Report - Assignment 2 - AI

1. A

Discount Factor = 0.1, Step cost = -4.3

Utility Table:

4.300	0.000	-0.913	43.000
-4.043	-4.717	-4.426	-0.913
-4.717	-4.771	0.000	-4.462
-4.771	-4.773	-8.600	-4.770

Policy Map:

* | R *
U L R U
U L | U
U L * R

Start state = (3, 0)
Policy = (3, 0) -Up-> (2, 0) -Up-> (1, 0) -Up-> (0, 0)
End state = (0, 0)
End state reward = 4.3

Conclusions:

- Since discount factor is very less, states that are far away from the start state have very little contribution in calculating the utility. So, it goes to the state without thinking far ahead, and goes to the nearest positive end state.

1.B

Discount Factor = 0.99, Step cost = -4.3

Utility Table:

4.300	0.000	36.463	43.000
16.980	24.748	31.286	36.463
12.674	18.374	0.000	30.647
7.482	10.141	-8.600	21.222

Policy Map:

* | R *
R R R U
R U | U

U U * U

Start state = (3, 0)
Policy = (3, 0) -Up-> (2, 0) -Right-> (2, 1) -Up-> (1, 1) -Right-> (1, 2) -Right-> (1, 3) -Up-> (0, 3)
End state = (0, 3)
End state reward = 43

Since discount factor is high here, it looks into the future properly and goes where the expected reward is highest given the step cost. Since reward at (0, 3) is very high (10 times) as compared to reward at (0, 0), it considers going to (0, 3) instead of (0, 0).

2.A

Discount Factor = 0.99, Step cost = 43

Utility table:

4.300	0.000	4299.990	43.000
4299.990	4299.990	4299.990	4299.990
4299.990	4299.990	0.000	4299.990
4299.990	4299.990	-8.600	4299.990

Policy Map:

* | L *
D R R D
R R | R
R L * R

Start state = (3, 0)
Policy = (3, 0) -Right-> (3, 1) -Left-> (3, 0) …. and so on
Since here step cost value is very high and positive, the environment is very suitable to agent and it does not want to come out of environment, therefore it tries to oscillate between (3,0) and (3,1). There are many other optimal policies for this case. For example instead of left, it could try going upwards also.

2.B

Discount Factor = 0.99, Step cost = -8.6

Utility Table:

4.300	0.000	30.521	43.000
-2.932	8.186	20.637	30.521
-12.531	-3.726	0.000	19.417
-22.164	-14.597	-8.600	6.578

Policy Map:
* | R *
R R R U
U U | U
U U * U

Start state = (3, 0)
Policy = (3, 0) -Up-> (2, 0) -Up-> (1, 0) -Right-> (1, 1)-Right-> (1, 2) -Right-> (1, 3) -Up-> (0, 3)
End state = (0, 3)
End state reward = 43

It's almost same as the policy obtained with step cost = -4.3 in part 1B. The difference is at (2, 0), it went upwards while when step cost was -4.3, it went rightwards. Reason may be because the step cost is higher here, so it takes the risk of ending up in the end state with less positive reward. If it would have gone to right, then there would be higher probability to go down to (3, 1) from which it would have tried to go up again, thereby increasing number of steps and higher unit step cost.

2.C

Discount Factor = 0.99, Step cost = -10.75

Utility Table:
-8.155 -0.035 15.312 27.549
-20.664 -14.233 0.000 13.802
-32.409 -21.055 -8.600 -0.744

Policy Map:
* | R *
U R R U
U U | U
U R * U

Start state = (3, 0)
Policy = (3, 0) -Up-> (2, 0) -Up-> (1, 0) -Up-> (0, 0)
End state = (0, 0)
End state reward = -8.155
Since the step cost is more negative here, and the surrounding cells all have negative values. So basically the whole environment is very negative and the agent would try to get out of it as soon as possible. It had a choice of exiting through (3,1) with fewer number of steps than reaching (0,0) but as the reward of exiting at (0,0) would be slightly greater than (3,1), hence it took that path even by adding one more step.

2.D

Discount Factor = 0.99, Step cost = -43

Utility Table:
4.300 0.000 -17.020 43.000
-56.052 -110.183 -64.556 -17.020
-110.183 -120.020 0.000 -70.424
-120.020 -68.472 -8.600 -63.022

Policy Map:
* | R *
U L R U
U D | U
R R * L

Start State = (3, 0)
Policy = (3, 0) -Right-> (3, 1) -Right-> (3, 2)
End state = (3, 2)
End state reward = -8.6

As the step cost is very high here, the environment is very painful to the agent and it wants to get out of it as soon as possible, so it goes to the nearest end state possible even if it has negative reward.