Q.1

$$\sum_{n \geq t} \gamma^n \eta_n \leq \sum_{n \geq t} \gamma^n$$

$$\leq \gamma^t \times \left(\sum_{n=0}^{\infty} \gamma^t \right)$$

$$\leq \gamma^t \times \frac{1}{(1-\gamma)} \quad \cdots (i)$$

$$\gamma^t = e^{\log \gamma^t} = e^{t \log \gamma} \leq e^{t(\gamma - 1)}$$

$$\Rightarrow \quad \gamma^t \leq e^{-t(1-\gamma)} \quad \cdots (ii)$$

$\Rightarrow$ Substituting in (i).

$$\sum_{n \geq t} \gamma^n \eta_n \leq \frac{\gamma^t}{(1-\gamma)} \leq \frac{e^{-t(1-\gamma)}}{(1-\gamma)}$$

$$\Rightarrow \boxed{\sum_{n \geq t} \gamma^n \eta_n \leq h e^{-t/h}}$$

q.2

01)

$$V^{\pi}(S=2) = R(S=2) + \gamma\left(P(S'=1|S=2,\uparrow)V^{\pi}(S=1) + P(S'=2|S=2,\uparrow)V^{\pi}(S=2) + P(S'=3|S=2,\uparrow)V^{\pi}(S=3)\right)$$

$$V^{\pi}(S=2) = 30 + \frac{2}{3}\left(\frac{1}{2}\times(-18) + \frac{1}{2}V^{\pi}(S=2)\right)$$

$$\frac{2}{3}V^{\pi}(S=2) = 30 - 6$$

$$\boxed{V^{\pi}(S=2) = 36}$$

$$V^{\pi}(S=3) = R(S=3) + \gamma\left(P(S'=1|S=3,\downarrow)V^{\pi}(S'=1) + P(S'=2|S=3,\downarrow)V^{\pi}(S'=2) + P(S'=3|S=3,\downarrow)V^{\pi}(S'=3)\right)$$

$$V^{\pi}(S=3) = -25 + \frac{2}{3}\left(0 + \frac{1}{2}\times 36 + \frac{1}{2}\times V^{\pi}(S'=3)\right)$$

$$= -25 + 6 + \frac{V^{\pi}(S'=3)}{2}$$

$$\Rightarrow V^{\pi}(S=3) = -19\times 2$$

$$\boxed{V^{\pi}(S=3) = -38}$$

b) $\pi'(s) = \underset{a}{\operatorname{argmax}}\, \phi^{\pi}(s,a)$

$$V^{\pi'}(s) \geq V^{\pi}(s) \quad \forall s \in S$$

$S = 1$

$$\Rightarrow \phi^{\pi}(S=1, \uparrow) = R(S=1) + \gamma\Big(P(S'=1|S=1,\uparrow)V^{\pi}(S'=1) + P(S'=2|S=1,\pi)V^{\pi}(S'=2) + P(S'=3|S=1,\pi)V^{\pi}(S'=3)\Big)$$

$$= -15 + \frac{2}{3}\left(\frac{3}{4_2} \times (-18) + \frac{1}{4_2} \times 36 + 0\right)$$

$$= -15 - 9 + 6$$

$$\phi^{\pi}(S=1,\uparrow) = -18$$

$$\Rightarrow \phi^{\pi}(S=1,\downarrow) = R(S=1) + \gamma\Big(P(S'=1|S=1,\downarrow)V^{\pi}(S'=1) + P(S'=2|S=1,\downarrow)V^{\pi}(S'=2) + P(S'=3|S=1,\downarrow)V^{\pi}(S'=3)\Big)$$

$$= -15 + \frac{2}{3}\left(\frac{1}{4_2} \times (-18) + \frac{3}{4_2} \times 36 + 0\right)$$

$$= -15 - 6 + 18$$

$$\phi^{\pi}(S=1,\downarrow) = -3$$

$\Rightarrow \pi'(s=1) = \downarrow$

$s=2$

$\Rightarrow q^{\pi}(s=2,\uparrow) = R(s=2) + \gamma\big(P(s'=1|s=2,\uparrow)V^{\pi}(s=1) + P(s'=2|s=2,\uparrow)V^{\pi}(s'=2) + P(s'=3|s=2,\uparrow)V^{\pi}(s'=3)\big)$

$= 30 + \frac{2}{3}\left(\frac{1}{2}\times(-18) + \frac{1}{2}\times 36 + 0\right)$

$q^{\pi}(s=2,\uparrow) = 36$

$\Rightarrow q^{\pi}(s=2,\downarrow) = (R(s=2)) + \gamma\big(P(s'=1|s=2,\downarrow)V^{\pi}(s'=1) + P(s'=2|s=2,\downarrow)V^{\pi}(s'=2) + P(s'=3|s=2,\downarrow)V^{\pi}(s'=3)\big)$

$= 30 + \frac{2}{3}\left(0 + \frac{1}{2}\times 36 + \frac{1}{2}\left(\frac{-38}{3}\right)\right)$

$= 30 + 12 - \frac{38}{3}$

$= 42 - \frac{38}{3}$

$q^{\pi}(s=2,\downarrow) = 29.\overline{3}$

$\Rightarrow \pi'(s=2) = \uparrow$

$S=3$

$$\Rightarrow q^{\pi}(S=3,\uparrow) = R(S=3) + \gamma\big(P(S'=1|S=3,\uparrow)V^{\pi}(S'=1) + P(S'=2|S=3,\uparrow)V^{\pi}(S'=2) + P(S'=3|S=3,\uparrow)V^{\pi}(S'=3)\big)$$

$$= -25 + \frac{2}{3}\left(0 + \frac{3}{22}\times\frac{36}{18} + \frac{1}{42}\left(\frac{-38}{-19}\right)\right)$$

$$= -25 + 18 - \frac{19}{3}$$

$$= -\frac{71-19}{3}$$

$$= -\frac{40}{3}$$

$$q^{\pi}(S=3,\uparrow) = -13.\overline{3}$$

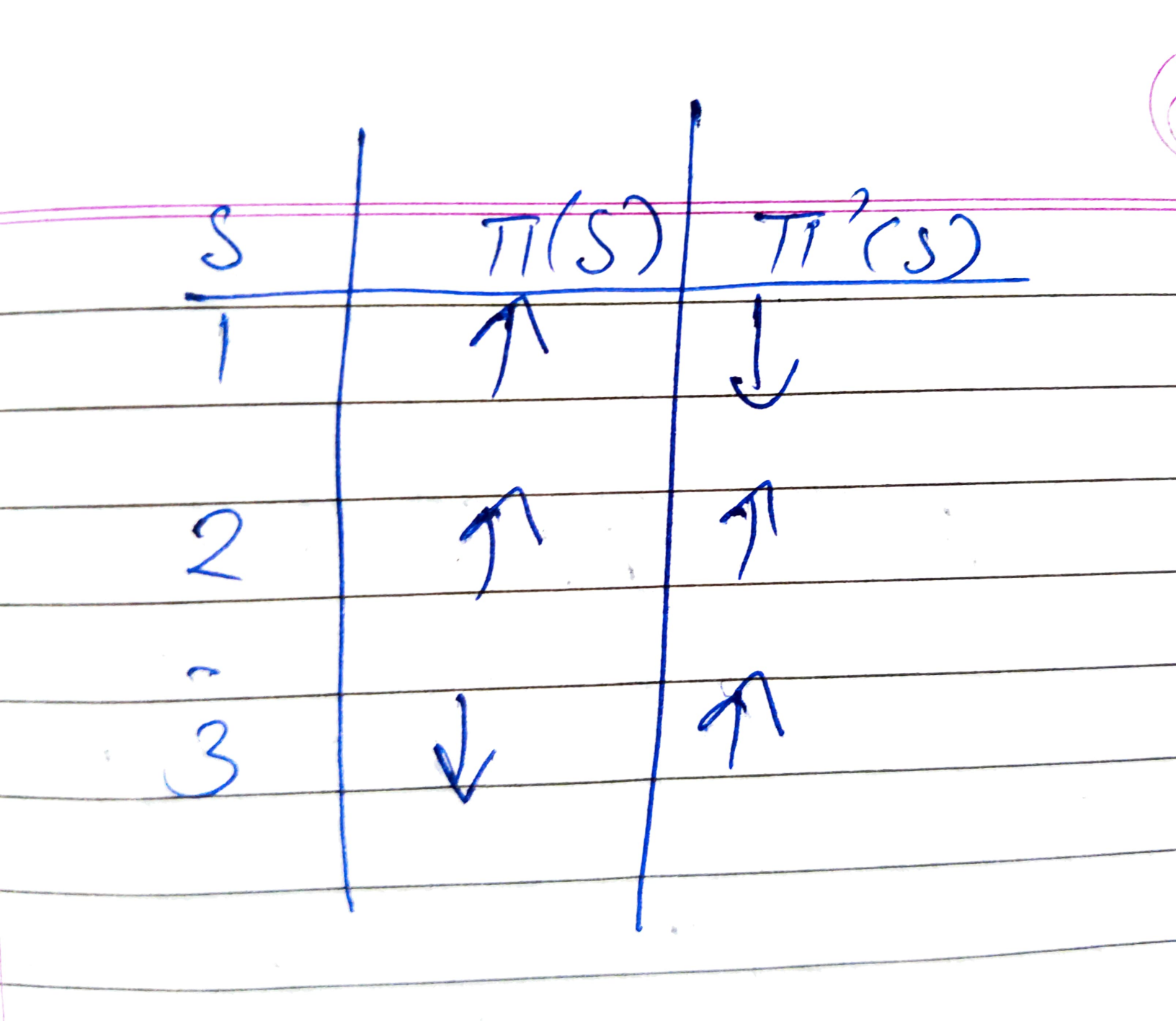
$$\Rightarrow q^{\pi}(S=3,\downarrow) = R(S=3) + \gamma\big(P(S'=1|S=3,\downarrow)V^{\pi}(S'=1) + P(S'=2|S=3,\downarrow)V^{\pi}(S'=2) + P(S'=3|S=3,\downarrow)V^{\pi}(S'=3)\big)$$

$$= -25 + \frac{2}{3}\left(0 + \frac{1}{42}\times\frac{63}{18} + \frac{2}{42}\left(\frac{-19}{-38}\right)\right)$$

$$= -25 + 3 - 19$$

$$q^{\pi}(S=3,\downarrow) = -41$$

$$\Rightarrow \pi'(S=3) = \uparrow$$


S
π(S)
π'(S)
1
2
3

Q.3

a) $V^{\pi}(s) = R(s) + \gamma \sum_{s' \neq 0}^{\infty} P(s'|s,a) V^{\pi}(s')$

$$V^{\pi}(s) = s + \gamma\left(\frac{2}{3} V^{\pi}(s) + \frac{V^{\pi}(s+1)}{3}\right)$$

b) $$as + b = s + \gamma\left(\frac{2}{3}(as+b) + \frac{1}{3}(a(s+1)+b)\right)$$

$$\Rightarrow as + b = \left(1 + \frac{2}{3}a\gamma + \frac{1}{3}a\gamma\right)s + \gamma\left(b + \frac{a}{3}\right)$$

$$\Rightarrow a = 1 + a\gamma \Rightarrow \boxed{a = \frac{1}{1-\gamma}}$$

$$b = \gamma\left(b + \frac{a}{3}\right) \Rightarrow b = \frac{1}{(1-\gamma)} \times \frac{1}{3} \times \frac{\gamma}{(1-\gamma)}$$

$$\Rightarrow \boxed{b = \frac{\gamma}{3(1-\gamma)^2}}$$

In [1]:

```
import numpy as np
import pandas as pd
import random
```

In [2]:

```
def loadFile(filename, multi_col=False):
    contents = []
    with open(filename) as file:
        for line in file:
            temp = line.strip('\n').strip()
            if multi_col:
                temp = temp.split()
            contents.append(temp)

    return np.asarray([np.array(x) for x in contents])
```

In [3]:

```
prob_a1_df = loadFile('prob_a1.txt', multi_col=True).astype(np.float32)
prob_a2_df = loadFile('prob_a2.txt', multi_col=True).astype(np.float32)
prob_a3_df = loadFile('prob_a3.txt', multi_col=True).astype(np.float32)
prob_a4_df = loadFile('prob_a4.txt', multi_col=True).astype(np.float32)
rewards_df = loadFile('rewards.txt').astype(np.int32)
```

In [4]:

```
gamma = 0.9925
```

In [5]:

```
probs = np.zeros((81, 81, 4))
```

In [6]:

```
condensed_probs = [prob_a1_df, prob_a2_df, prob_a3_df, prob_a4_df]
for i in range(len(condensed_probs)):
    condensed_prob = condensed_probs[i]
    for prob in condensed_prob:
        probs[int(prob[0]) - 1][int(prob[1]) - 1][i] = prob[2]
```

In [22]:

```
valid_states = np.array([3, 11,12,15,16,17,20,22,23,24,26,29,30, 31, 34, 35, 39, 43, 48, 56
valid_states = valid_states - 1
```

In [7]:

```
def calcValFunc(policy):
    newProbMat = None
    for pi in range(policy.shape[0]):
        temp = probs[pi, :, policy[pi]]
        if newProbMat is None:
            newProbMat = temp
        else:
            newProbMat = np.vstack((newProbMat, temp))
    newProbMat = np.identity((newProbMat.shape[0])) - gamma * newProbMat

    return np.matmul(np.linalg.inv(newProbMat), rewards_df)
```

In [8]:

```
def calcNewValFunc(valueFunction, policy):
    newProbMat = None
    for pi in range(policy.shape[0]):
        temp = probs[pi, :, policy[pi]]
        if newProbMat is None:
            newProbMat = temp
        else:
            newProbMat = np.vstack((newProbMat, temp))
    return rewards_df + gamma * np.sum(newProbMat * valueFunction, axis=0)
```

In [9]:

```
def getNewPolicy(valueFunction, policy):
    for pi in range(policy.shape[0]):
        max_val = -np.inf
        max_action = -np.inf
        for action in range(4):
            temp = np.sum(probs[pi, :, action] * valueFunction)
            if max_val < temp:
                max_val = temp
                max_action = action
        policy[pi] = max_action
    return policy
```

In [23]:

```
def getRandomPolicy():
    policy = np.empty((81,), dtype=np.int32)
    for i in range(policy.shape[0]):
        policy[i] = random.randint(0, 3)
    return policy
```

In [24]:

```
policy = getRandomPolicy()
valueFunc = calcValFunc(policy)
opt_policy = getNewPolicy(valueFunc, policy)
newValueFunc = calcNewValFunc(valueFunc, opt_policy)#calcValFunc(opt_policy)#
```

In [25]:

```
i = 0
while not np.equal(newValueFunc, valueFunc).all():
    i += 1
    valueFunc = newValueFunc.copy()
    opt_policy = getNewPolicy(valueFunc, opt_policy)
    newValueFunc = calcValFunc(opt_policy)
print('Convergence in {} steps'.format(i))
```

```
Convergence in 5 steps
```

In [26]:

```
valueFunc
```

Out[26]:

```
array([   0.        ,    0.        ,  100.7010102 ,    0.        ,
          0.        ,    0.        ,    0.        ,    0.        ,
          0.        ,    0.        ,  102.37529123,  101.52367323,
          0.        ,    0.        ,  109.48995052,  110.40904748,
        111.33585966,    0.        ,    0.        ,  103.23464898,
          0.        ,  106.77828777,  107.67464526,  108.57850454,
          0.        ,  112.27045183,    0.        ,    0.        ,
        104.10123631,  104.97509851,  105.88855749,    0.        ,
          0.        ,  114.16323791,  113.21288929,    0.        ,
          0.        ,    0.        ,  103.78143159,    0.        ,
          0.        ,    0.        ,  115.12156408,    0.        ,
          0.        ,    0.        , -133.33333333,   90.98540322,
       -133.33333333,    0.        , -133.33333333,  116.08793478,
        122.02491933,    0.        ,    0.        ,   81.39950233,
         93.67166485,   95.1728646 ,  108.34262576,  109.58365563,
        123.64307385,  123.18224439,    0.        ,    0.        ,
       -133.33333333,   81.39950233, -133.33333333,    0.        ,
       -133.33333333,  125.24979123,  124.20738921,    0.        ,
          0.        ,    0.        ,    0.        ,    0.        ,
          0.        ,    0.        ,  133.33333333,    0.        ,
          0.        ])
```

In [27]:

```
def valueFuncIteration(valueFunction):
    newProbMat = None
    for pi in range(policy.shape[0]):
        max_val = -np.inf
        max_temp = None
        for a in range(4):
            temp = probs[pi, :, a]
            temp_val = np.sum(temp * valueFunction, axis=0)
            if max_val < temp_val:
                max_val = temp_val
                max_temp = temp
        if newProbMat is None:
            newProbMat = max_temp
        else:
            newProbMat = np.vstack((newProbMat, max_temp))
    return rewards_df + gamma * np.sum(newProbMat * valueFunction, axis=1)
```

In [28]:

```
policy = getRandomPolicy()
valueFunc2 = np.array([0]*policy.shape[0])
tempFunc = valueFuncIteration(valueFunc2)
```

In [29]:

```
i = 0
while np.sum(np.absolute(valueFunc2 - tempFunc)) > 1e-6:
    i += 1
    valueFunc2 = tempFunc
    tempFunc = valueFuncIteration(valueFunc2)
print('Convergence in {} steps'.format(i))
```

```
Convergence in 2304 steps
```

In [30]:

```
valueFunc2
```

Out[30]:

```
array([   0.        ,    0.        ,  100.70100671,    0.        ,
          0.        ,    0.        ,    0.        ,    0.        ,
          0.        ,    0.        ,  102.37528774,  101.52366974,
          0.        ,    0.        ,  109.48994702,  110.40904399,
        111.33585617,    0.        ,    0.        ,  103.23464549,
          0.        ,  106.77828428,  107.67464177,  108.57850104,
          0.        ,  112.27044834,    0.        ,    0.        ,
        104.10123282,  104.97509502,  105.888554  ,    0.        ,
          0.        ,  114.16323441,  113.2128858 ,    0.        ,
          0.        ,    0.        ,  103.78142811,    0.        ,
          0.        ,    0.        ,  115.12156059,    0.        ,
          0.        ,    0.        , -133.33332942,   90.98540012,
       -133.33332942,    0.        , -133.33332942,  116.08793128,
        122.02491563,    0.        ,    0.        ,   81.39949978,
         93.67166194,   95.17286168,  108.34262248,  109.58365233,
        123.64307016,  123.18224068,    0.        ,    0.        ,
       -133.33332942,   81.39949978, -133.33332942,    0.        ,
       -133.33332942,  125.24978753,  124.20738551,    0.        ,
          0.        ,    0.        ,    0.        ,    0.        ,
          0.        ,    0.        ,  133.33332942,    0.        ,
          0.        ])
```

In [35]:

```
action_map = {0: 'Left', 1: 'Up', 2: 'Right', 3: 'Down'}
```

In [40]:

```
for pi in range(policy.shape[0]):
    if pi in valid_states:
        print("""State: {} | Action: {} | Policy Iteration Value: {} | Value Iteration Valu
```

```
State: 3 | Action: Right | Policy Iteration Value: 100.7010102002196 | Value
Iteration Value: 100.70100670774
State: 11 | Action: Right | Policy Iteration Value: 102.37529123178209 | Val
ue Iteration Value: 102.3752877393026
State: 12 | Action: Up | Policy Iteration Value: 101.52367322965442 | Value
Iteration Value: 101.52366973717493
State: 15 | Action: Down | Policy Iteration Value: 109.48995051690076 | Valu
e Iteration Value: 109.48994702413327
State: 16 | Action: Down | Policy Iteration Value: 110.40904747912874 | Valu
e Iteration Value: 110.40904398636131
State: 17 | Action: Right | Policy Iteration Value: 111.33585966247094 | Val
ue Iteration Value: 111.33585616970353
State: 20 | Action: Right | Policy Iteration Value: 103.23464897839825 | Val
ue Iteration Value: 103.23464548591886
State: 22 | Action: Down | Policy Iteration Value: 106.7782877691457 | Value
Iteration Value: 106.77828427637834
State: 23 | Action: Down | Policy Iteration Value: 107.6746452597115 | Value
Iteration Value: 107.67464176694396
State: 24 | Action: Left | Policy Iteration Value: 108.57850453501321 | Valu
e Iteration Value: 108.57850104224573
State: 26 | Action: Right | Policy Iteration Value: 112.27045183161601 | Val
ue Iteration Value: 112.2704483388487
State: 29 | Action: Down | Policy Iteration Value: 104.10123631374762 | Valu
e Iteration Value: 104.10123282126823
State: 30 | Action: Down | Policy Iteration Value: 104.97509851098043 | Valu
e Iteration Value: 104.97509501850112
State: 31 | Action: Left | Policy Iteration Value: 105.88855749291497 | Valu
e Iteration Value: 105.8885540001552
State: 34 | Action: Right | Policy Iteration Value: 114.16323790726078 | Val
ue Iteration Value: 114.16323441449354
State: 35 | Action: Up | Policy Iteration Value: 113.21288929449881 | Value
Iteration Value: 113.21288580173153
State: 39 | Action: Left | Policy Iteration Value: 103.7814315891324 | Value
Iteration Value: 103.78142810703108
State: 43 | Action: Right | Policy Iteration Value: 115.1215640788626 | Valu
e Iteration Value: 115.12156058609541
State: 48 | Action: Left | Policy Iteration Value: 90.98540321957923 | Value
Iteration Value: 90.985400121465
State: 56 | Action: Down | Policy Iteration Value: 81.39950233342955 | Value
Iteration Value: 81.3994997783592
State: 57 | Action: Down | Policy Iteration Value: 93.67166484707683 | Value
Iteration Value: 93.67166194260669
State: 58 | Action: Down | Policy Iteration Value: 95.17286459965607 | Value
Iteration Value: 95.1728616815331
State: 59 | Action: Down | Policy Iteration Value: 108.34262576285258 | Valu
e Iteration Value: 108.34262247533641
State: 60 | Action: Down | Policy Iteration Value: 109.58365562763508 | Valu
e Iteration Value: 109.58365233013537
State: 61 | Action: Right | Policy Iteration Value: 123.64307385195802 | Val
ue Iteration Value: 123.64307016465737
State: 62 | Action: Right | Policy Iteration Value: 123.182244385098 | Value
```

```
Iteration Value: 123.18224068296979
State: 66 | Action: Left | Policy Iteration Value: 81.39950233342954 | Value
Iteration Value: 81.3994997783592
State: 70 | Action: Right | Policy Iteration Value: 125.24979123468198 | Val
ue Iteration Value: 125.24978753198937
State: 71 | Action: Up | Policy Iteration Value: 124.20738921318804 | Value
Iteration Value: 124.2073855105102
```

In [ ]:

```
```

9.5 $\Delta_k = \max_s \left[V_k(s) - V^{\pi}(s) \right]$

$$= \max_s \left[\left(R(s) + \gamma \sum_{s'} P(s'|s, \pi(s)) V_{k-1}(s') \right) - \left(R(s) + \gamma \sum_{s'} P(s'|s, \pi(s)) V^{\pi}(s') \right) \right]$$

$$= \gamma \max_s \left[\sum_{s'} P(s'|s, \pi(s)) V_{k-1}(s') - \sum_{s'} P(s'|s, \pi(s)) V^{\pi}(s') \right]$$

$$= \gamma \max_s \left[\sum_{s'} P(s'|s, \pi(s)) \left(V_{k-1}(s') - V^{\pi}(s') \right) \right]$$

$$= \gamma \max_{s,s'} \left[V_{k-1}(s') - V^{\pi}(s') \right]$$

~~$= \gamma \Delta_{k-1}$~~

~~$= \gamma \max$~~

$= \gamma \Delta_{k-1}$

if $\gamma \leq 1 \Rightarrow \Delta_k \leq \gamma \Delta_{k-1}$

and as $k \to \infty \Rightarrow \Delta_k \to 0$ for $\gamma < 1$

$\Delta_k = \gamma^k \Delta_0 \Rightarrow \Delta_k$ decays exponentially fast in no. of iterations k

$$\Rightarrow \lim_{k \to \infty} V_k(s) = V^{\pi}(s) \quad \forall s \in S$$

96

$$\mu = E[X] \qquad \{x_1, x_2, \ldots\}$$

$$\mu_k \leftarrow \mu_{k-1} + \alpha_k (x_k - \mu_{k-1})$$

$$\mu_0 = 0, \quad \alpha_k = \frac{1}{k}$$

a) $\sum_{k=1}^{\infty} \alpha_k^2 < \infty \quad \& \quad \sum_{k=} \alpha_k \geq \infty$

Using an identity:-

$$\int_1^{\infty} f(x)\,dx \leq \sum_{n=1}^{\infty} f(n) \leq f(1) + \int_1^{\infty} f(x)\,dx$$

Thus,

$$\sum_{k=1}^{\infty} \alpha_k = \sum_{k=1}^{\infty} \frac{1}{k} \geq \int_1^{\infty} \frac{1}{x}\,dx = \infty \quad \ldots (i)$$

Similarly,

$$\sum_{k=1}^{\infty} \alpha_k^2 = \sum_{k=1}^{\infty} \frac{1}{k^2} \leq 1 + \int_1^{\infty} \frac{1}{x^2}\, dx \leq 1+1 = 2$$

$$\Rightarrow \sum_{k=1}^{\infty} \alpha_k^2 < \infty \qquad \text{... (ii)}$$

b) $\mu_k \leftarrow \mu_{k-1} + \alpha_k (x_k - \mu_{k-1})$

To prove:

$$\mu_k = \frac{1}{k}(x_1 + x_2 \ldots + x_k) \qquad \text{for } \alpha = \frac{1}{k}$$

(i) Base case $\rightarrow \mu_1 = \mu_0 + \alpha_1 (x_1 - \mu_0)$

$$\mu_1 = x_1 \qquad \left[\because \alpha_1 = \frac{1}{1}\right]$$

(ii) Induction step:

Assume $\mu_{k-1} = \frac{1}{k-1}(x_1 + x_2 + \ldots x_{k-1})$

$$\Rightarrow \mu_k = \mu_{k-1} + \alpha_k (x_k - \mu_{k-1})$$

$$= \mu_{k-1}(1 - \alpha_k) + \alpha_k x_k$$

$$= \frac{1}{(k-1)}(x_1 + x_2 + \ldots x_{k-1}) \times \left(1 - \frac{1}{k}\right) + \frac{1}{k} x_k$$

$$\Rightarrow \mu_k = \frac{1}{k}(x_1 + x_2 + \ldots x_{k-1}) + \frac{1}{k} x_k$$

$$\boxed{\mu_k = \frac{1}{k}(x_1 + x_2 + \ldots \ldots x_k)}$$

Hence proved.