Reinforcement Learning Assignment 2
Harsh Pathak, 2016041

Q1.

$S = \{high, low\}$

$A(h) = \{search, wait\}$

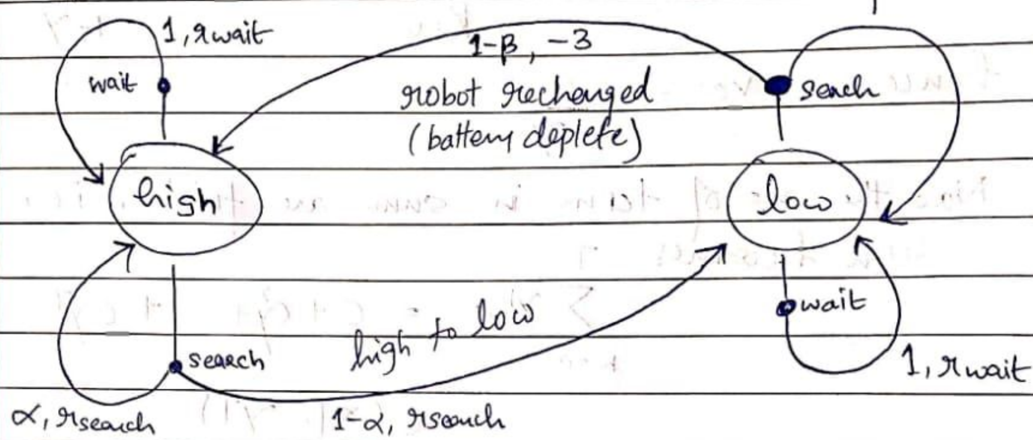
$A(low) = \{search, wait, recharge\}$

high to low with probability $1-\alpha$

high to high " " α

low to dead " " $1-\beta$

low to low " " β



s	a	s'	r	$p(s', r \mid s, a)$
high	wait	high	r_{wait}	1
high	search	high	r_{search}	α
high	search	low	r_{search}	$1-\alpha$
low	search	low	r_{search}	β
low	search	high	-3	$1-\beta$
low	wait	low	r_{wait}	1

Q2.

```
[Harshs-MacBook-Pro:A2 harshpathak$ python3 q1.py
3.31 8.79 4.43 5.32 1.49
1.52 2.99 2.25 1.91 0.55
0.05 0.74 0.67 0.36 -0.40
-0.97 -0.44 -0.35 -0.59 -1.18
-1.86 -1.35 -1.23 -1.42 -1.98
Harshs-MacBook-Pro:A2 harshpathak$
```

Fig 1 -> V(s) for all the 25 states

Q3.

Q3.15 $V_\pi(s) = E[G_t | S_t = s]$

$$= V_\pi(s) = E\left[\sum_{k=0}^{\infty} \gamma^k \{R_{t+k+1} + c\} \,\middle|\, S_t = s\right]$$

$$= E\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right] + E\left[\sum_{k=0}^{\infty} c\gamma^k \,\middle|\, S_t = s\right]$$

$$= V_\pi(s) + \sum_{k=0}^{\infty} \gamma^k c = V_\pi(s) + \frac{c}{1-\gamma}$$

hence $v_c = \frac{c}{1-\gamma}$

Q3.16 Now the no. of term in sum are finite, i.e, sum becomes

$$\sum_{k=0}^{T} \gamma^k c = c + c\gamma + \dots + c\gamma^T$$

$$= \frac{c(1-\gamma^T)}{(1-\gamma)}$$

$$= c(1+\gamma+\gamma^2+\dots+\gamma^{T-1})$$

$$V_\pi(s)^{(2)} = V_\pi(s) + \frac{c(1-\gamma^T)}{(1-\gamma)}$$

This will affect the relative values because consider that if the gridworld is an episodic task (some terminal states), then

~~$V_\pi(s)$ will change depending on where s appear in the given episode. Hence depending on when S comes earlier or later $\frac{1-\gamma^T}{1-\gamma}$ will become more or less respectively~~

Since G_T is now also affected by the length of the sequence after it, hence $V_\pi(s) = E[G_t | S_t = s]$ is also dependent on length of episodic task

Q4.

Solving bellman optimality equation using linear programming
v*(s) = max (p(s',r|s,a)[r + yv*(s')])

since each s has 4 actions, we have 4 inequalities for each state, of the form
v*(s) >= p(s',r|s,a)[r + yv*(s')), for each possible s'

since there are 25 states, we have 100 inequalities
We now want to minimize sigma(v*(s)) with respect to these inequalities

Let c = e, where e is a vector of all 1's with length 25
A -> 100 x 25 matrix that will capture the inequalities
b -> 100 length vector that stores the constants -> (-r * p(s',r|s,a))
Optimization -> minimize $c^T x$ such Ax <= b
where x is what we want to find (25 length vector storing all v(s))
This now can be solved using linear programming

```
 success: True
       x: array([21.97748507, 24.41942783, 21.97748505, 19.41942792, 17.47748518,
       19.7797366 , 21.97748504, 19.77973655, 17.8017629 , 16.02158665,
       17.80176298, 19.77973653, 17.80176291, 16.02158664, 14.41942805,
       16.02158672, 17.80176287, 16.02158665, 14.41942803, 12.97748532,
       14.41942813, 16.02158658, 14.41942804, 12.97748532, 11.67973693])
Harshs-MacBook-Pro:A2 harshpathak$
```

Fig2 : Value of x, which can be seen as 5 x 5 matrix

$$v_*(s) = \max_{a \in \mathcal{A}(s)} q_{\pi_*}(s, a)$$

Q5.
Q6.
Policy Iteration -

```
iter 0
0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00
X up up up
up up up up
up up up up
up up up X
--------------------
iter 1
0.00 -13.93 -19.91 -21.90
-13.93 -17.92 -19.91 -19.91
-19.91 -19.91 -17.93 -13.95
-21.90 -19.91 -13.95 0.00
X left left left
up up left down
up up down down
up right right X
--------------------
iter 2
0.00 -1.00 -2.00 -3.00
-1.00 -2.00 -3.00 -2.00
-2.00 -3.00 -2.00 -1.00
-3.00 -2.00 -1.00 0.00
X left left left
up up up down
up up down down
up right right X
--------------------
```

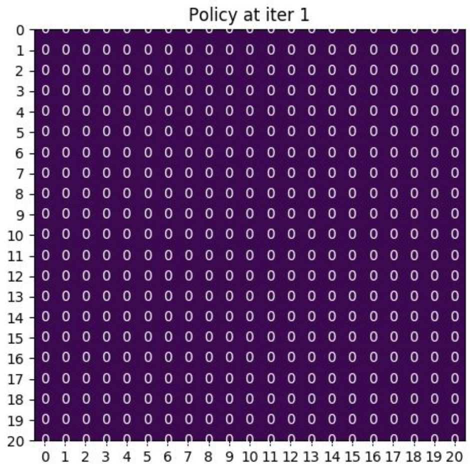
Value Iteration -

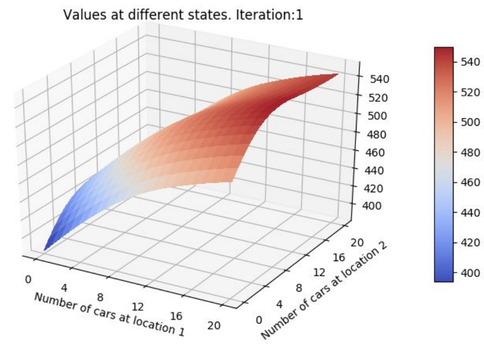
```
iter 1
0.00 -1.00 -1.00 -1.00
-1.00 -1.00 -1.00 -1.00
-1.00 -1.00 -1.00 -1.00
-1.00 -1.00 -1.00 0.00
--------------------
iter 2
0.00 -1.00 -2.00 -2.00
-1.00 -2.00 -2.00 -2.00
-2.00 -2.00 -2.00 -1.00
-2.00 -2.00 -1.00 0.00
--------------------
iter 3
0.00 -1.00 -2.00 -3.00
-1.00 -2.00 -3.00 -2.00
-2.00 -3.00 -2.00 -1.00
-3.00 -2.00 -1.00 0.00
--------------------
iter 4
0.00 -1.00 -2.00 -3.00
-1.00 -2.00 -3.00 -2.00
-2.00 -3.00 -2.00 -1.00
-3.00 -2.00 -1.00 0.00
--------------------
```

Q7.

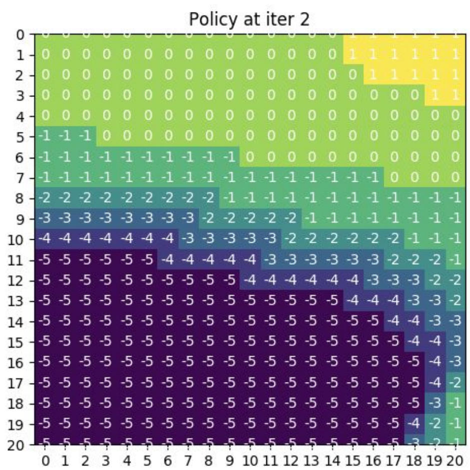
Iter	Policy	Value Function

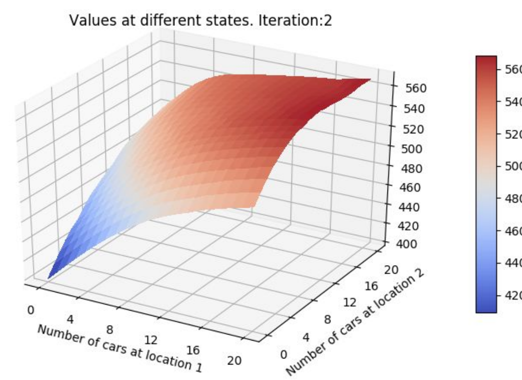
1


Policy at iter 1

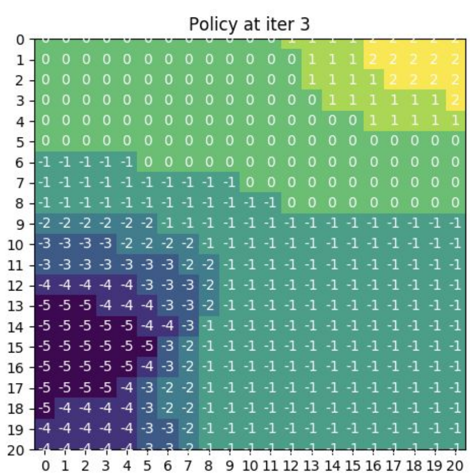

Values at different states. Iteration:1
Number of cars at location 1
Number of cars at location 2

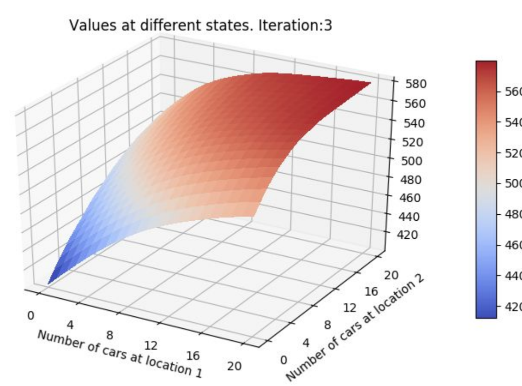
2

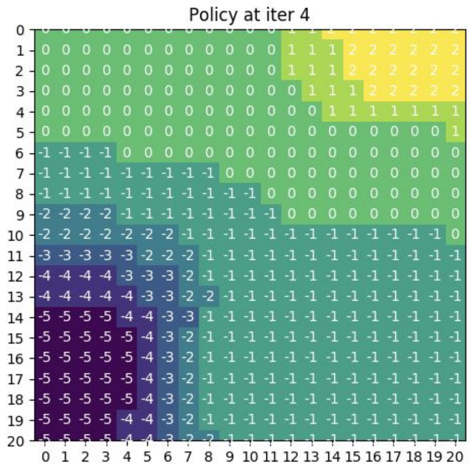
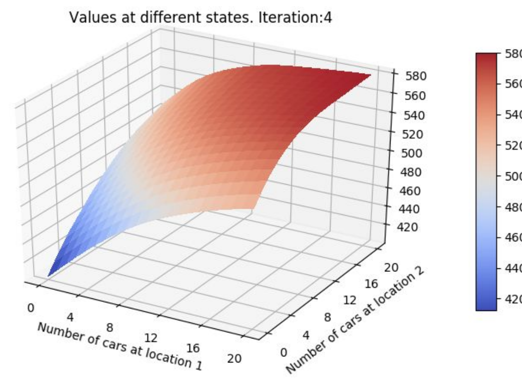
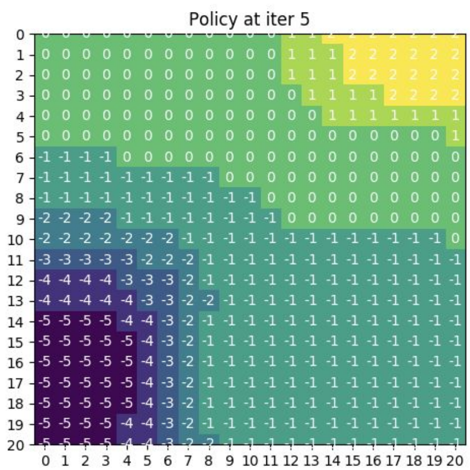
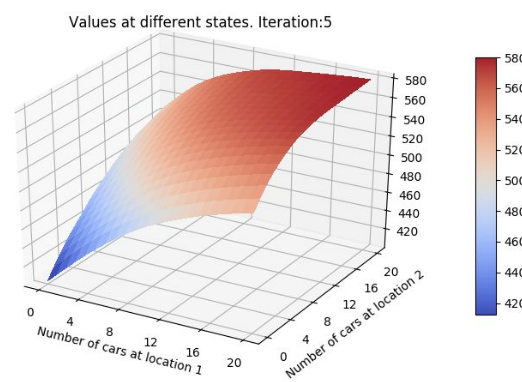

Policy at iter 2


Values at different states. Iteration:2
Number of cars at location 1
Number of cars at location 2

3


Policy at iter 3


Values at different states. Iteration:3
Number of cars at location 1
Number of cars at location 2

4	 Policy at iter 4	 Values at different states. Iteration:4 Number of cars at location 1 Number of cars at location 2 0 4 8 12 16 20 420 440 460 480 500 520 540 560 580
5	 Policy at iter 5	 Values at different states. Iteration:5 Number of cars at location 1 Number of cars at location 2 0 4 8 12 16 20 420 440 460 480 500 520 540 560 580