Exercise 1

Question 1:

a) Markov decision process (MDP) – A MDP is a discrete time stochastic control process. It provides a mathematical framework for modelling decision making in situations where outcomes are partly random and partly under the control of a decision maker.   MDP's are useful for studying optimization problems. In MDP a model of the environment is given (for reward and state transition probabilities) and the environment is fully observable. One key property in MDP is that the future is independent of the past given the present.

Applications:

1) Agriculture – how much to plant based on weather and soil state

2) Water resources – keep the correct water level at the reservoirs

3) A dialogue system to interact with people

4) Deciding how much to invest in a stock

b) State Value Function –
Definition: The state value function of a MDP is the expected return starting from state S

Explanation: The expected return is the sum of all discounted expected rewards.

c) Action value function – action value function q(s,a) is the expected return which are the total discounted future reward starting from state S taking an action a and then following policy π.
It describes how good it is to take a particular action a (not necessarily from policy π) from a a given state S and then following policy.

d) Policy – Is a mapping from states to actions. An optimal policy is a mapping from states to actions that maximizes expected total future rewards. A policy defines the agent's behaviour and it can be deterministic or stochastic.

e) Dynamic Programming – Is a method which simplifying a complicated problem by breaking it down into simpler sub problems in a recursive manner. We solve each time a sub problem recursively until we solve the whole problem.

f) Value iteration – It is solving Bellman optimality equation to find the optimal value function. Using value iteration we derive the optimal policy which is greedy policy because it greedily selects the best action using the value function.

g) Policy iteration – composed out of a policy evaluation and policy improvement step. For a given policy π the action value function is estimated. From the action value function an improved policy is derived following a greedy strategy. This cycle repeated until convergence to the optimal policy.

h) Reinforcement learning – Is the training of machine learning models to make a sequence of decisions. The agent learns to achieve a goal in an uncertain, potentially complex environment where the model of the environment is unknown. Problems that can be solved using reinforcement learning:

1) Traffic light control

2) Robotics - train robots to do certain tasks

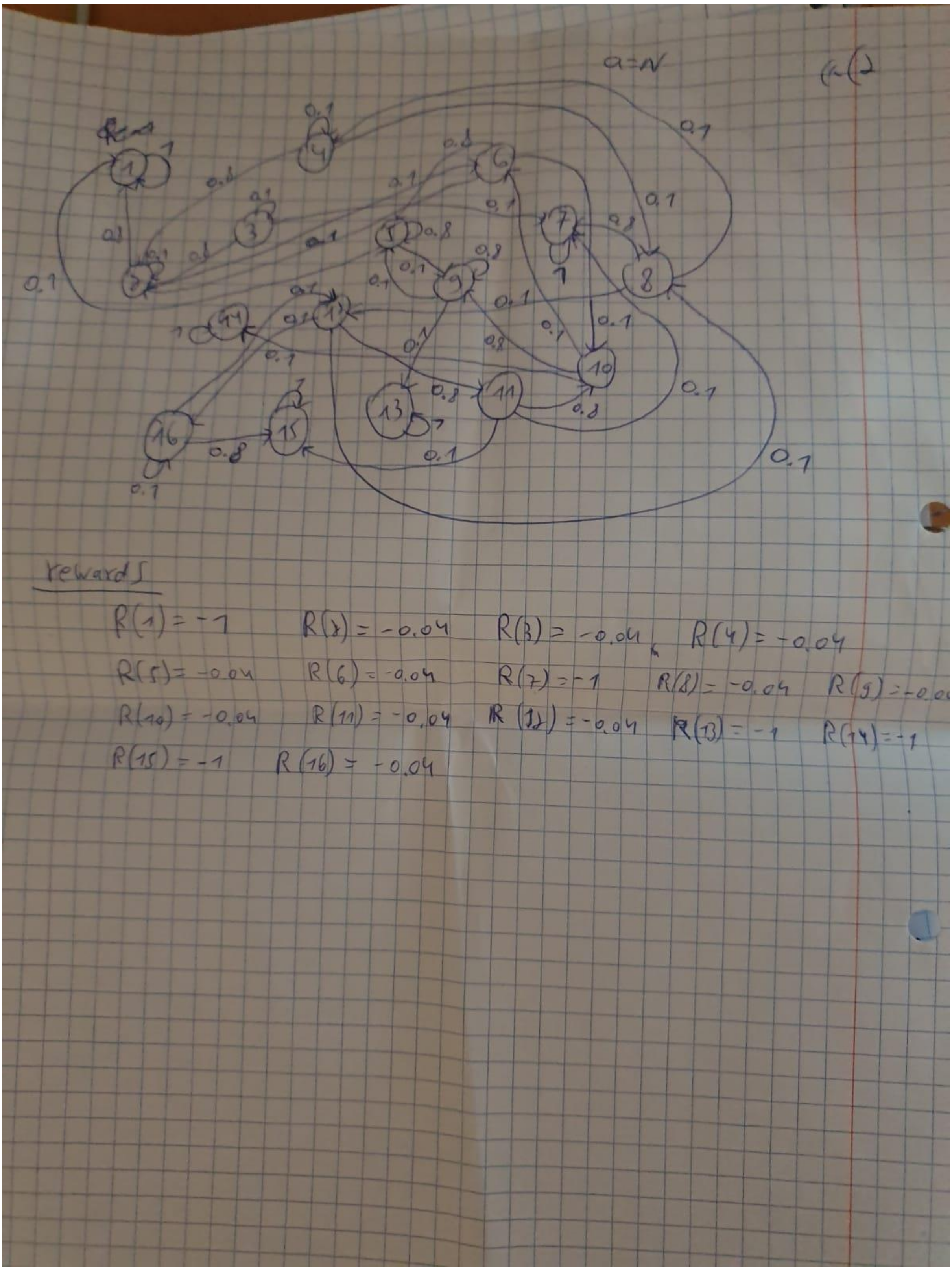
3) Web system configuration

4) Personalized recommendations

- The difference between RL and MDP is that in MDP the model of the environment is known and in RL it's unknown. For example teaching a robot to achieve a specific goal where we know how the environment works it’s an MDP while training a robot to trade in the stock market is RL because we don't know the model of the stock market.

Question 2:

a) graphical model:



reward*s*

R(1) = -1 R(2) = -0.04 R(3) = -0.04 R(4) = -0.04

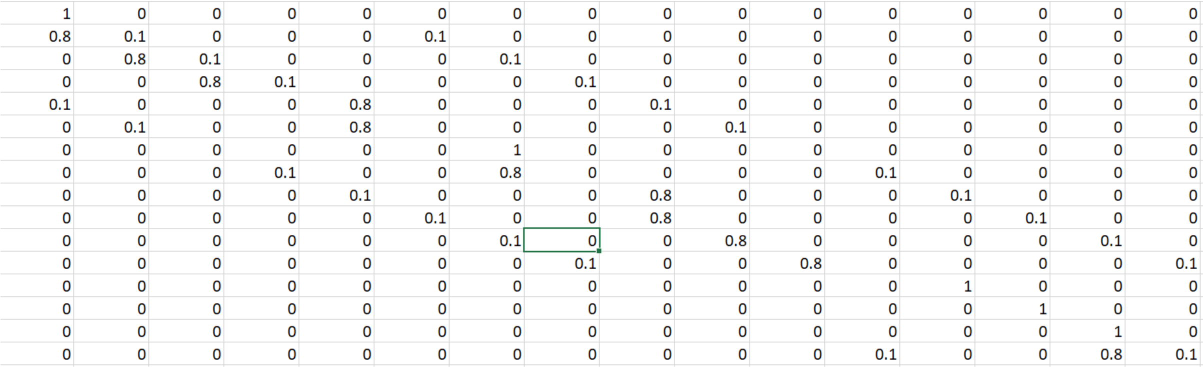
R(5) = -0.04 R(6) = -0.04 R(7) = -1 R(8) = -0.04 R(9) = -0.0

R(10) = -0.04 R(11) = -0.04 R(12) = -0.04 R(13) = -1 R(14) = -1

R(15) = -1 R(16) = -0.04

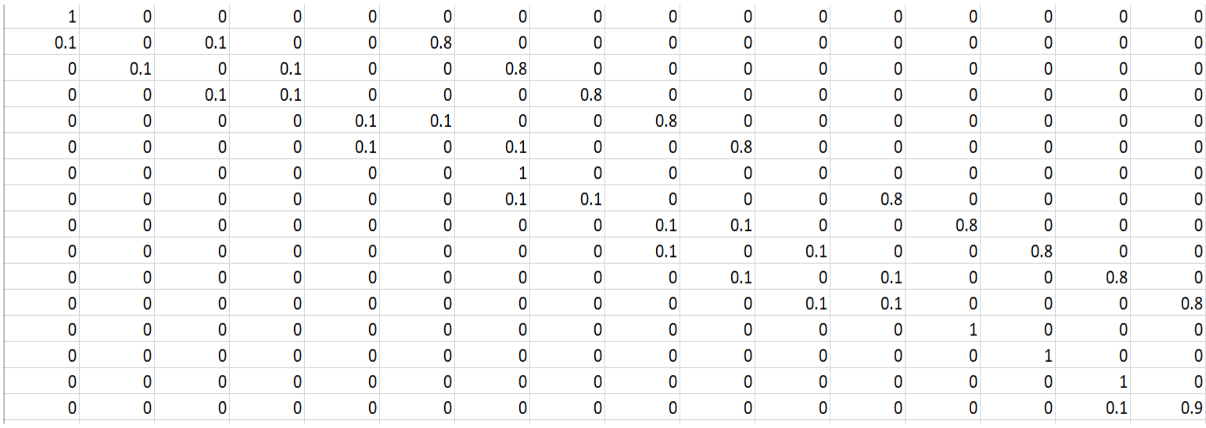
Transition probabilities:

a=N:



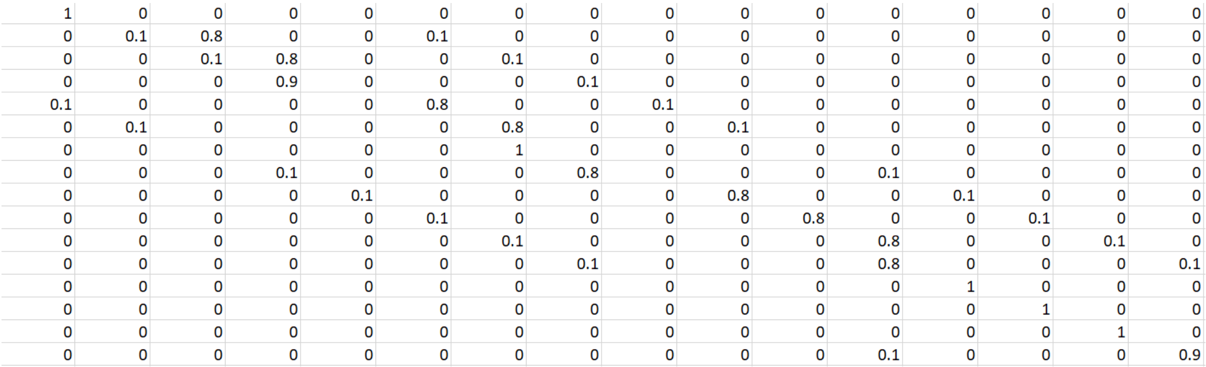
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.8	0.1	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0
0	0.8	0.1	0	0	0	0.1	0	0	0	0	0	0	0	0	0
0	0	0.8	0.1	0	0	0	0.1	0	0	0	0	0	0	0	0
0.1	0	0	0	0.8	0	0	0	0.1	0	0	0	0	0	0	0
0	0.1	0	0	0.8	0	0	0	0	0.1	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0.1	0	0	0.8	0	0	0	0	0.1	0	0	0	0
0	0	0	0	0.1	0	0	0	0.8	0	0	0	0.1	0	0	0
0	0	0	0	0	0.1	0	0	0.8	0	0	0	0	0.1	0	0
0	0	0	0	0	0	0.1	0	0	0.8	0	0	0	0	0.1	0
0	0	0	0	0	0	0	0.1	0	0	0.8	0	0	0	0	0.1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0.8	0.1

a=E:



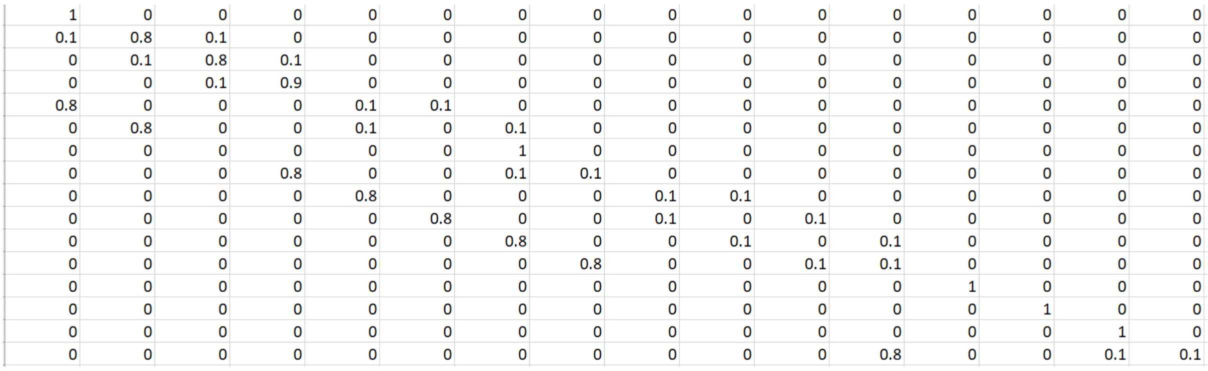
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.1	0	0.1	0	0	0.8	0	0	0	0	0	0	0	0	0	0
0	0.1	0	0.1	0	0	0.8	0	0	0	0	0	0	0	0	0
0	0	0.1	0.1	0	0	0	0.8	0	0	0	0	0	0	0	0
0	0	0	0	0.1	0.1	0	0	0.8	0	0	0	0	0	0	0
0	0	0	0	0.1	0	0.1	0	0	0.8	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0.1	0.1	0	0	0	0.8	0	0	0	0
0	0	0	0	0	0	0	0	0.1	0.1	0	0	0.8	0	0	0
0	0	0	0	0	0	0	0	0.1	0	0.1	0	0	0.8	0	0
0	0	0	0	0	0	0	0	0	0.1	0	0.1	0	0	0.8	0
0	0	0	0	0	0	0	0	0	0	0.1	0.1	0	0	0	0.8
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0.9

a=S:



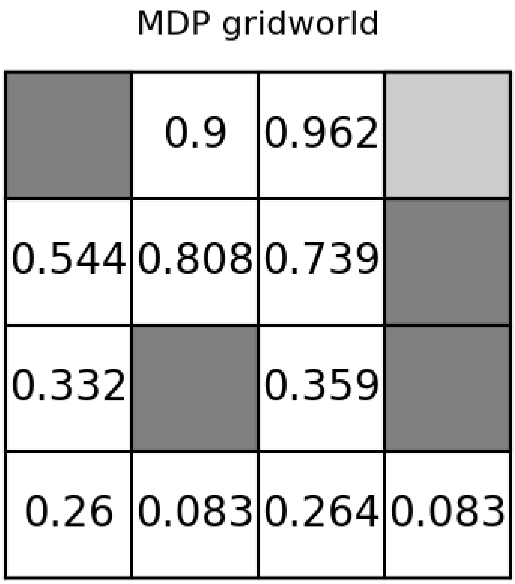
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0.1	0.8	0	0	0.1	0	0	0	0	0	0	0	0	0	0
0	0	0.1	0.8	0	0	0.1	0	0	0	0	0	0	0	0	0
0	0	0	0.9	0	0	0	0.1	0	0	0	0	0	0	0	0
0.1	0	0	0	0	0.8	0	0	0.1	0	0	0	0	0	0	0
0	0.1	0	0	0	0	0.8	0	0	0.1	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0.1	0	0	0	0.8	0	0	0	0.1	0	0	0	0
0	0	0	0	0.1	0	0	0	0	0.8	0	0	0.1	0	0	0
0	0	0	0	0	0.1	0	0	0	0	0.8	0	0	0.1	0	0
0	0	0	0	0	0	0.1	0	0	0	0	0.8	0	0	0.1	0
0	0	0	0	0	0	0	0.1	0	0	0	0.8	0	0	0	0.1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0.9

a=W:



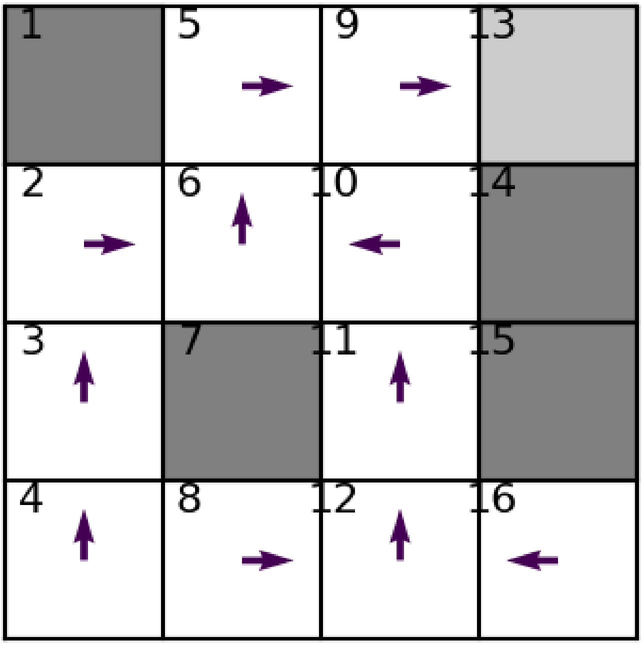
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.1	0.8	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0.1	0.8	0.1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0.1	0.9	0	0	0	0	0	0	0	0	0	0	0	0
0.8	0	0	0	0.1	0.1	0	0	0	0	0	0	0	0	0	0
0	0.8	0	0	0.1	0	0.1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0.8	0	0	0.1	0.1	0	0	0	0	0	0	0	0
0	0	0	0	0.8	0	0	0	0.1	0.1	0	0	0	0	0	0
0	0	0	0	0	0.8	0	0	0.1	0	0.1	0	0	0	0	0
0	0	0	0	0	0	0.8	0	0	0.1	0	0.1	0	0	0	0
0	0	0	0	0	0	0	0.8	0	0	0.1	0.1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0.8	0	0	0.1	0.1

b) value:



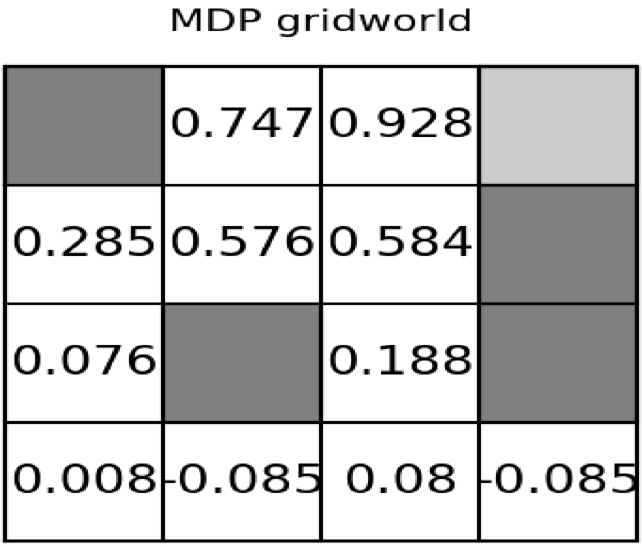
■	0.9	0.962	□
0.544	0.808	0.739	■
0.332	■	0.359	■
0.26	0.083	0.264	0.083

Policy:



1 ■	5 →	9 →	13 □
2 →	6 ↑	10 ←	14 ■
3 ↑	7 ■	11 ↑	15 ■
4 ↑	8 →	12 ↑	16 ←

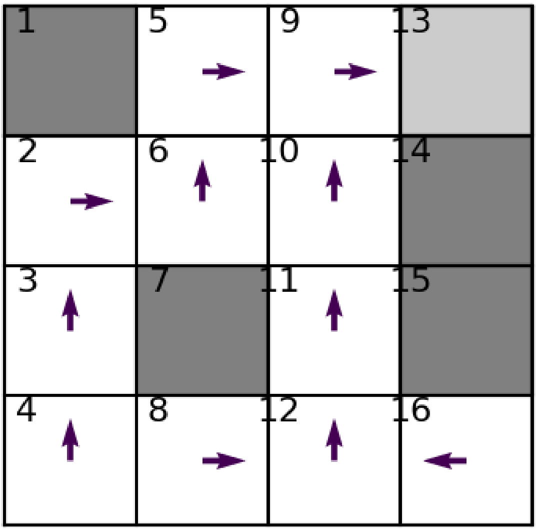
c) value:



MDP gridworld

▓	0.747	0.928	░
0.285	0.576	0.584	▓
0.076	▓	0.188	▓
0.008	-0.085	0.08	-0.085

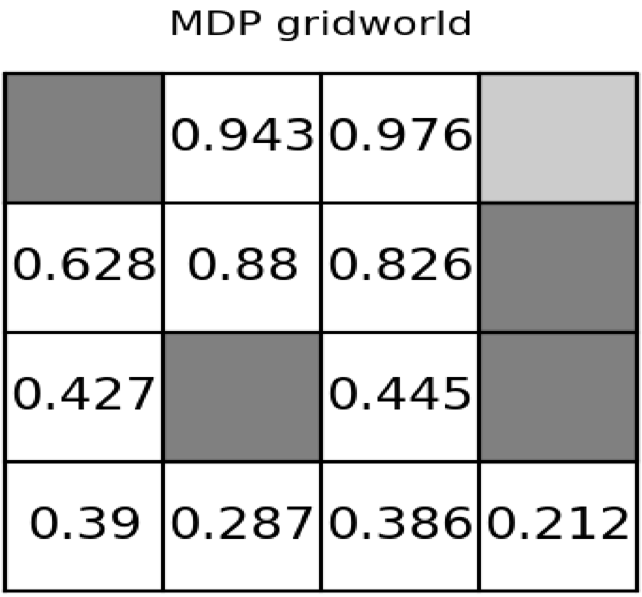
policy:



1 ▓	5 →	9 →	13 ░
2 →	6 ↑	10 ↑	14 ▓
3 ↑	7 ▓	11 ↑	15 ▓
4 ↑	8 →	12 ↑	16 ←

The change affected only state number 10. The optimal action in the 10th state changed from W to N and the gap between the 6th state to the 9th state become larger in favour of the 9th state because the discount factor reduced 6th state more than the 9th state. In general the reduction of the discount factor puts more focus on the closest rewards.

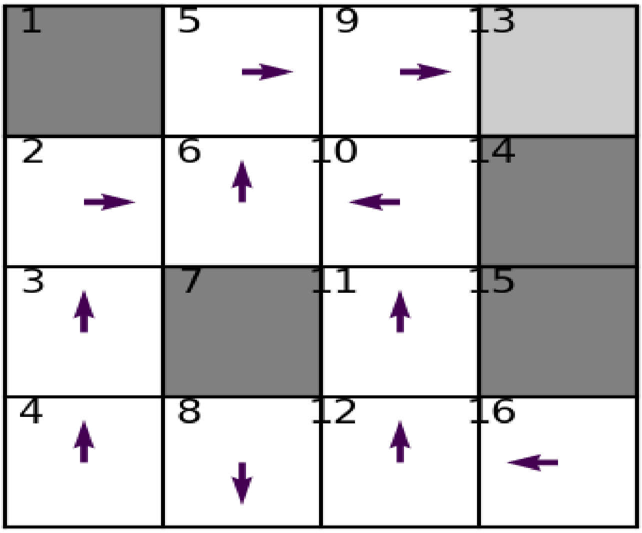
d) value:



MDP gridworld

	0.943	0.976	
0.628	0.88	0.826	
0.427		0.445	
0.39	0.287	0.386	0.212

Policy:

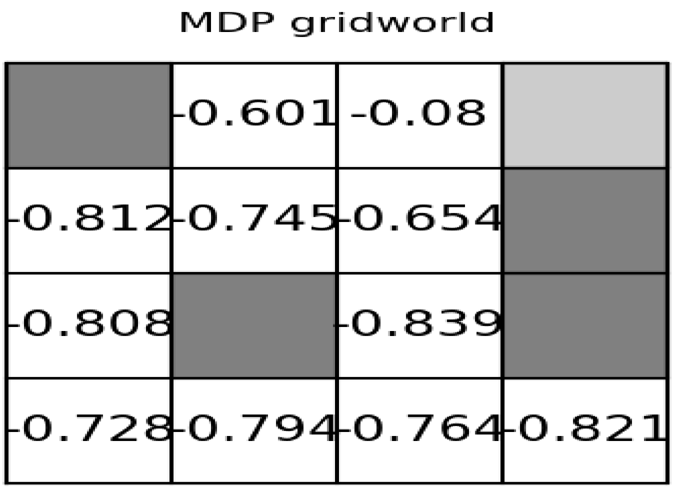


1	5 →	9 →	13
2 →	6 ↑	10 ←	14
3 ↑	7	11 ↑	15
4 ↑	8 ↓	12 ↑	16 ←

There is only one change and it occurs in state number 8. We can see that the decrease of the reward caused the penguin to take longer path until he reaches the target because in section b when the reward was more negative the optimal action was going E which is closer the target while in section d when we lower the punishment the penguin become less efficient.

e) iteration 1:

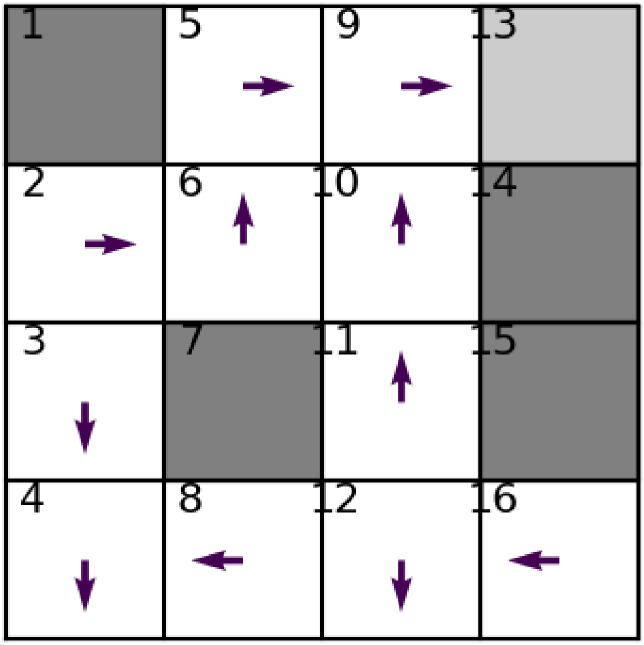
value:



MDP gridworld

	-0.601	-0.08	
-0.812	-0.745	-0.654	
-0.808		-0.839	
-0.728	-0.794	-0.764	-0.821

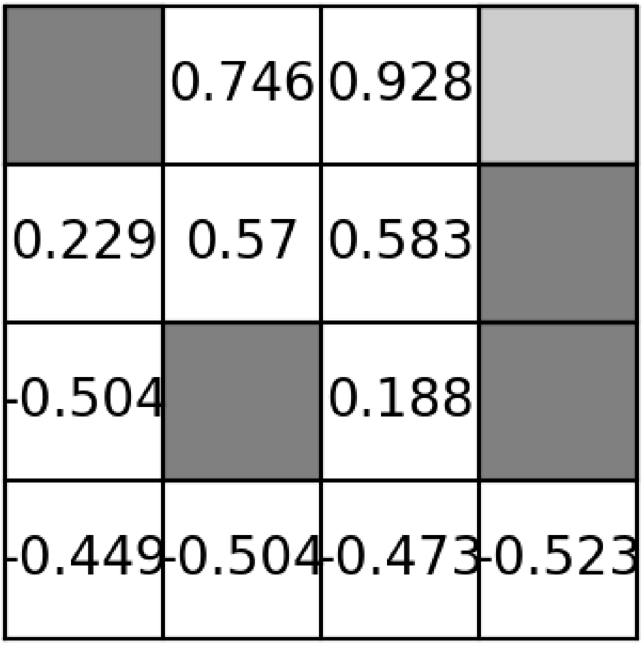
Policy:



Iteration 2:

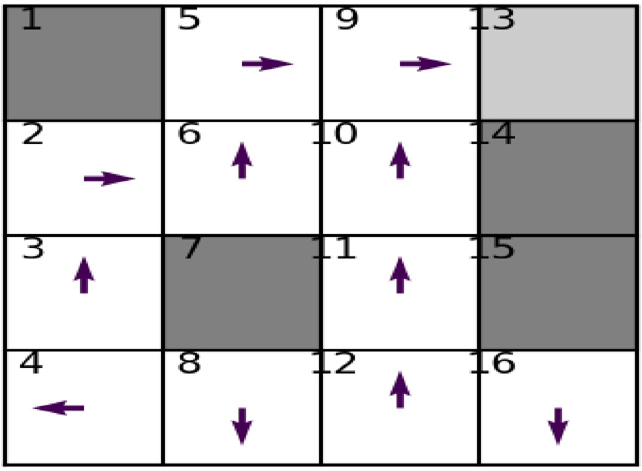
Value:

MDP gridworld



	0.746	0.928	
0.229	0.57	0.583	
-0.504		0.188	
-0.449	-0.504	-0.473	-0.523

Policy:

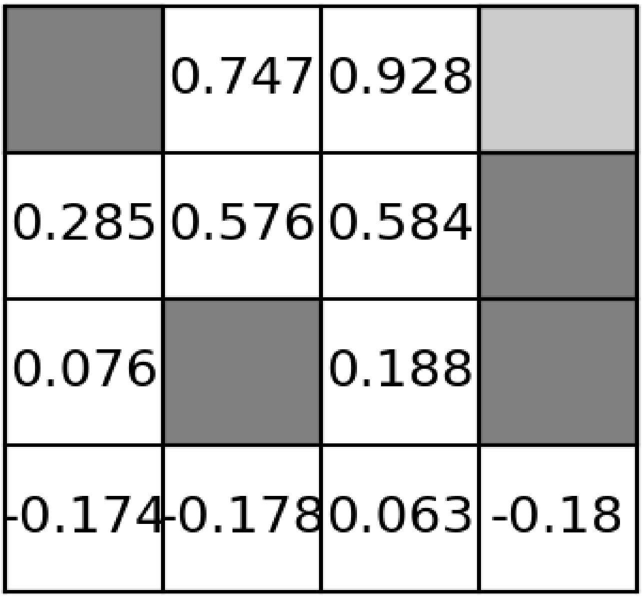


1	5 →	9 →	13
2 →	6 ↑	10 ↑	14
3 ↑	7	11 ↑	15
4 ←	8 ↓	12 ↑	16 ↓

Iteration 3:

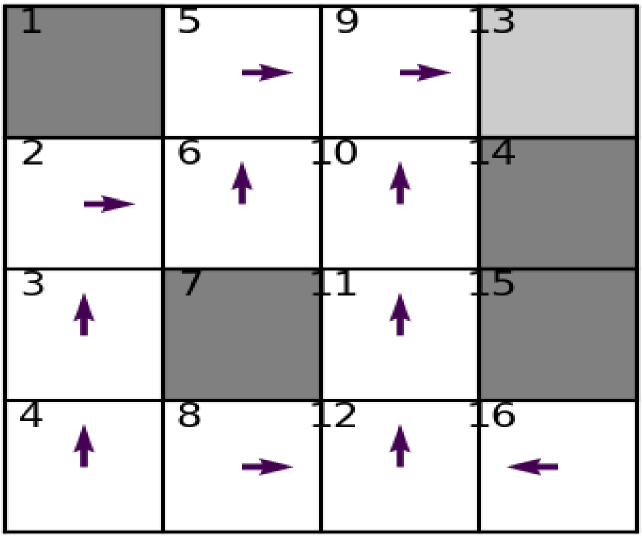
Value:

MDP gridworld



	0.747	0.928	
0.285	0.576	0.584	
0.076		0.188	
-0.174	-0.178	0.063	-0.18

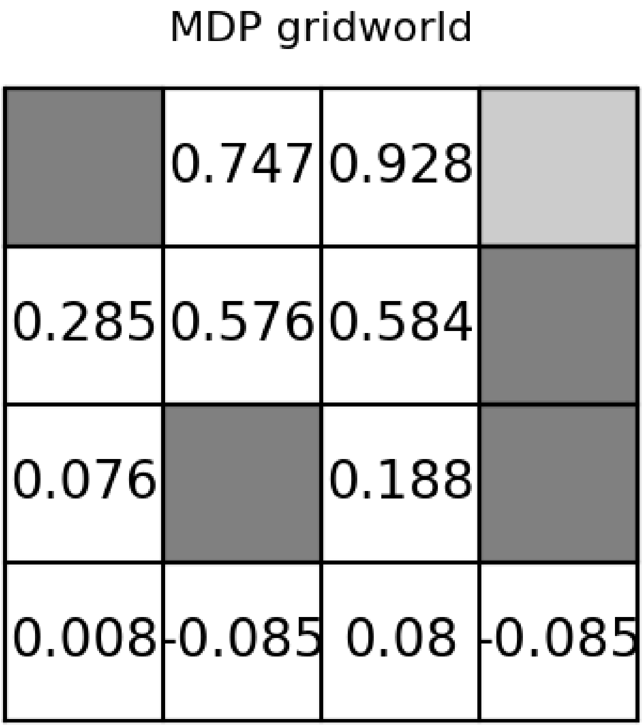
Policy:



1	5 →	9 →	13
2 →	6 ↑	10 ↑	14
3 ↑	7	11 ↑	15
4 ↑	8 →	12 ↑	16 ←

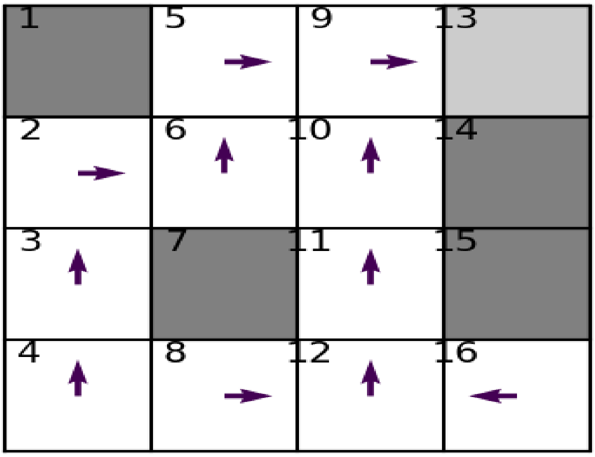
Iteration 4:

Value:



	0.747	0.928	
0.285	0.576	0.584	
0.076		0.188	
0.008	-0.085	0.08	-0.085

Policy:



We can see that the optimal policy achieved by value iteration is equal to the optimal policy achieved by policy iteration.