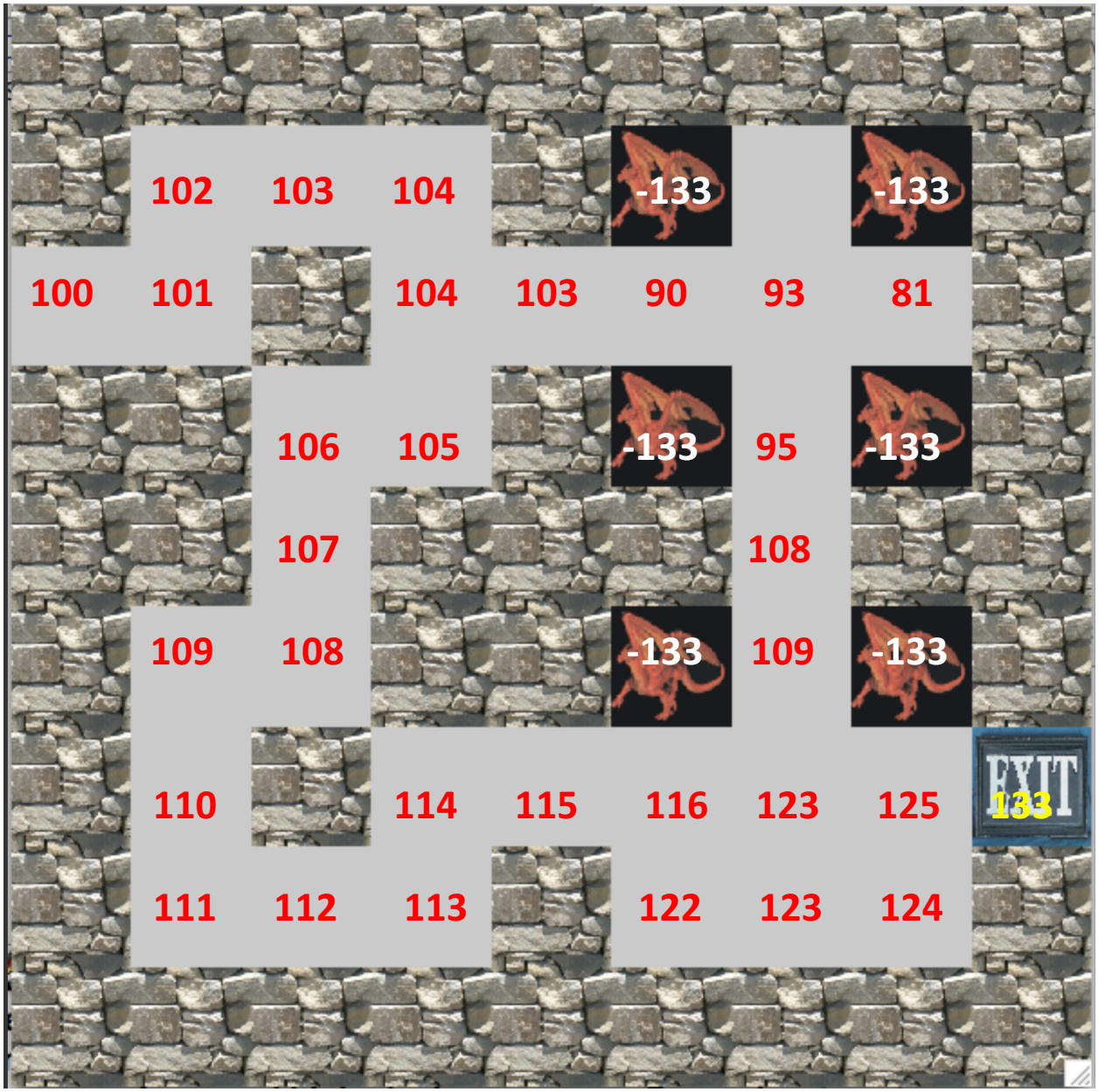
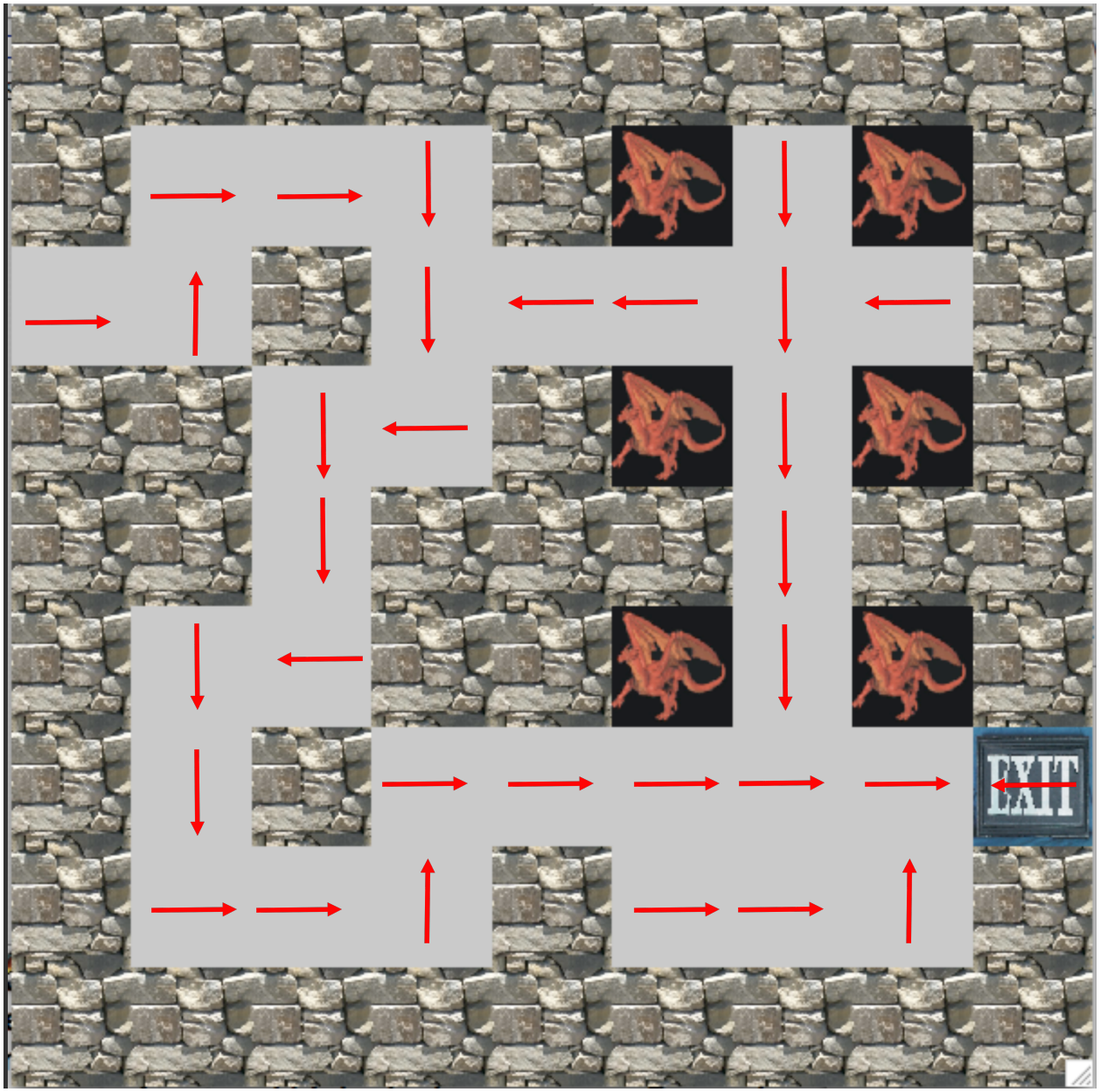
9.4) Policy and value iteration :

a) (i) Optimal value function : V*(s) visualized (rounded as an integer)



All the rocks have a value function of 0. The dragons have -133, and the exit state has +133.

(ii) Optimal policy –



The policy also dictates that at the rocks and at the dragons we move West (since our initial policy assumed that)

Optimal value function by policy iteration–

0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	102.38	103.23	104.10	0.00	-133.33	81.40	-133.33	0.00
100.70	101.52	0.00	104.98	103.78	90.99	93.67	81.40	0.00
0.00	0.00	106.78	105.89	0.00	-133.33	95.17	-133.33	0.00
0.00	0.00	107.67	0.00	0.00	0.00	108.34	0.00	0.00
0.00	109.49	108.58	0.00	0.00	-133.33	109.58	-133.33	0.00
0.00	110.41	0.00	114.16	115.12	116.09	123.64	125.25	133.33
0.00	111.34	112.27	113.21	0.00	122.02	123.18	124.21	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

(b)

The optimum state value function computed using value iteration is:

0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	102.36	103.22	104.09	0.00	-133.32	81.39	-133.32	0.00
100.69	101.51	0.00	104.96	103.77	90.98	93.66	81.39	0.00
0.00	0.00	106.77	105.88	0.00	-133.32	95.16	-133.32	0.00
0.00	0.00	107.66	0.00	0.00	0.00	108.33	0.00	0.00
0.00	109.48	108.57	0.00	0.00	-133.32	109.57	-133.32	0.00
0.00	110.40	0.00	114.15	115.11	116.08	123.63	125.24	133.32
0.00	111.32	112.26	113.20	0.00	122.01	123.17	124.20	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

As we can see the numbers are off by atmost 0.01

Hence we can confirm that the results by value iteration confirm with those by policy iterations.

Source code:

```
# coding: utf-8

# In[18]:


from collections import defaultdict
import numpy as np


# In[54]:


with open('data/rewards.txt') as f:
    temp=f.read().strip().split()
    rewards = np.zeros((81,1),dtype=np.float)
    for i in enumerate(temp):
        rewards[int(i[0])]=int(i[1])


# In[33]:


with open('data/prob_a1.txt') as f:
    temp=f.readlines()
    proba1 = np.zeros((81,81),dtype=np.float)
    for i in temp:
        t=i.strip().split()
        proba1[int(t[0])-1,int(t[1])-1]=float(t[2])


# In[34]:
```

```
with open('data/prob_a2.txt') as f:
    temp=f.readlines()
    proba2 = np.zeros((81,81),dtype=np.float)
    for i in temp:
        t=i.strip().split()
        proba2[int(t[0])-1,int(t[1])-1]=float(t[2])


# In[35]:


with open('data/prob_a3.txt') as f:
    temp=f.readlines()
    proba3 = np.zeros((81,81),dtype=np.float)
    for i in temp:
        t=i.strip().split()
        proba3[int(t[0])-1,int(t[1])-1]=float(t[2])


# In[36]:


with open('data/prob_a4.txt') as f:
    temp=f.readlines()
    proba4 = np.zeros((81,81),dtype=np.float)
    for i in temp:
        t=i.strip().split()
        proba4[int(t[0])-1,int(t[1])-1]=float(t[2])


# In[149]:


policy=np.zeros((81,1),dtype=np.int)
gamma=0.9925
vp=np.zeros((81,1),dtype=np.float)
vpbefore=np.ones((81,1),dtype=np.float)
ind=1
while(max(abs(vp-vpbefore))>0.0001):
    print ind
    ind+=1

    ppi=np.zeros((81,81),dtype=np.float)
    for s in range(81):
        for sd in range(81):
            if policy[s]==0:
                ppi[s,sd]=proba1[s,sd]
            if policy[s]==1:
                ppi[s,sd]=proba2[s,sd]
            if policy[s]==2:
                ppi[s,sd]=proba3[s,sd]
            if policy[s]==3:
                ppi[s,sd]=proba4[s,sd]

    vpbefore=vp.copy()
    vp=np.linalg.inv(np.identity(81)-
ppi*gamma).dot(rewards)

    for s in range(81):
        temp=[]

        sump=0
        for sd in range(81):
            sump+=proba1[s,sd]*vp[sd,0]
        temp.append(sump)

        sump=0
        for sd in range(81):
            sump+=proba2[s,sd]*vp[sd,0]
        temp.append(sump)

        sump=0
        for sd in range(81):
            sump+=proba3[s,sd]*vp[sd,0]
        temp.append(sump)

        sump=0
        for sd in range(81):
            sump+=proba4[s,sd]*vp[sd,0]
        temp.append(sump)

        policy[s,0]=temp.index(max(temp))


# In[137]:


vt=vp.reshape((9,9)).transpose()
for i in range(9):
    for j in range(9):
        print '{:7.2f}'.format(vt[i,j]),
    print


# In[118]:


for i in vp.reshape((9,9)).transpose().tolist():
    for j in i:
        print '{0:.3f}'.format(j),
```

```
    print '\n'


# In[104]:


for i in range(vp.shape[0]):
    print i+1,'\t',vp[i,0],'\t\t',policy[i,0]


# In[101]:


for i in policy.reshape((9,9)).transpose().tolist():
    print map(int,i)


# In[151]:


policy=np.zeros((81,1),dtype=np.int)
gamma=0.9925
vp=np.zeros((81,1),dtype=np.float)
vpbefore=np.ones((81,1),dtype=np.float)
ind=1

while(max(abs(vp-vpbefore))>0.00001):
    print ind
    ind+=1

    vpbefore=vp.copy()
    for s in range(81):

        temp=[]

        sump=0
        for sd in range(81):
            sump+=proba1[s,sd]*vp[sd,0]
        temp.append(sump)

        sump=0
        for sd in range(81):
            sump+=proba2[s,sd]*vp[sd,0]
        temp.append(sump)

        sump=0
        for sd in range(81):
            sump+=proba3[s,sd]*vp[sd,0]
        temp.append(sump)

        sump=0
        for sd in range(81):
            sump+=proba4[s,sd]*vp[sd,0]
        temp.append(sump)

        vp[s,0]=rewards[s,0]+gamma*max(temp)


# In[148]:


vt=vp.reshape((9,9)).transpose()
for i in range(9):
    for j in range(9):
        print '{:7.4f}'.format(vt[i,j]),
    print


# In[150]:


vt=vp.reshape((9,9)).transpose()
for i in range(9):
    for j in range(9):
        print '{:7.4f}'.format(vt[i,j]),
    print
```