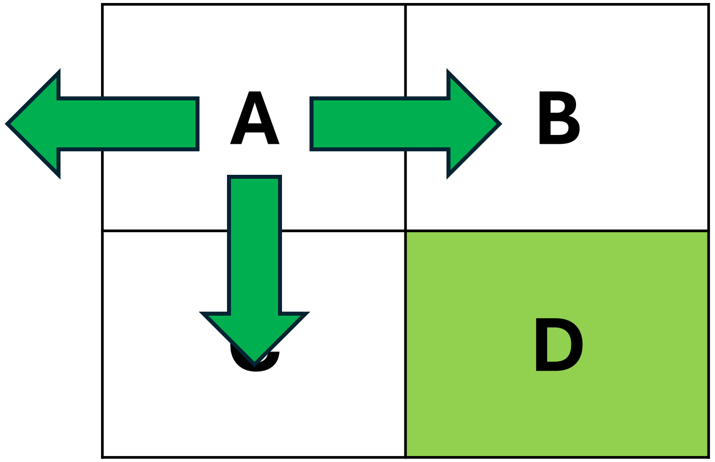


0	0
0	

$$v_{k+1}(A) = \frac{1}{4}[(-1 + v(A)) + (\underbrace{\mathbf{-1} + \boldsymbol{v(B)}}) + (-1 + v(C)) + (-1 + v(A))$$

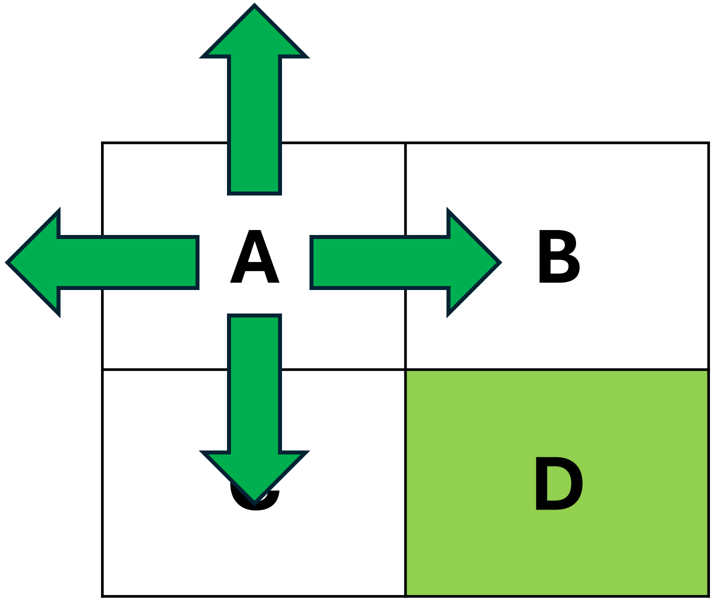
Second term: When we go **RIGHT**, we go to **state B** and we will get a reward of -1 plus the value function of **state B**.

0	0
0	



$$v_{k+1}(A) = \frac{1}{4}[(-1 + v(A)) + (-1 + v(B)) + (\underbrace{\boldsymbol{-1 + v(C)}}) + (-1 + v(A))$$

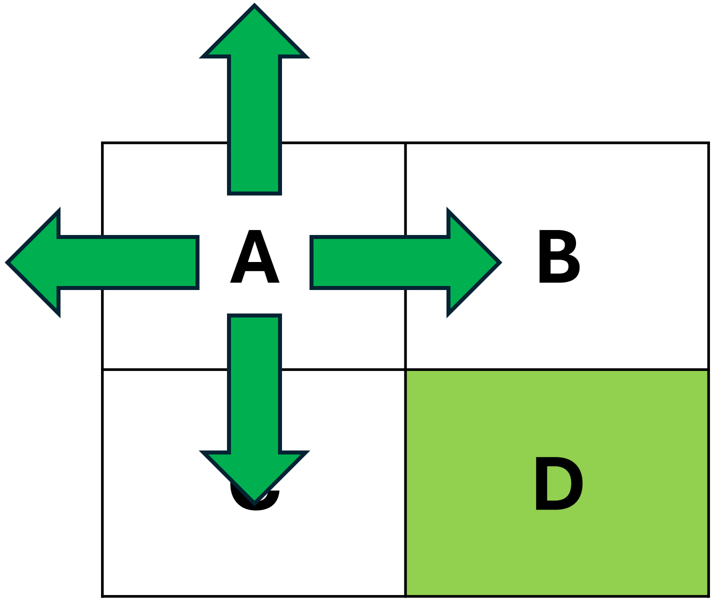
Third term: When we go **DOWN**, we go to **state C** and we will get a reward of -1 plus the value function of **state A**.



0	0
0	

$$v_{k+1}(A) = \frac{1}{4}[(-1 + v(A)) + (-1 + v(B)) + (-1 + v(C)) + (\underbrace{\boldsymbol{-1 + v(A)}})$$

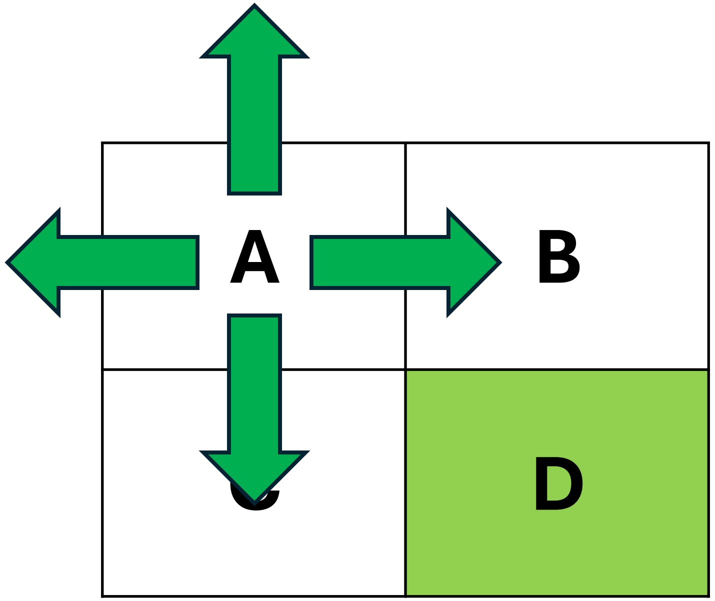
Fourth term: When we go **UP**, we stay in **state A** and we will get a reward of -1 plus the value function of **state A**.

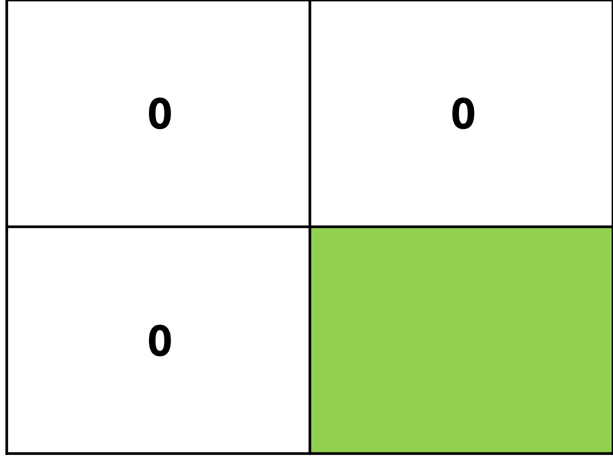


0	0
0	

$$v_{k+1}(A) = \frac{1}{4}\left[(-1 + v(A)) + (-1 + v(B)) + (-1 + v(C)) + (-1 + v(A))\right]$$

One fourth is the uniform random **policy** for all actions. The agent follows this policy of having a 25% probability of choosing **action $\mathcal{A}$**.





$$v_{k+1}(A) = \frac{1}{4}[(-1 + v(A)) + (-1 + v(B)) + (-1 + v(C)) + (-1 + v(A))$$

The discount factor γ is omitted because we set it to 1. If we have a different value for the discount factor, we would multiply it to each value function of the next state.

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0		
B	0		
C	0		

0	0
0	

$k = 1$

$$v_{k+1}(s) = \sum_a \pi(a \mid s)[r(s,a) + \gamma v_k(s')]$$

$$v_{k+1}(A) = \frac{1}{4}[(-1 + v(A)) + (-1 + v(B)) + (-1 + v(C)) + (-1 + v(A))$$

$$v_{k+1}\ (A) = \frac{1}{4}[(-1 + 0) + (-1 + 0) + (-1 + 0) + (-1 + 0)$$

$$\boxed{v_{k+1}(A) = -1}$$

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0	-1	
B	0		
C	0		

-1	0
0	

$\boxed{k = 1}$

$$v_{k+1}(A) = \frac{1}{4}[(-1 + v(A)) + (-1 + v(B)) + (-1 + v(C)) + (-1 + v(A))$$

$$v_{k+1}\ (A) = \frac{1}{4}[(-1 + 0) + (-1 + 0) + (-1 + 0) + (-1 + 0)$$

$$\boxed{v_{k+1}(A) = -1}$$

We now store the new value function of **state A**

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0	-1	
B	0		
C	0		

-1	0
0	

$\boxed{k = 1}$

$$v_{k+1}(B) = \frac{1}{4}[(-1 + v(A)) + (-1 + v(B)) + (-1 + v(D)) + (-1 + v(B))$$

$$v_{k+1}\,(B) = \frac{1}{4}[(-1 + 0) + (-1 + 0) + (-1 + 0) + (-1 + 0)$$

$$\boxed{v_{k+1}(B) = -1}$$

We do the same process for **state B**

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0	-1	
B	0	-1	
C	0		

-1	-1
0	

$k = 1$

$$v_{k+1}(C) = \frac{1}{4}[(-1 + v(C)) + (-1 + v(D)) + (-1 + v(C)) + (-1 + v(A))$$

$$v_{k+1}\ (C) = \frac{1}{4}[(-1 + 0) + (-1 + 0) + (-1 + 0) + (-1 + 0)$$

$$\boxed{v_{k+1}(C) = -1}$$

And **state C**

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0	-1	
B	0	-1	
C	0	-1	

-1	-1
-1	

$k = 1$

$$v_{k+1}(C) = \frac{1}{4}[(-1 + v(C)) + (-1 + v(D)) + (-1 + v(C)) + (-1 + v(A))$$

$$v_{k+1}\,(C) = \frac{1}{4}[(-1 + 0) + (-1 + 0) + (-1 + 0) + (-1 + 0)$$

$$\boxed{v_{k+1}(C) = -1}$$

And **state C**

0	**0**
0	**0**

$k = 0$

-1	**-1**
-1	**0**

$k = 1$

After one iteration, the value functions for **states A, B** and **C** were changed from -1 to 0.

-1	-1
-1	0

$k = 1$

$$\boxed{q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]}$$

We now use the Bellman equation to compute the action-value functions for each state to improve our existing **policy**.

-1	-1
-1	0

$k = 1$

$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

Let us start computing the action-value functions of **state A**

-1	-1
-1	0

$k = 1$

$$q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]$$

$$q(A, LEFT) = -1 + v(A)$$

$$q(A, LEFT) == -1 + (-1)$$

$$\boxed{q(A, LEFT) = -2}$$

-1	**-1**
-1	**0**

$k = 1$

$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

$$q(A, RIGHT) = -1 + v(B)$$

$$q(A, RIGHT) = = -1 + (-1)$$

$$\boxed{q(A, RIGHT) = -2}$$

-1	**-1**
-1	**0**

$k = 1$

$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

$$q(A, DOWN) = -1 + v(C)$$

$$q(A, DOWN) == -1 + (-1)$$

$$\boxed{q(A, DOWN) = -2}$$

-1	**-1**
-1	**0**

$k = 1$

$$\boxed{q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]}$$

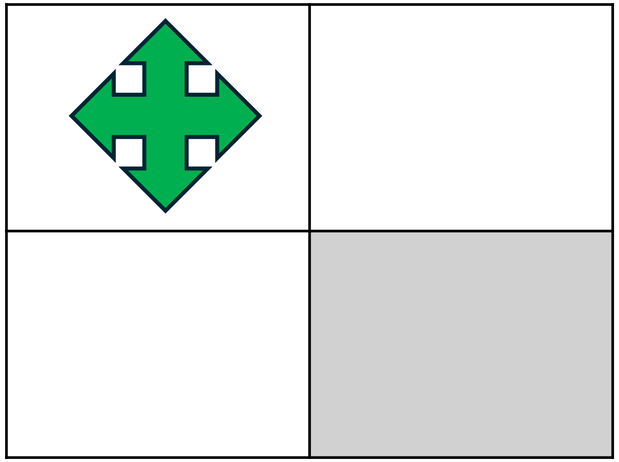
$$q(A, UP) = -1 + v(A)$$

$$q(A, UP) = = -1 + (-1)$$

$$\boxed{q(A, UP) = -2}$$

$$q(A, UP) = -2$$

$$q(A, LEFT) = -2$$



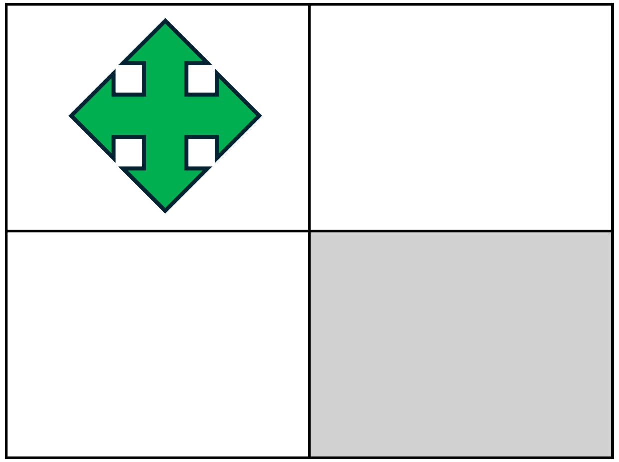
$$q(A, RIGHT) = -2$$

$$q(A, DOWN) = -2$$

We can now map the value for each **action** from **state A**

$$q(A, UP) = -2$$

$$q(A, LEFT) = -2$$



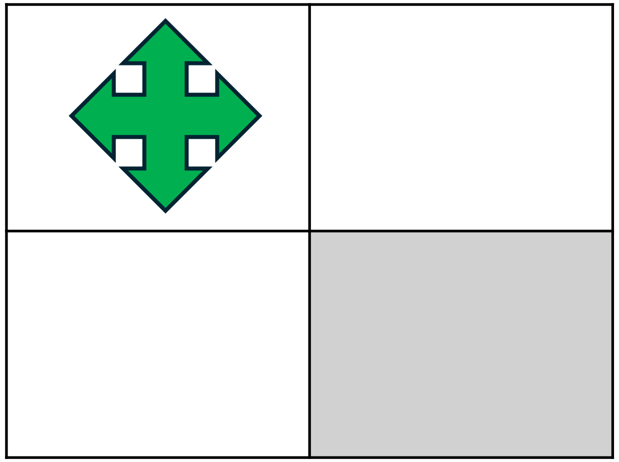
$$q(A, RIGHT) = -2$$

$$q(A, DOWN) = -2$$

This simply tells us that from **state A**, there is no best action to take because all of their values are the same.

$$q(A, UP) = -2$$

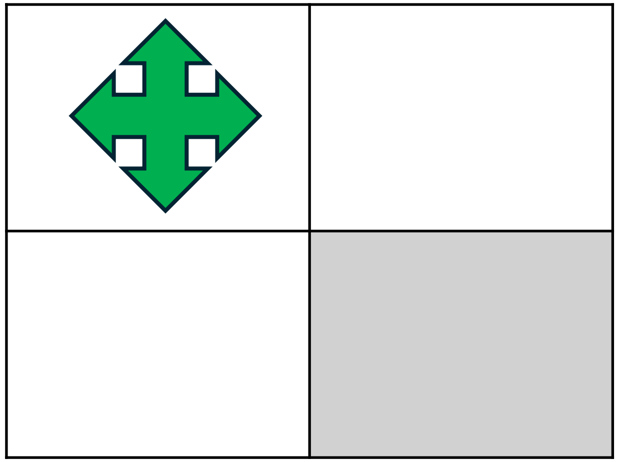
$$q(A, LEFT) = -2$$



$$q(A, RIGHT) = -2$$

$$q(A, DOWN) = -2$$

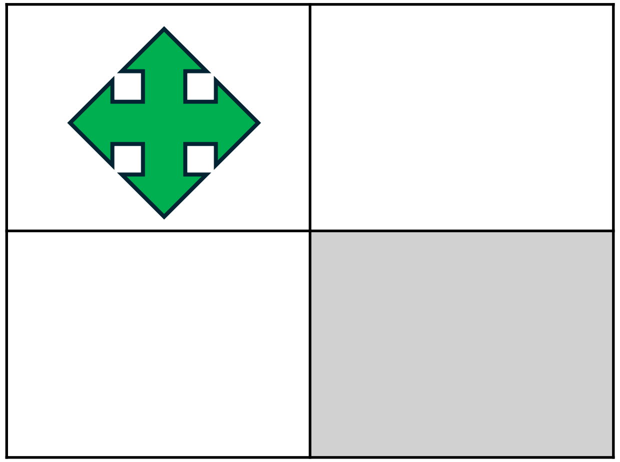
Because all of the values are the same, the **policy** will also be the same for **state A**



Let us now calculate the action-value functions for **state B**

-1	-1
-1	0

$k = 1$

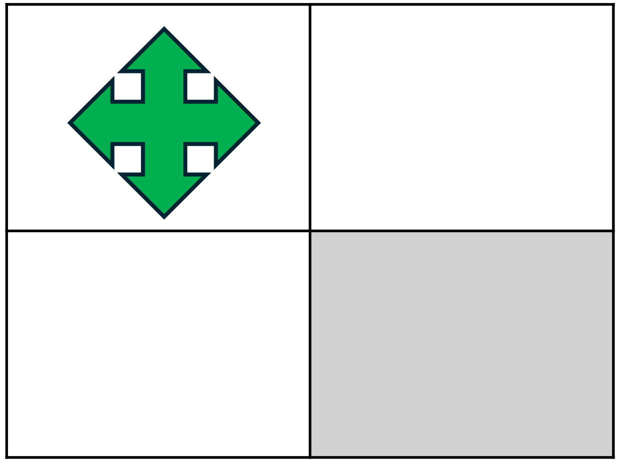


$$q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]$$

Similar to **state A**, we use the Bellman equation to compute the action-value functions for **state B.**

-1	-1
-1	0

$k = 1$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

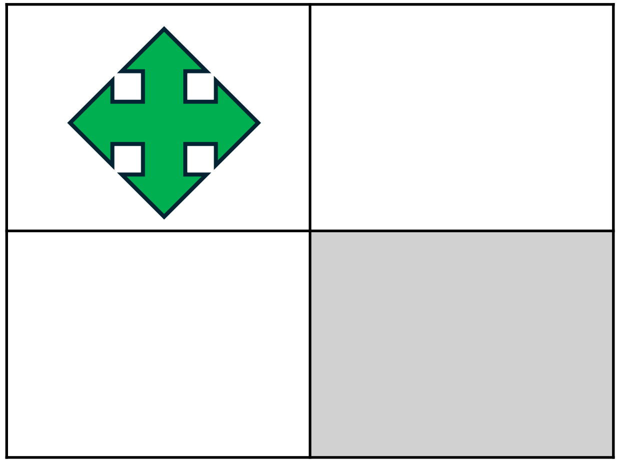
$$q(B, LEFT) = -1 + v(A)$$

$$q(B, LEFT) = -1 + (-1)$$

$$\boxed{q(B, LEFT) = -2}$$

-1	-1
-1	0

$k = 1$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

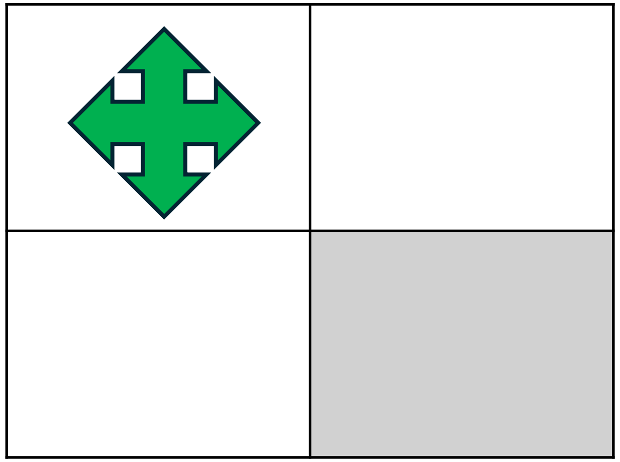
$$q(B, RIGHT) = -1 + v(B)$$

$$q(B, RIGHT) = -1 + (-1)$$

$$\boxed{q(B, RIGHT) = -2}$$

-1	-1
-1	0

$k = 1$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

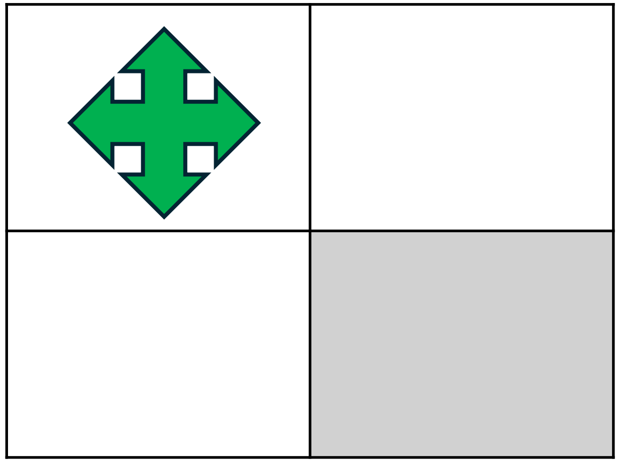
$$q(B, DOWN) = -1 + v(D)$$

$$q(B, DOWN) = -1 + (0)$$

$$\boxed{q(B, DOWN) = -1}$$

-1	**-1**
-1	**0**

$k = 1$



$$\boxed{q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]}$$

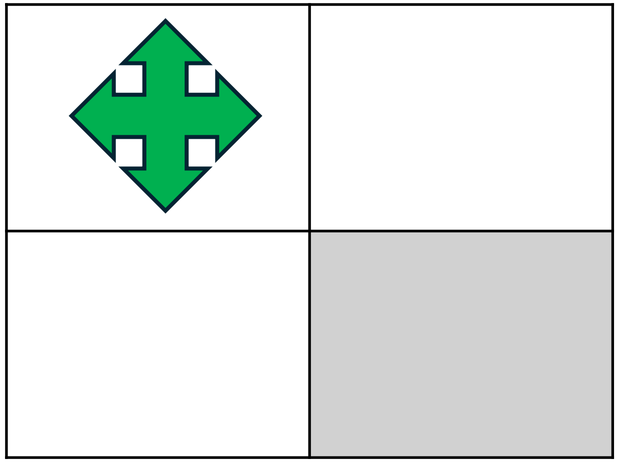
$$q(B, UP) = -1 + v(B)$$

$$q(B, UP) = -1 + (-1)$$

$$\boxed{q(B, UP) = -2}$$

$$\boxed{q(B, UP) = -2}$$

$$\boxed{q(B, LEFT) = -2}$$



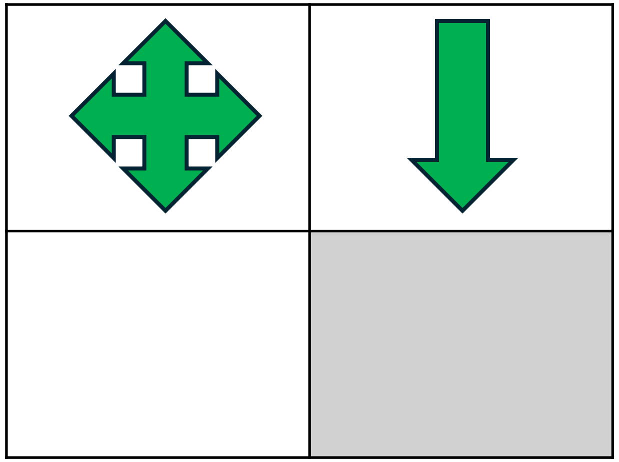
$$\boxed{q(B, RIGHT) = -2}$$

$$\boxed{q(B, DOWN) = -1}$$

We can now map the value for each **action** from **state B**

$$q(B, UP) = -2$$

$$q(B, LEFT) = -2$$



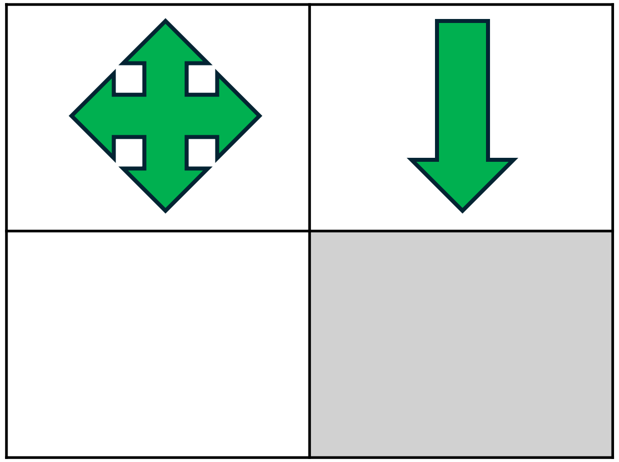
$$q(B, RIGHT) = -2$$

$$\boxed{q(B, DOWN) = -1}$$

This simply tells us that from **state B**, the best action to take is to go **DOWN** because going down has the highest value of all actions.

$$q(B, UP) = -2$$

$$q(B, LEFT) = -2$$



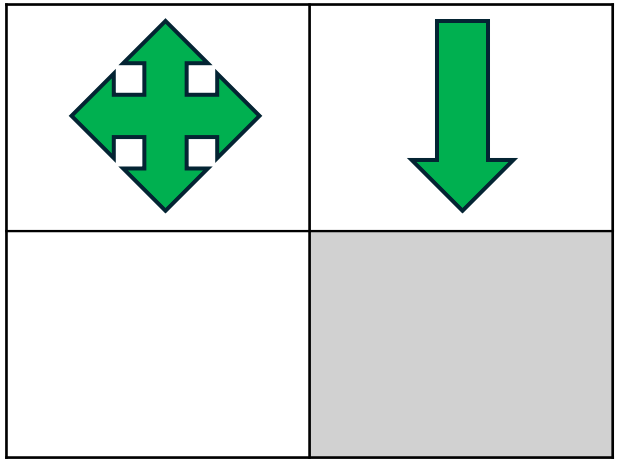
$$q(B, RIGHT) = -2$$

$$\boxed{q(B, DOWN) = -1}$$

Which makes sense because going **DOWN** will go to our goal which is **state D.**

$$q(B, UP) = -2$$

$$q(B, LEFT) = -2$$

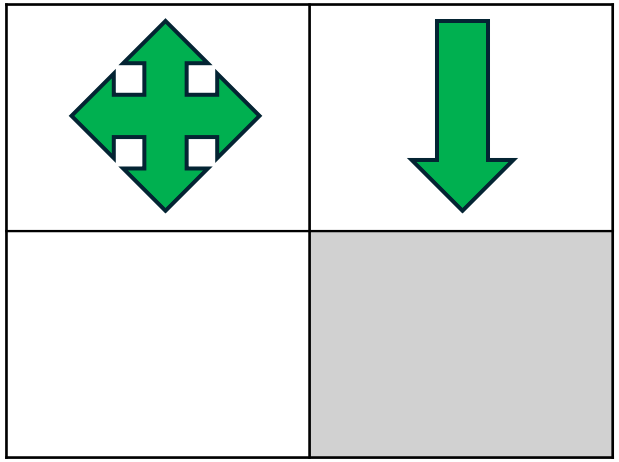


$$q(B, RIGHT) = -2$$

$$\boxed{q(B, DOWN) = -1}$$

Because of this, we can now update the policy for **state B.**

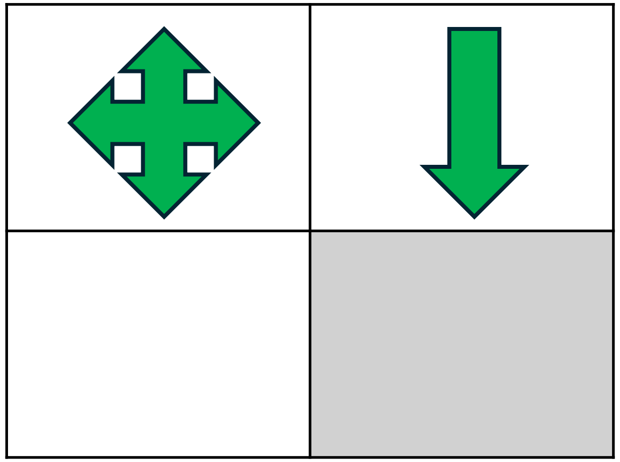
$$\boxed{\pi(B) = \{DOWN\}}$$



Let us now calculate the action-value functions for **state C**

-1	-1
-1	0

$k = 1$

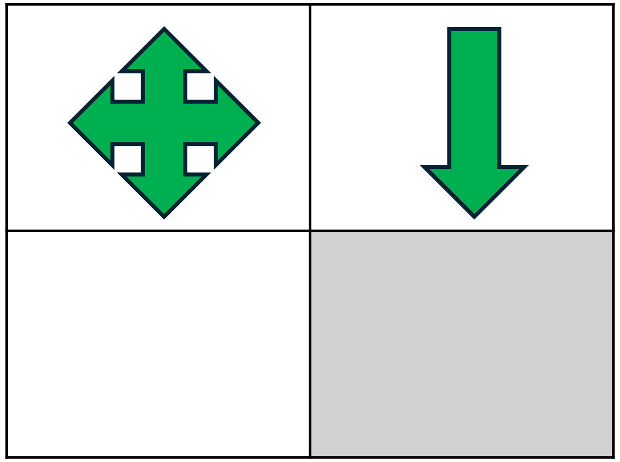


$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

Similar to **state A** and **B**, we use the Bellman equation to compute the action-value functions for **state C.**

-1	-1
-1	0

$k = 1$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

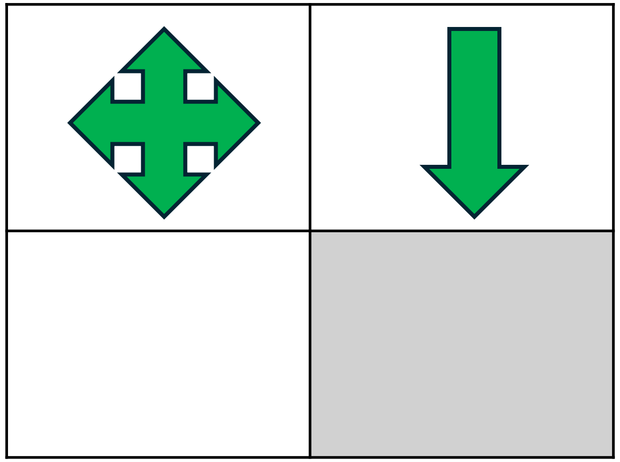
$$q(C, LEFT) = -1 + v(C)$$

$$q(C, LEFT) = -1 + (-1)$$

$$\boxed{q(C, LEFT) = -2}$$

-1	-1
-1	0

$k = 1$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

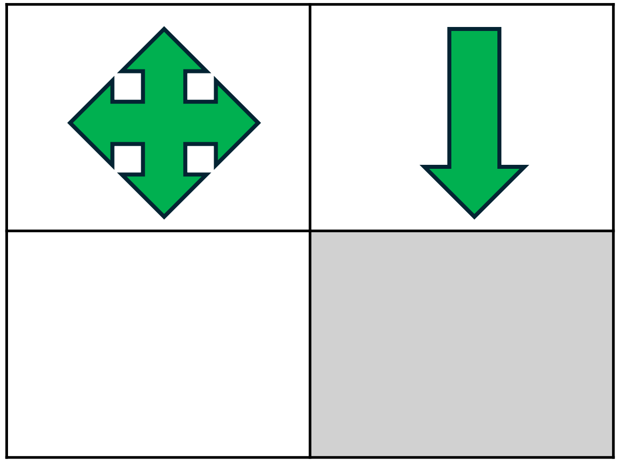
$$q(C, RIGHT) = -1 + v(D)$$

$$q(C, RIGHT) = -1 + (0)$$

$$\boxed{q(C, RIGHT) = -1}$$

-1	-1
-1	0

$k = 1$



$$\boxed{q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]}$$

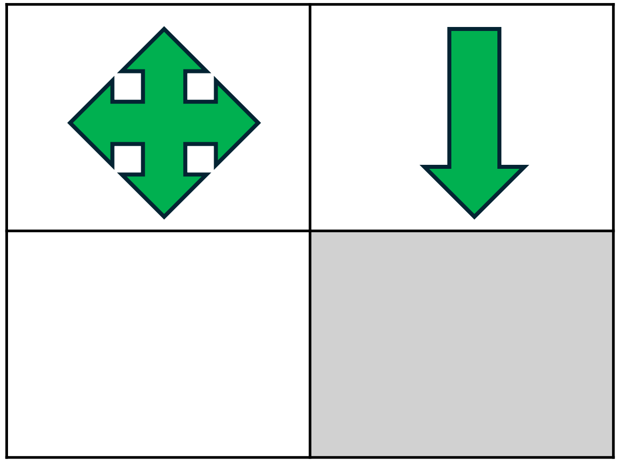
$$q(C, DOWN) = -1 + v(C)$$

$$q(C, DOWN) = -1 + (-1)$$

$$\boxed{q(C, DOWN) = -2}$$

-1	-1
-1	0

$k = 1$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

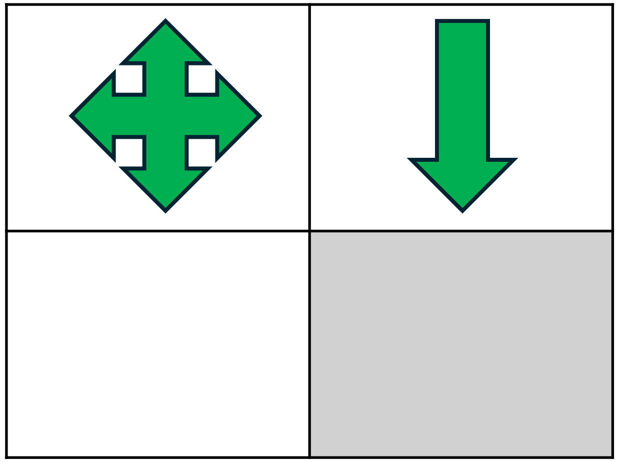
$$q(C,UP) = -1 + v(A)$$

$$q(C,UP) = -1 + (-1)$$

$$\boxed{q(C,UP) = -2}$$

$$q(C, UP) = -2$$

$$q(C, LEFT) = -2$$



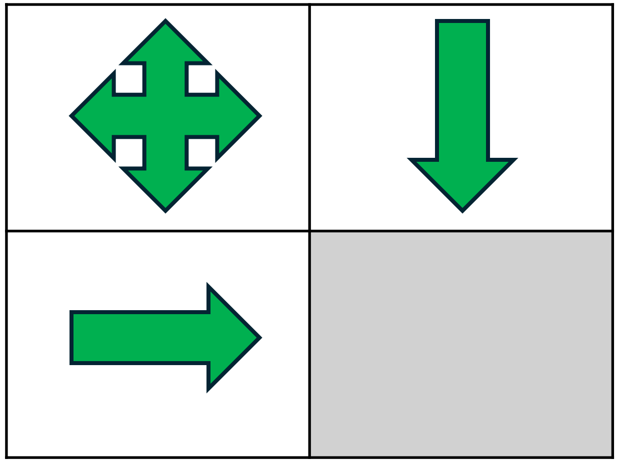
$$q(C, RIGHT) = -1$$

$$q(C, DOWN) = -2$$

We can now map the value for each **action** from **state C**

$$q(C, UP) = -2$$

$$q(C, LEFT) = -2$$



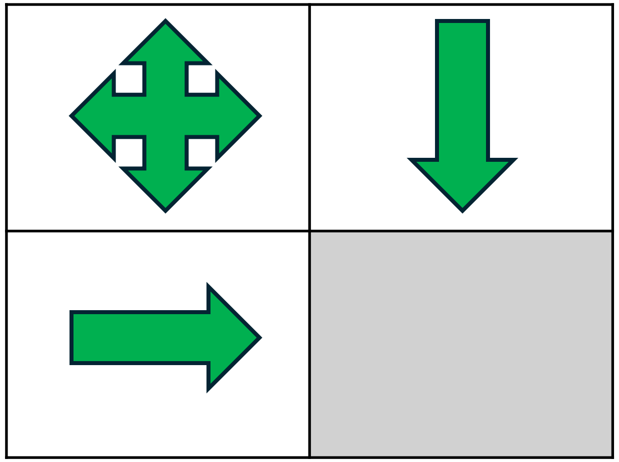
$$\boxed{q(C, RIGHT) = -1}$$

$$q(C, DOWN) = -2$$

This simply tells us that from **state C**, the best action to take is to go **RIGHT** because going right has the highest value of all actions.

$$q(C, UP) = -2$$

$$q(C, LEFT) = -2$$



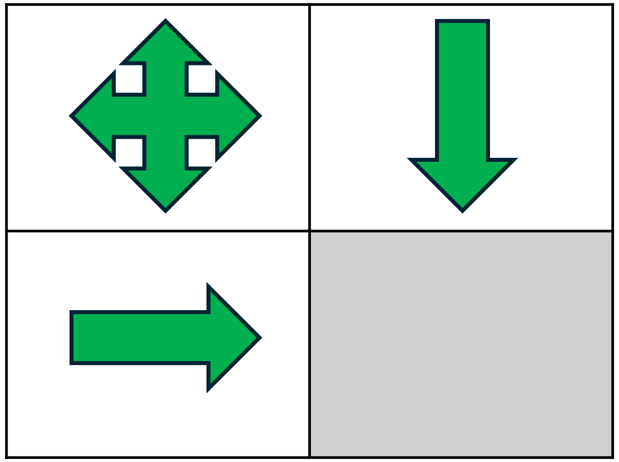
$$\boxed{q(C, RIGHT) = -1}$$

$$q(C, DOWN) = -2$$

Which makes sense because going **RIGHT** will go to our goal which is **state D.**

$$q(C, UP) = -2$$

$$q(C, LEFT) = -2$$



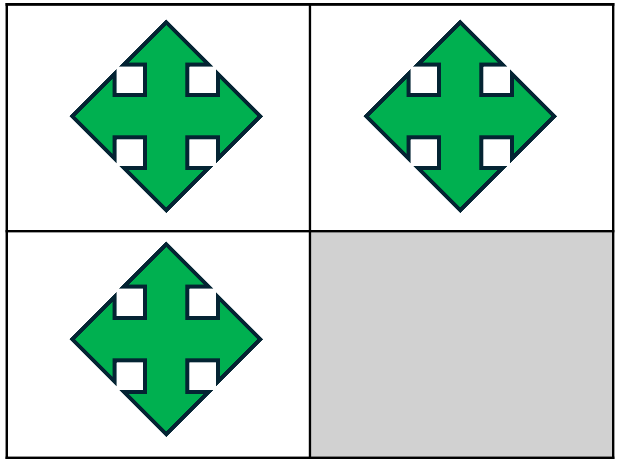
$$\boxed{q(C, RIGHT) = -2}$$

$$q(C, DOWN) = -1$$

Just like before, we can now update our policy for **state C.**

$$\boxed{\pi(C) = \{RIGHT\}}$$

$\boxed{k = 0}$



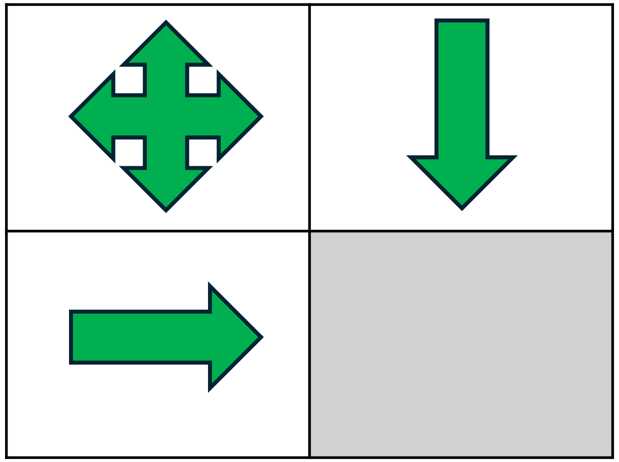
$\pi(A) = \{LEFT, RIGHT, UP, DOWN\}$

$\pi(B) = \{LEFT, RIGHT, UP, DOWN\}$

$\pi(C) = \{LEFT, RIGHT, UP, DOWN\}$

Initially, we started with this **uniform random policy**

$$\boxed{k = 1}$$



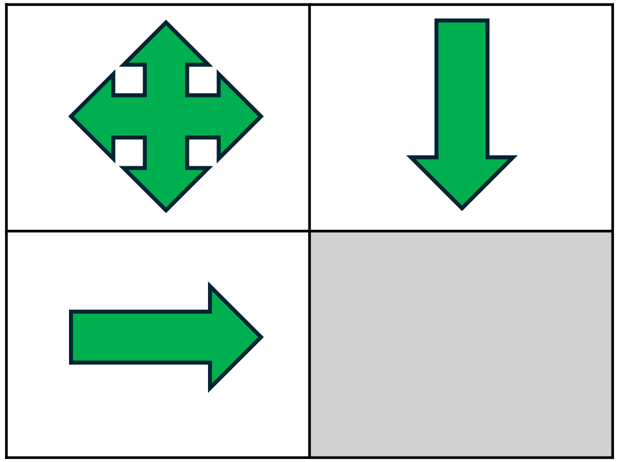
$$\pi(A) = \{LEFT, RIGHT, UP, DOWN\}$$

$$\pi(B) = \{DOWN\}$$

$$\pi(C) = \{RIGHT\}$$

After 1 iteration, we improved our **previous policy**.

$$\boxed{k = 1}$$



$$\pi(A) = \{LEFT, RIGHT, UP, DOWN\}$$

$$\pi(B) = \{DOWN\}$$

$$\pi(C) = \{RIGHT\}$$

Let us do one last iteration if we can reach convergence, or in RL terms, find the **optimal policy π_***

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0	-1	-2
B	0	-1	
C	0	-1	

-2	-1
-1	

$k = 1$

$$v_{k+2}(s) = \sum_a \pi(a \mid s)[r(s,a) + \gamma v_{k+1}(s')]$$

$$v_{k+2}(A) = \frac{1}{4}[(-1 + v_{k+1}(A)) + (-1 + v_{k+1}(B)) + (-1 + v_{k+1}(C)) + (-1 + v_{k+1}(A))$$

$$v_{k+2}\,(A) = \frac{1}{4}[(-1 - 1) + (-1 - 1) + (-1 - 1) + (-1 - 1)$$

$$\boxed{v_{k+2}(A) = -2}$$

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0	-1	-2
B	0	-1	-1.75
C	0	-1	

-2	-1.75
-1	

$k = 1$

$$\boxed{v_{k+2}(s) = \sum_a \pi(a \mid s)[r(s,a) + \gamma v_{k+1}(s')]}$$

$$v_{k+2}(B) = \frac{1}{4}[(-1 + v_{k+1}(A)) + (-1 + v_{k+1}(B)) + (-1 + v_{k+1}(D)) + (-1 + v_{k+1}(B))$$

$$v_{k+2}\,(B) = \frac{1}{4}[(-1 - 1) + (-1 - 1) + (-1 + 0) + (-1 - 1)$$

$$\boxed{v_{k+2}(B) = -1.75}$$

	$v_k(s)$	$v_{k+1}(s)$	$v_{k+2}(s)$
A	0	-1	-2
B	0	-1	-1.75
C	0	-1	-1.75

-2	-1.75
-1.75	

$k = 1$

$$\boxed{v_{k+2}(s) = \sum_a \pi(a \mid s)[r(s,a) + \gamma v_{k+1}(s')]}$$

$$v_{k+2}(C) = \frac{1}{4}[(-1 + v_{k+1}(C)) + (-1 + v_{k+1}(D)) + (-1 + v_{k+1}(C)) + (-1 + v_{k+1}(A))$$

$$v_{k+2}\,(C) = \frac{1}{4}[(-1 - 1) + (-1 + 0) + (-1) + (-1 - 1)$$

$$\boxed{v_{k+2}(C) = -1.75}$$

-2	**-1.75**
-1.75	**0**

$k = 2$

$$\boxed{q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]}$$

$$q(A, LEFT) = -1 + v(A)$$

$$q(A, LEFT) = = -1 + (-2)$$

$$\boxed{q(A, LEFT) = -3}$$

-2	**-1.75**
-1.75	**0**

$k = 2$

$$\boxed{q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]}$$

$$q(A, RIGHT) = -1 + v(B)$$

$$q(A, RIGHT) = = -1 + (-1.75)$$

$$\boxed{q(A, RIGHT) = -2.75}$$

-2	**-1.75**
-1.75	**0**

$k = 2$

$$\boxed{q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]}$$

$$q(A, DOWN) = -1 + v(C)$$

$$q(A, DOWN) = = -1 + (-1.75)$$

$$\boxed{q(A, DOWN) = -2.75}$$

-2	-1.75
-1.75	0

$k = 2$

$$q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]$$

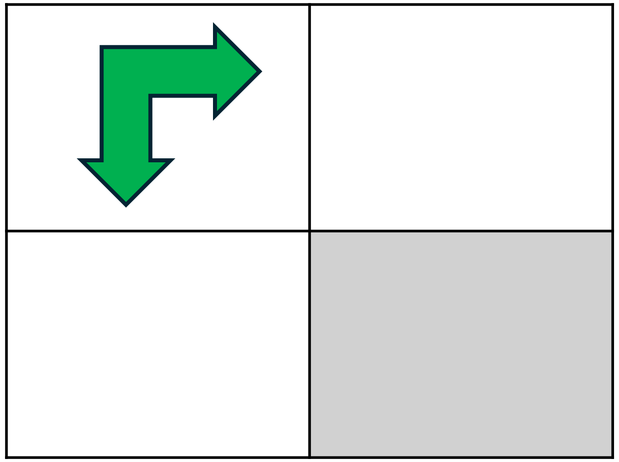
$$q(A, UP) = -1 + v(A)$$

$$q(A, UP) = = -1 + (-2)$$

$$\boxed{q(A, UP) = -3}$$

$$q(A, UP) = -3$$

$$q(A, LEFT) = -3$$



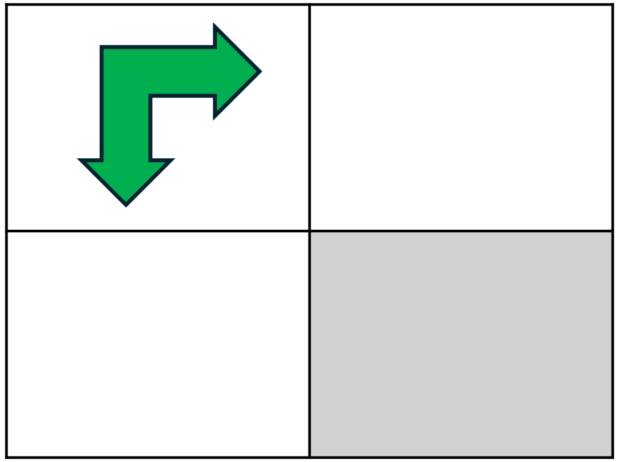
$$q(A, RIGHT) = -2.75$$

$$q(A, DOWN) = -2.75$$

We can now map the value for each **action** from **state A**

$$q(A, UP) = -3$$

$$q(A, LEFT) = -3$$



$$\boxed{q(A, RIGHT) = -2.75}$$

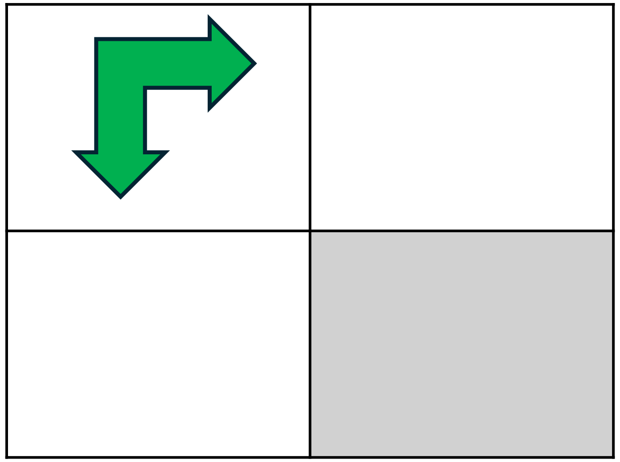
$$\boxed{q(A, DOWN) = -2.75}$$

And update the existing policy for **state A**

$$\boxed{\pi(A) = \{RIGHT, DOWN\}}$$

-2	**-1.75**
-1.75	**0**

$k = 2$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

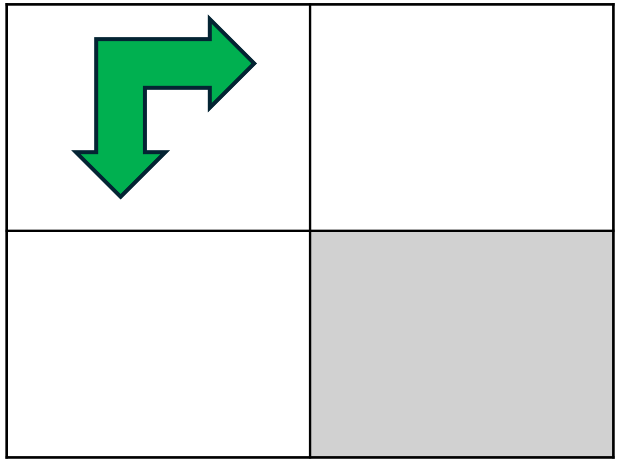
$$q(B, LEFT) = -1 + v(A)$$

$$q(B, LEFT) = -1 + (-2)$$

$$\boxed{q(B, LEFT) =- -3}$$

-2	-1.75
-1.75	0

$k = 2$



$$q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]$$

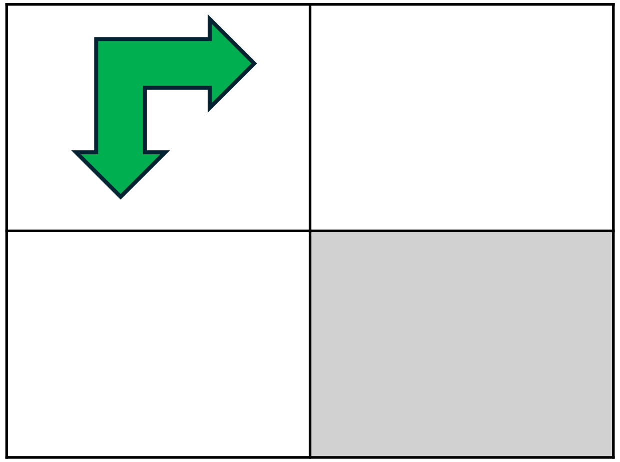
$$q(B, RIGHT) = -1 + v(B)$$

$$q(B, RIGHT) = -1 + (-1.75)$$

$$\boxed{q(B, RIGHT) = -2.75}$$

-2	-1.75
-1.75	0

$k = 2$



$$\boxed{q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]}$$

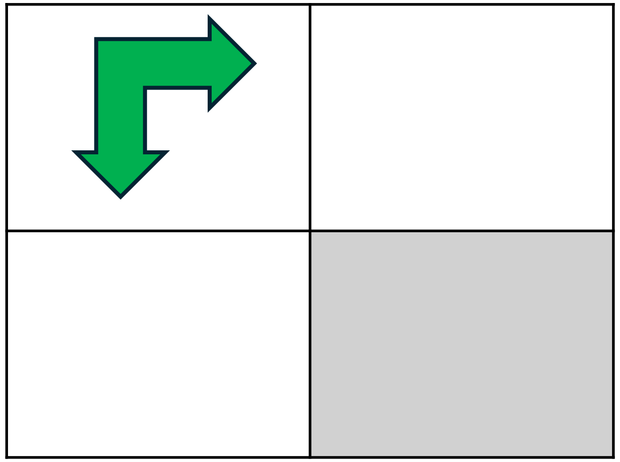
$$q(B, DOWN) = -1 + v(D)$$

$$q(B, DOWN) = -1 + (0)$$

$$\boxed{q(B, DOWN) = -1}$$

-2	-1.75
-1.75	0

$k = 2$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

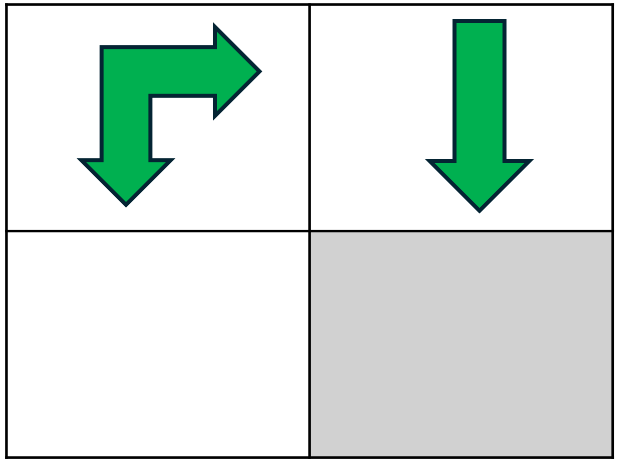
$$q(B,UP) = -1 + v(B)$$

$$q(B,UP) = -1 + (-1.75)$$

$$\boxed{q(B,UP) = -2.75}$$

$$q(B, UP) = -2.75$$

$$q(B, LEFT) = -3$$



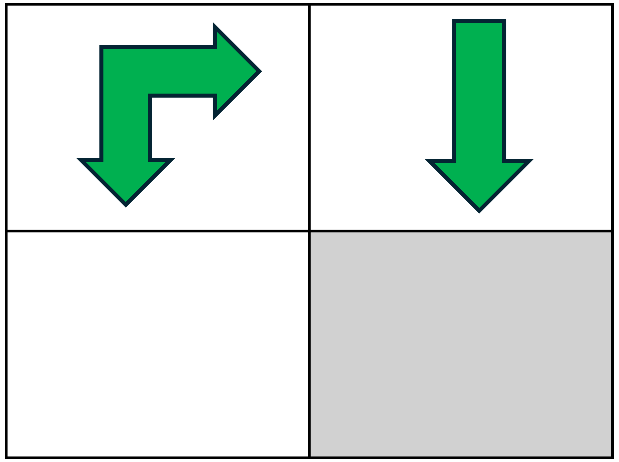
$$q(B, RIGHT) = -2.75$$

$$q(B, DOWN) = -1$$

We can now map the value for each **action** from **state B**

$$q(B, UP) = -2.75$$

$$q(B, LEFT) = -3$$



$$q(B, RIGHT) = -2.75$$

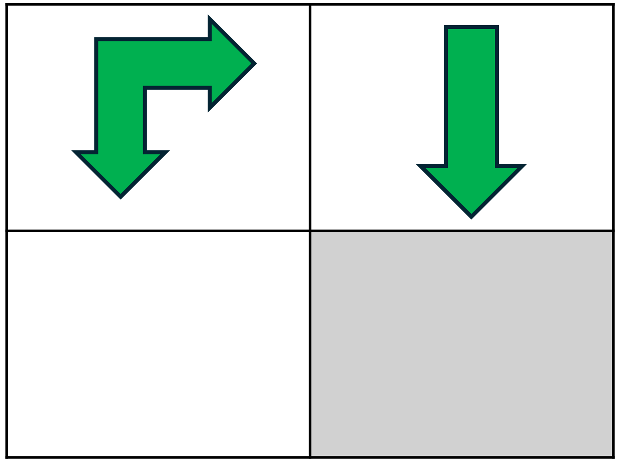
$$\boxed{q(B, DOWN) = -1}$$

Because **DOWN** has the highest action-value function The policy of **state B will not change**

$$\boxed{\pi(B) = \{DOWN\}}$$

-2	**-1.75**
-1.75	**0**

$k = 2$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

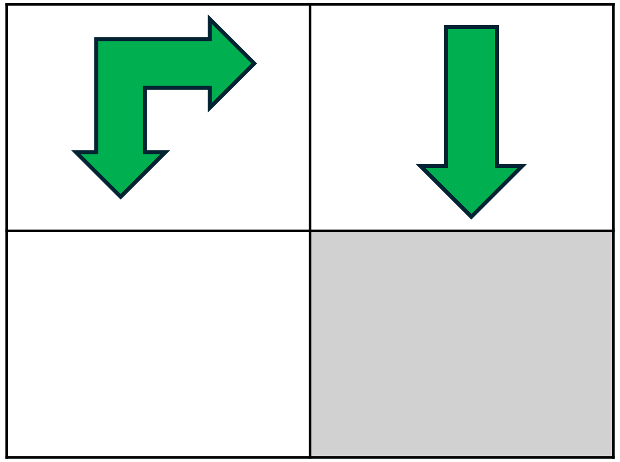
$$q(C, LEFT) = -1 + v(C)$$

$$q(C, LEFT) = -1 + (-1.75)$$

$$\boxed{q(C, LEFT) = -2.75}$$

-2	-1.75
-1.75	0

$k = 2$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

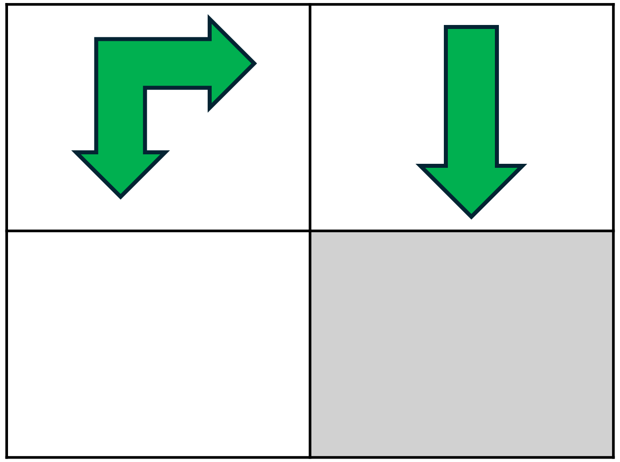
$$q(C, RIGHT) = -1 + v(D)$$

$$q(C, RIGHT) = -1 + (0)$$

$$\boxed{q(C, RIGHT) = -1}$$

-2	-1.75
-1.75	0

$k = 2$



$$q(s,a) = \sum_{s',r} P(s',r \mid s,a)[r + v(s')]$$

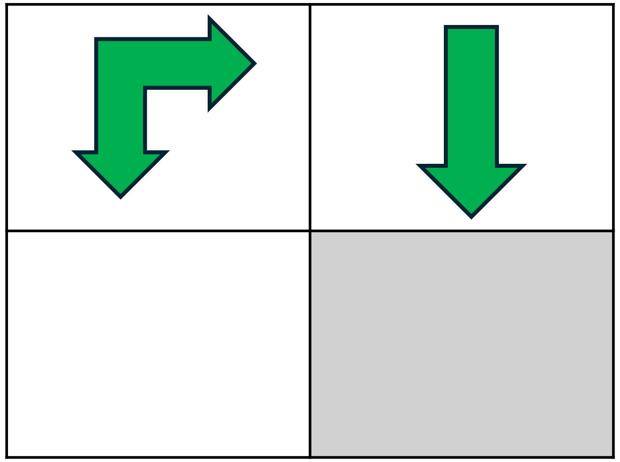
$$q(C, DOWN) = -1 + v(C)$$

$$q(C, DOWN) = -1 + (-1.75)$$

$$\boxed{q(C, DOWN) = -2.75}$$

-2	**-1.75**
-1.75	**0**

$k = 2$



$$q(s,a) = \sum_{s',r} P(s', r \mid s, a)[r + v(s')]$$

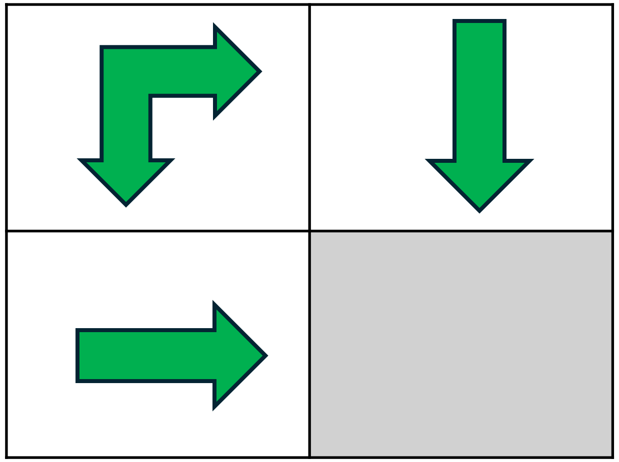
$$q(C, UP) = -1 + v(A)$$

$$q(C, UP) = -1 + (-2)$$

$$\boxed{q(C, UP) = -3}$$

$$q(C, UP) = -3$$

$$q(C, LEFT) = -2.75$$



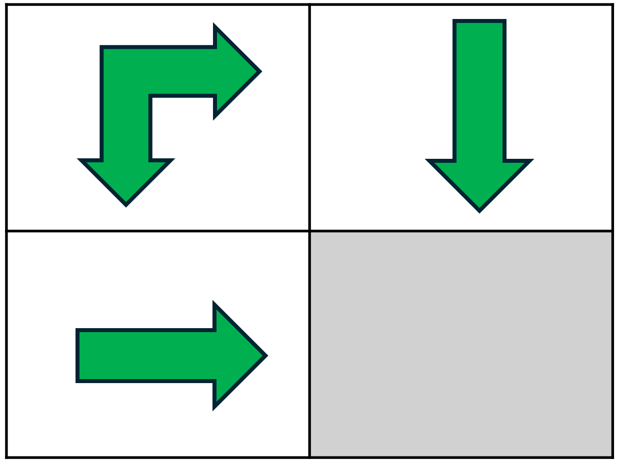
$$\boxed{q(C, RIGHT) = -1}$$

$$q(C, DOWN) = -2.75$$

We can now map the value for each **action** from **state C** and update our **policy**

$$q(C, UP) = -3$$

$$q(C, LEFT) = -2.75$$



$$\boxed{q(C, RIGHT) = -1}$$

$$q(C, DOWN) = -2.75$$

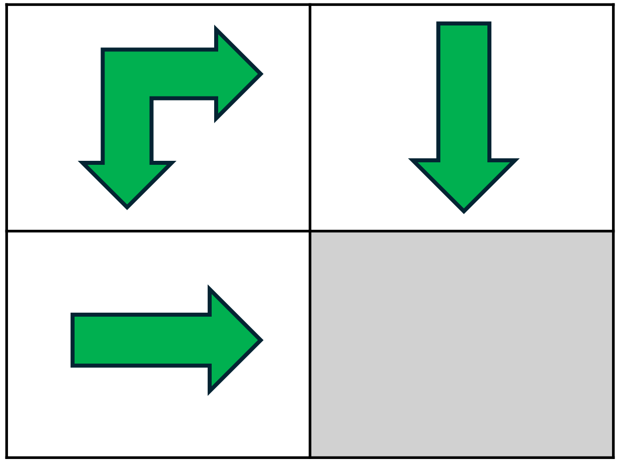
Similar to **state B**, the **policy** for **state C** will stay the same because going right has the highest action value

$$\boxed{\pi(C) = \{RIGHT\}}$$

$k = 2$

-2	**-1.75**
-1.75	**0**

Value function v_k
at time step k



Greedy Policy with
respect to v_k

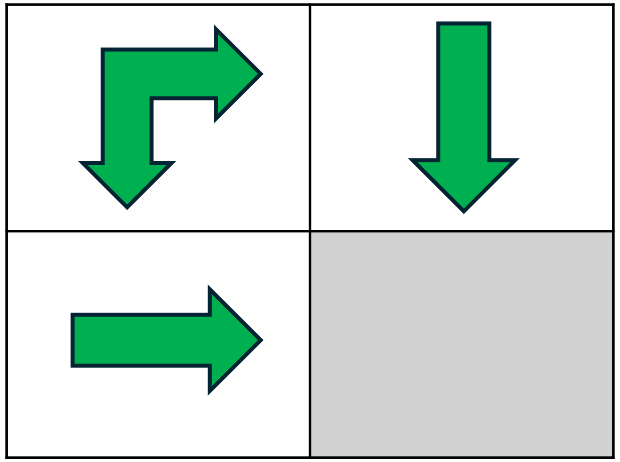
After the second iteration, we can see that we achieved the optimal policy π_*

$\boxed{k = 2}$

$\pi(A) = \{LEFT, RIGHT\}$

$\pi(B) = \{DOWN\}$

$\pi(C) = \{RIGHT\}$



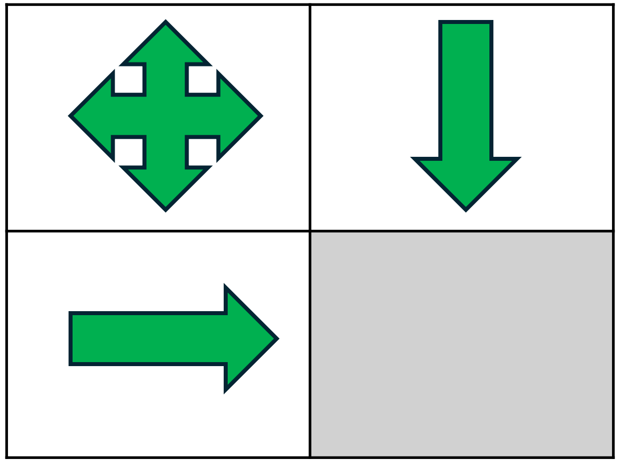
Greedy Policy with respect to v_k

After the second iteration, we can see that we achieved the optimal policy π_*

Value function v_k at time step $k = 1$

-1	-1
-1	0

Greedy Policy with respect to v_k

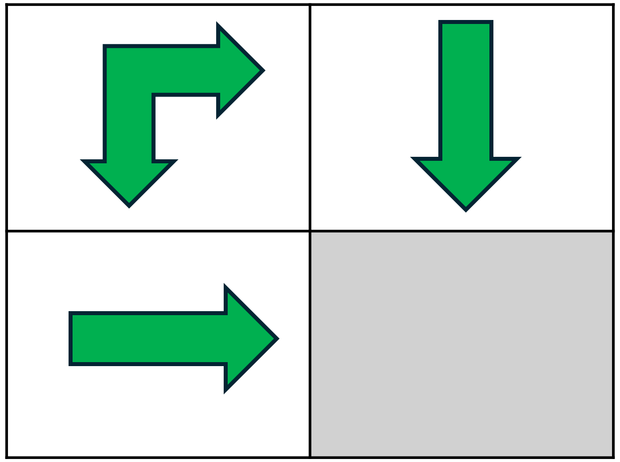


$\boxed{k = 1}$

Value function v_k at time step $k = 2$

-2	-1.75
-1.75	0

Greedy Policy with respect to v_k

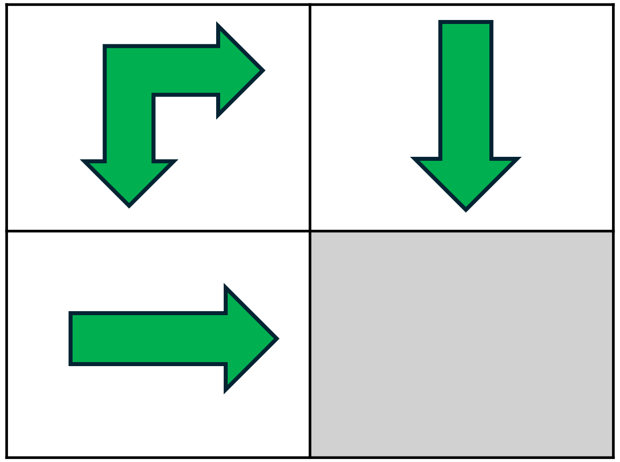


$\boxed{k = 2}$

Value function v_k at time step $k = 3$

−2.875	−2.375
−2.375	0

Greedy Policy with respect to v_k



$\boxed{k = 3\ onwards}$

Value function for all states at each k iteration

$k = 0$

0	0
0	0

$k = 1$

-1	-1
-1	0

$k = 2$

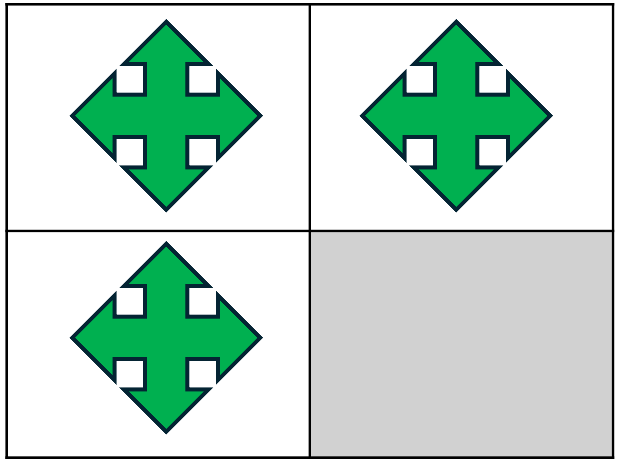
-2	-1.75
-1.75	0

$k = 3$

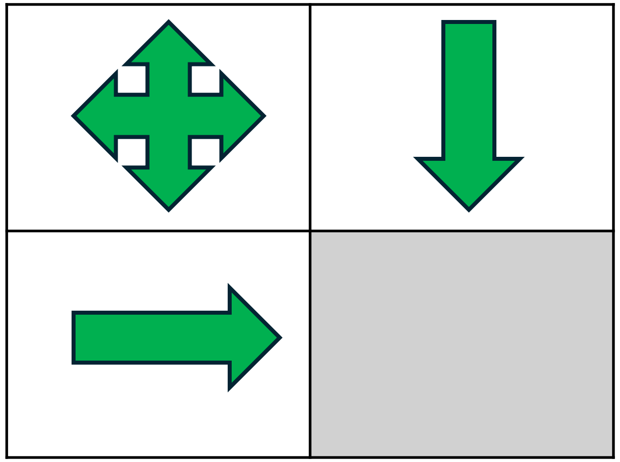
-2.875	-2.375
-2.375	0

Policy improvement each k iteration

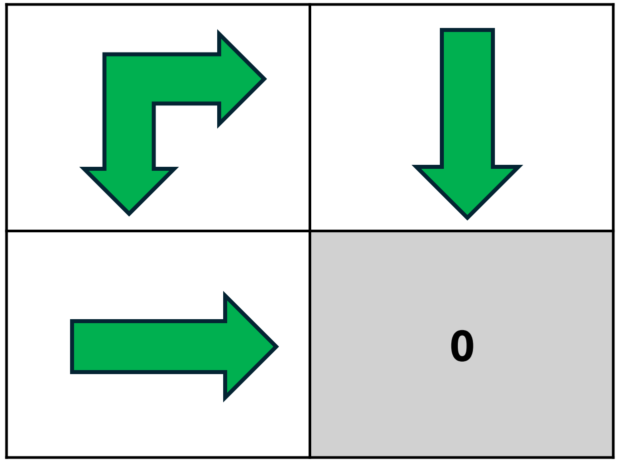
$k = 0$



$k = 1$



$k = 2$



$k = 3$