Programming Assignment 4 & 5 Report
Wei Fan

Programming Assignment 4:

How to run my code(for assignment 4):
Open 'PA4.ipynb'
Run functions run50episodes() and run50update_v()

Notation: (Sometimes for a convenience to read, I denote state like 'RU8p' to 'S1')

RU8p	S1
TU10p	S2
RU10p	S3
RD10p	S4
RU8a	S5
RD8a	S6
TU10a	S7
RU10a	S8
RD10a	S9
TD10a	S10

Implement detail
Data structure used to store the models
I use a class 'State' to store each state. 'Node_name' is like 'S1','S2'….which allows me to do a quick review on my code. 'Situation', 'homework', 'time' is like 'R', 'D', '8p'… 'isExit' represents is the state is a terminal state(11am class begins).

```
class State:
    node_name = ""
    situation = ""
    homework = ""
    time = ""
    isExit = 0

    def __init__(self,node_name, situation, homework, time, isExit):
        self.node_name = node_name
        self.situation = situation
        self.homework = homework
        self.time = time
        self.isExit = isExit
```

And I create these states.

```
s1 = State("S1", "R", "U", "8p", 0)
s2 = State("S2", "T", "U", "10p",0)
s3 = State("S3", "R", "U", "10p",0)
s4 = State("S4", "R", "D", "10p",0)
s5 = State("S5", "R", "U", "8a",0)
s6 = State("S6", "R", "D", "8a",0)
s7 = State("S7", "T", "U", "10a",0)
s8 = State("S8", "R", "U", "10a",0)
s9 = State("S9", "R", "D", "10a",0)
s10 = State("S10", "T", "D", "10a",0)
s11 = State("S11", "Class", "Begins", "11a",1)
```

To store the 'edges' in the Markov decision process, I create a structure called 'transition' (state, action, result-state, probability, rewards)

```
# (state, action, result-state, probability, rewards)
transition = [
    [s1, "P", s2, 1, 2],
    [s1, "R", s3, 1, 0],
    [s1, "S", s4, 1, -1],
    [s2, "R", s5, 1, 0],
    [s2, "P", s8, 1, 2],
    [s3, "R", s5, 1, 0],
    [s3, "S", s6, 1, -1],
    [s3, "P", s5, 0.5, 2],
    [s3, "P", s8, 0.5, 2],
    [s4, "R", s6, 1, 0],
    [s4, "P", s6, 0.5, 2],
    [s4, "P", s9, 0.5, 2],
    [s5, "P", s7, 1, 2],
    [s5, "R", s8, 1, 0],
    [s5, "S", s9, 1, -1],
    [s6, "R", s9, 1, 0],
    [s6, "P", s10, 1, 2],
    [s7, "P", s11, 1, -1],
    [s8, "P", s11, 1, 0],
    [s9, "P", s11, 1, 4],
    [s10, "P", s11, 1, 3],
    [s7, "R", s11, 1, -1],
    [s8, "R", s11, 1, 0],
    [s9, "R", s11, 1, 4],
    [s10, "R", s11, 1, 3],
    [s7, "S", s11, 1, -1],
```

And a dict called 'v' to store value of each state.

```
v = {
    s1:0,
    s2:0,
    s3:0,
    s4:0,
    s5:0,
    s6:0,
    s7:0,
    s8:0,
    s9:0,
    s10:0,
    s11:0
}
```

Here is the function to pick an action considering the possibility and random policy.

```
def get_random_action(state):
    trans = get_transitions(state)
    pool = []
    for t in trans:
        if t[3] == 1:
            pool.append(t[1])
            pool.append(t[1])
        elif t[3] == 0.5:
            pool.append(t[1])
    index = random.randint(0,len(pool)-1)
    return pool[index]
```

And for each iteration, we need to calculate vk to update value function using rewards.

```
def update_v():
    for vk in v.keys():
        trans = get_transitions(vk)
        num = get_number(trans)
        rewards = 0
        for t in trans:
            resultstate = t[2]
            v_resultstate = v[resultstate]
            r = t[4]
            rewards += ((r + v_resultstate) * t[3] / num)
        v[vk] = rewards
    return v
```

Here is the function which allows 50 episodes selecting actions from s1 to s11(11am class begins). The function keeps picking up actions until it meets the terminal state.

```
def run50episodes():
    sum_reward = 0
    for i in range(0,50):
        curstate = State("S1", "R", "U", "8p", 0)
        sequence = []
        nodes = []
        actions = []
        r = 0
        rewards = []
        nodes.append(curstate)
        sequence.append(curstate.node_name)
        while curstate.isExit!=1:
            action = get_random_action(curstate)
            actions.append(action)
            result_state = get_resultstate(curstate, action)
            r += get_rewards(curstate, action, result_state)
            rewards.append(r)
            sequence.append(result_state.node_name)
            nodes.append(result_state)
            curstate = result_state
        re = []
        for i in range(0,len(nodes)-1):
            re.append([get_state_name(nodes[i]), actions[i], rewards[i]])
        for x in re:
            print (x, end='')
        print ([get_state_name(s11),None, rewards[-1]], end='')
        print ("  ",end='')
        print ("Sum rewards is ", end='')
```

Results

Result of running 50 episodes sequences:

```
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'P', 3]['TD 10a', 'S', 6]['ClassBegins 11a', None, 6]  Sum rewards is 6
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'R', 4]['ClassBegins 11a', None, 4]  Sum rewards is 4
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'S', 4]['ClassBegins 11a', None, 4]  Sum rewards is 4
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'S', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'R', 2]['RU 10a', 'S', 2]['ClassBegins 11a', None, 2]  Sum rewards is 2
['RU 8p', 'S', -1]['RD 10p', 'R', -1]['RD 8a', 'R', -1]['RD 10a', 'P', 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'P', 4]['TU 10a', 'S', 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'R', 4]['ClassBegins 11a', None, 4]  Sum rewards is 4
['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'S', 1]['RD 10a', 'R', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'R', 2]['RU 10a', 'R', 2]['ClassBegins 11a', None, 2]  Sum rewards is 2
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'P', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'R', -1]['RD 10a', 'S', 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'R', 0]['RU 10p', 'R', 0]['RU 8a', 'P', 2]['TU 10a', 'S', 1]['ClassBegins 11a', None, 1]  Sum rewards is 1
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'P', 1]['TD 10a', 'R', 4]['ClassBegins 11a', None, 4]  Sum rewards is 4
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'R', 2]['RU 10a', 'R', 2]['ClassBegins 11a', None, 2]  Sum rewards is 2
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'R', -1]['RD 10a', 'R', 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'P', 4]['TU 10a', 'P', 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'P', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'S', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'S', 1]['RD 10a', 'R', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'R', 2]['RU 10a', 'P', 2]['ClassBegins 11a', None, 2]  Sum rewards is 2
['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'R', 2]['RU 10a', 'S', 2]['ClassBegins 11a', None, 2]  Sum rewards is 2
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'P', 3]['TD 10a', 'R', 6]['ClassBegins 11a', None, 6]  Sum rewards is 6
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'S', 4]['ClassBegins 11a', None, 4]  Sum rewards is 4
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'S', 1]['RD 10a', 'P', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'S', 1]['RD 10a', 'P', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
```

We can interpret the output like that:

Take the row0 `['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'P', 3]['TD 10a', 'S', 6]['ClassBegins 11a', None, 6]  Sum rewards is 6`

as an example, we can see: from ‘RU8p’, take S action, the result state is ‘RU10p’ and -1 represents the rewards after taking the corresponding action. The sum rewards of this row is 6, which are called ’the Return’.
Complete sequences of experiences are included in the appendix.

The run50episodes() function returns an average Return:

```
        sum_reward += rewards[-1]
    return sum_reward/50
```

And the average return of this 50 cases it returns is 3.82.

```
['RU 8p', 'S', -1]['RD 10p', 'R', -1]['RD 8a',
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', '
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', '

Out[39]: 3.82
```

Then we need to calculate the values of each state using ‘update_v()’ function(which has been described above)
I try to run it for 50 times but it seems v doesn’t change since k reaches to 5.
The initial value of each state(k=1) is 0.

```
def run50update_v():
    for i in range(1,51):
        print ("k=", end='')
        print (i, end='')
        print (print_v_format(v))
        update_v()
```

```
k=1{'S1': 0, 'S2': 0, 'S3': 0, 'S4': 0, 'S5': 0, 'S6': 0, 'S7': 0, 'S8': 0, 'S9': 0, 'S10': 0, 'S11': 0}
k=2{'S1': 0.3333333333333333, 'S2': 1.0, 'S3': 0.3333333333333333, 'S4': 1.0, 'S5': 0.3333333333333333, 'S6': 1.0, 'S7': -1.0, 'S8': 0.0, 'S
9': 4.0, 'S10': 3.0, 'S11': 0}
k=3{'S1': 1.1111111111111112, 'S2': 1.1666666666666667, 'S3': 0.8333333333333333, 'S4': 2.75, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.
0, 'S8': 0.0, 'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=4{'S1': 1.9166666666666665, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=5{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=6{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=7{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=8{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=9{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=10{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=11{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=12{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=13{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
k=14{'S1': 3.5138888888888884, 'S2': 1.6666666666666665, 'S3': 2.5, 'S4': 5.375, 'S5': 1.3333333333333333, 'S6': 4.5, 'S7': -1.0, 'S8': 0.0,
'S9': 4.0, 'S10': 3.0, 'S11': 0}
```

To map the status node name with names in the assignment requirement, the result is shown as below.

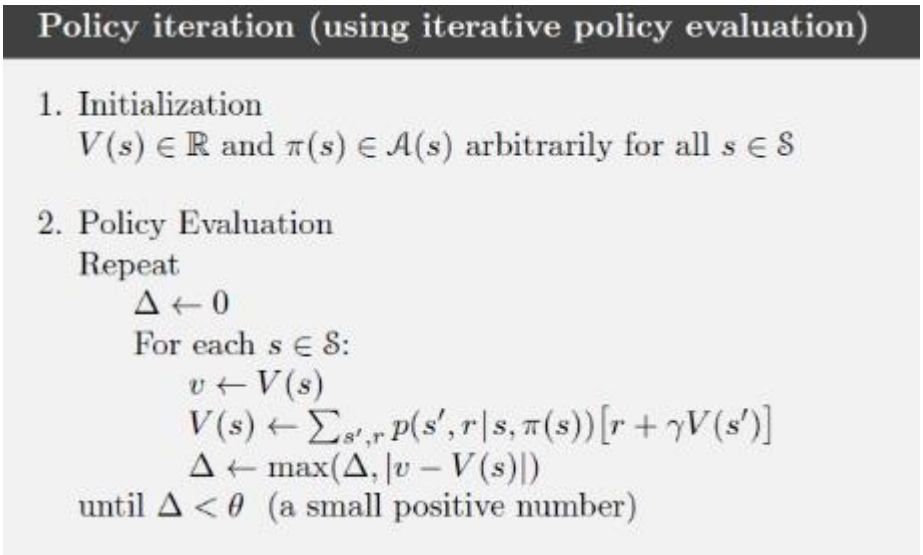
k=1{'RU8p': 0, 'TU10p': 0, 'RU10p': 0, 'RD10p': 0, 'RU8a': 0, 'RD8a': 0, 'TU10a': 0, 'RU10a': 0, 'RD10a': 0, 'TD10a': 0, 'ClassBegins11a': 0}
k=2{'RU8p': 0.3333333333333333, 'TU10p': 1.0, 'RU10p': 0.3333333333333333, 'RD10p': 1.0, 'RU8a': 0.3333333333333333, 'RD8a': 1.0, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=3{'RU8p': 1.1111111111111112, 'TU10p': 1.1666666666666667, 'RU10p': 0.8333333333333333, 'RD10p': 2.75, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=4{'RU8p': 1.9166666666666665, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=5{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=6{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=7{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=8{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=9{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=10{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=11{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=12{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=13{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0, 'ClassBegins11a': 0}
k=14{'RU8p': 3.5138888888888884, 'TU10p': 1.6666666666666665, 'RU10p': 2.5, 'RD10p': 5.375, 'RU8a': 1.3333333333333333, 'RD8a': 4.5, 'TU10

We can see that when k >5, the v doesn't change.

Programming 5

How to run my code(for assignment 5 (for only policy evaluation version)):
Open 'PA5.ipynb'
Execute functions policy_evalution() and print_v_format_name(v)

How to run my code(for assignment 5+ (for policy evaluation PLUS policy improvement version)):
Open 'PA6.ipynb'
Execute functions policy_improvement()



I follow this pseudo-code from slides(Lecture 15 page 22). The professor says all policies are Party and we do not need to perform the policy improvement.

I write this function to get the result states and corresponding rewards from a state when taking 'Party' function.

```
def party_time(state):
    result = []
    for t in transition:
        if t[0]==state and t[1]=="P":
            result.append(t)
    return result
```

Code for policy evaluation

```
def policy_evalution():
    delta = 1
    theta=0.001
    k = 1
    while delta>=theta:
        delta = 0
        for s in [s1,s2,s3,s4,s5,s6,s7,s8,s9,s10]:
            trans = get_transitions(s)
            pi = party_time(s)
            number = get_number(pi)
            v1 = v[s]
            v_tmp = 0
            for p in pi:
                result_node = p[2]
                v_tmp += (v[result_node] + p[4])*(p[3]/number)
            v[s] = v_tmp
            delta = max(delta, abs(v1 - v_tmp))
        print("interation ", end='')
        print (k, end='')
        print ("    policy", end='')
        print (" P")
        print (print_v_format_name(v))
        k += 1
    return v
```

Result:

```
interation 1    policy P
{'RU8p': 2.0, 'TU10p': 2.0, 'RU10p': 2.0, 'RD10p': 2.0, 'RU8a': 2.0, 'RD8a': 2.0, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0,
'ClassBegins11a': 0}
interation 2    policy P
{'RU8p': 4.0, 'TU10p': 2.0, 'RU10p': 3.0, 'RD10p': 5.0, 'RU8a': 1.0, 'RD8a': 5.0, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0,
'ClassBegins11a': 0}
interation 3    policy P
{'RU8p': 4.0, 'TU10p': 2.0, 'RU10p': 2.5, 'RD10p': 6.5, 'RU8a': 1.0, 'RD8a': 5.0, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0,
'ClassBegins11a': 0}
```

```
Out[16]: {'RU8p': 4.0,
          'TU10p': 2.0,
          'RU10p': 2.5,
          'RD10p': 6.5,
          'RU8a': 1.0,
          'RD8a': 5.0,
          'TU10a': -1.0,
          'RU10a': 0.0,
          'RD10a': 4.0,
          'TD10a': 3.0,
          'ClassBegins11a': 0}
```

And the professor says completing policy improvement can be a bonus.
So I do the policy improvement by following the pseudo-code from the slides.

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
      old-action $\leftarrow \pi(s)$
      $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
      If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

I write a function to find the max value action of a state.

```
def find_max_action(state):
    trans = get_transitions(state)
    actions = find_actions(state)
    max_value = -100
    max_act = []
    l = None
    for act in actions:
        tmp = get_tran_by_action(state, act)
        numbers = get_number(trans)
        r = 0
        for t in tmp:
            r += (t[3]/numbers)*(t[4] + v[t[2]])
        if r == max_value:
            max_act.append(act)
        if r>max_value:
            max_value = r
            max_act=[act]
            l = tmp
    return max_act, l
```

And I create some structures to store the policy.

```
def policy_init2():
    pi = dict()
    for s in [s1,s2,s3,s4,s5,s6,s7,s8,s9,s10]:
        pi[s] = party_time(s)
    return pi

def policy_init():
    pi = dict()
    for s in [s1,s2,s3,s4,s5,s6,s7,s8,s9,s10]:
        pi[s] = "P"
    return pi

PI = policy_init()
PO = policy_init2()
```

I modify the policy evaluation function.

```
def policy_evalution():
    delta = 1
    theta=0.001
    k = 1
    while delta>=theta:
        delta = 0
        for s in [s1,s2,s3,s4,s5,s6,s7,s8,s9,s10]:
            trans = get_transitions(s)
            pi = PO[s]
            number = get_number(pi)
            v1 = v[s]
            v_tmp = 0
            for p in pi:
                result_node = p[2]
                v_tmp += (v[result_node] + p[4])*(p[3]/number)
            v[s] = v_tmp
            delta = max(delta, abs(v1 - v_tmp))
        k += 1
    print (print_v_format_name(v))
    return v
```

And here is the policy improvement method.

```
def policy_improvement():
    policy_evalution()
    policy_stable = True
    for s in [s1,s2,s3,s4,s5,s6,s7,s8,s9,s10]:
        old_action = PI[s]
        PI[s] = find_max_action(s)[0]
        PO[s] = find_max_action(s)[1]
        print (PI[s])
        if old_action != PI[s]:
            policy_stable = False

    if policy_stable == True:
        return  v, PI
    else:
        policy_improvement()
```

Here is the result for each iteration:

```
{'RU8p': 4.0, 'TU10p': 2.0, 'RU10p': 2.5, 'RD10p': 6.5, 'RU8a': 1.0, 'RD8a': 5.0, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0,
'ClassBegins11a': 0}
['S']
['P']
['S']
['P']
['S']
['P']
['P', 'S', 'R']
['P', 'S', 'R']
['P', 'S', 'R']
['P', 'S', 'R']
{'RU8p': 5.5, 'TU10p': 2.0, 'RU10p': 4.0, 'RD10p': 6.5, 'RU8a': 3.0, 'RD8a': 5.0, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0,
'ClassBegins11a': 0}
['S']
['R']
['S']
['P']
['S']
['P']
['P', 'S', 'R']
['P', 'S', 'R']
['P', 'S', 'R']
['P', 'S', 'R']
{'RU8p': 5.5, 'TU10p': 3.0, 'RU10p': 4.0, 'RD10p': 6.5, 'RU8a': 3.0, 'RD8a': 5.0, 'TU10a': -1.0, 'RU10a': 0.0, 'RD10a': 4.0, 'TD10a': 3.0,
'ClassBegins11a': 0}
['S']
['R']
['S']
```

```
['P']
['S']
['P']
['P', 'S', 'R']
['P', 'S', 'R']
['P', 'S', 'R']
['P', 'S', 'R']
```

[p,s,r]means you can take any actions.

Appendix:

```
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'P', 3]['TD 10a', 'S
', 6]['ClassBegins 11a', None, 6]  Sum rewards is 6
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'R', 4]['ClassBegins
11a', None, 4]  Sum rewards is 4
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'S', 4]['ClassBegins
11a', None, 4]  Sum rewards is 4
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'S
', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'R', 2]['RU 10a', 'S',
 2]['ClassBegins 11a', None, 2]  Sum rewards is 2
['RU 8p', 'S', -1]['RD 10p', 'R', -1]['RD 8a', 'R', -1]['RD 10a', '
P', 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'P', 4]['TU 10a', 'S',
 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'R', 4]['ClassBegins
11a', None, 4]  Sum rewards is 4
['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'S', 1]['RD 10a', 'R',
 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'R', 2]['RU 10a', 'R',
 2]['ClassBegins 11a', None, 2]  Sum rewards is 2
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'P
', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'R', -1]['RD 10a', 'S
', 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'R', 0]['RU 10p', 'R', 0]['RU 8a', 'P', 2]['TU 10a', 'S',
 1]['ClassBegins 11a', None, 1]  Sum rewards is 1
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'P', 1]['TD 10a', 'R
', 4]['ClassBegins 11a', None, 4]  Sum rewards is 4
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'R', 2]['RU 10a', 'R',
 2]['ClassBegins 11a', None, 2]  Sum rewards is 2
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'R', -1]['RD 10a', 'R
', 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'P', 4]['TU 10a', 'P',
 3]['ClassBegins 11a', None, 3]  Sum rewards is 3
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'P
', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'S
', 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'S', 1]['RD 10a', 'R',
 5]['ClassBegins 11a', None, 5]  Sum rewards is 5
['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'R', 2]['RU 10a', 'P',
 2]['ClassBegins 11a', None, 2]  Sum rewards is 2
```

['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'R', 2]['RU 10a', 'S', 2]['ClassBegins 11a', None, 2] Sum rewards is 2
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'P', 3]['TD 10a', 'R', 6]['ClassBegins 11a', None, 6] Sum rewards is 6
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'S', 4]['ClassBegins 11a', None, 4] Sum rewards is 4
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'S', 1]['RD 10a', 'P', 5]['ClassBegins 11a', None, 5] Sum rewards is 5
['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'S', 1]['RD 10a', 'P', 5]['ClassBegins 11a', None, 5] Sum rewards is 5
['RU 8p', 'S', -1]['RD 10p', 'R', -1]['RD 8a', 'R', -1]['RD 10a', 'P', 3]['ClassBegins 11a', None, 3] Sum rewards is 3
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'R', 2]['RU 10a', 'R', 2]['ClassBegins 11a', None, 2] Sum rewards is 2
['RU 8p', 'S', -1]['RD 10p', 'R', -1]['RD 8a', 'P', 1]['TD 10a', 'P', 4]['ClassBegins 11a', None, 4] Sum rewards is 4
['RU 8p', 'S', -1]['RD 10p', 'R', -1]['RD 8a', 'P', 1]['TD 10a', 'S', 4]['ClassBegins 11a', None, 4] Sum rewards is 4
['RU 8p', 'S', -1]['RD 10p', 'R', -1]['RD 8a', 'R', -1]['RD 10a', 'P', 3]['ClassBegins 11a', None, 3] Sum rewards is 3
['RU 8p', 'S', -1]['RD 10p', 'R', -1]['RD 8a', 'R', -1]['RD 10a', 'S', 3]['ClassBegins 11a', None, 3] Sum rewards is 3
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'P', 3]['TD 10a', 'P', 6]['ClassBegins 11a', None, 6] Sum rewards is 6
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'S', 4]['ClassBegins 11a', None, 4] Sum rewards is 4
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'P', 4]['ClassBegins 11a', None, 4] Sum rewards is 4
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'R', -1]['RD 10a', 'R', 3]['ClassBegins 11a', None, 3] Sum rewards is 3
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'R', 5]['ClassBegins 11a', None, 5] Sum rewards is 5
['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'S', 1]['RD 10a', 'S', 5]['ClassBegins 11a', None, 5] Sum rewards is 5
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'P', 4]['ClassBegins 11a', None, 4] Sum rewards is 4
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'S', 1]['RD 10a', 'S', 5]['ClassBegins 11a', None, 5] Sum rewards is 5
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'S', 5]['ClassBegins 11a', None, 5] Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'P', 1]['TD 10a', 'S', 4]['ClassBegins 11a', None, 4] Sum rewards is 4
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'R', -1]['RD 10a', 'S', 3]['ClassBegins 11a', None, 3] Sum rewards is 3

['RU 8p', 'R', 0]['RU 10p', 'R', 0]['RU 8a', 'S', -1]['RD 10a', 'P', 3]['ClassBegins 11a', None, 3] Sum rewards is 3
['RU 8p', 'R', 0]['RU 10p', 'P', 2]['RU 8a', 'S', 1]['RD 10a', 'R', 5]['ClassBegins 11a', None, 5] Sum rewards is 5
['RU 8p', 'R', 0]['RU 10p', 'S', -1]['RD 8a', 'R', -1]['RD 10a', 'R', 3]['ClassBegins 11a', None, 3] Sum rewards is 3
['RU 8p', 'P', 2]['TU 10p', 'R', 2]['RU 8a', 'P', 4]['TU 10a', 'P', 3]['ClassBegins 11a', None, 3] Sum rewards is 3
['RU 8p', 'S', -1]['RD 10p', 'R', -1]['RD 8a', 'R', -1]['RD 10a', 'S', 3]['ClassBegins 11a', None, 3] Sum rewards is 3
['RU 8p', 'S', -1]['RD 10p', 'P', 1]['RD 8a', 'R', 1]['RD 10a', 'R', 5]['ClassBegins 11a', None, 5] Sum rewards is 5
['RU 8p', 'P', 2]['TU 10p', 'P', 4]['RU 10a', 'R', 4]['ClassBegins 11a', None, 4] Sum rewards is 4