Assignment2 code

- name : FangZhou
- Studentid : 2018124048

Question 1 (RL)

Q1_RL.py

```
"""
  take Reinforcement Learning in two-dimension Array based on Q-Learning
  Lead the robot training for several times, and the robot can find the
shortest path
  FangZhou 2018124048
"""
import random
# some global var
MAX_EPISODES = 200   # 学习的次数
GOLD_STATE = (5, 4)      # 目标状态
INIT_STATE = (7, 3)      # 初始状态
GOLD_REWORD = 10
STATES = [(2, 6), (2, 5), (2, 4),
          (2, 3), (2, 2), (3, 6),
          (3, 3), (3, 2), (4, 6), (4, 5), (4, 4),
          (4, 2), (5, 6), (5, 4), (5, 2), (6, 6), (6, 3),
          (6, 2), (7, 6), (7, 5), (7, 4), (7, 3), (7, 1),
          (8, 2), (8, 1)]         # 状态的形式,只有在这些状态中的位置, 才能被robot
访问。 为list形式的
ACTIONS = ["up", "down", "left", "right"]  # 动作分类,  为list形式
GREEDY_RATE = 0.9    # 贪婪值，控制着在选则行为的时候，有0.9的概率会按值选择，有0.1
的概率会随机选择
ALPHA = 0.01      # 学习速率
GAMMA = 0.95      # 衰减因子，控制着之后的reword对此刻action选择的影响衰减比例


def init_q_table(the_states, the_actions):
    """
    根据the_states , the_actions 来初始化一个q_table
    q_table的形式为:
    {
        (2, 6):          #不管状态(2, 6)能否采取这种形式的action，都先初始化
        {
            "up" : xxx
            "down" : xxx
            "left" : xxx
```

```
            "right" : xxx
        }
        ...
    }
    :return: 一个Qtable
    """
    my_q_table = dict()
    for state in the_states:
        my_q_table[state] = dict()
        for action in the_actions:
            my_q_table[state][action] = 0.00
    return my_q_table


def choose_action(q_table, current_state, greedy_rate = GREEDY_RATE):
    """
    根据一个q_table 来选择 current_state下的应该选择的action
    :param q_table:
    :param current_state: 当前的状态
    :return: 一个action
    """
    # 在10%的概率下 或者 当前q_table对应的全为0的情况下， 则随机选择一个action
    if random.randint(0, 100) > greedy_rate * 100 or
all_zero(q_table[current_state]):
        action_length = len(ACTIONS)
        return ACTIONS[random.randint(0, action_length-1)]
    else:
        # 此时为按值选取
        ac, val = max_action_val(q_table[current_state])
        return ac


def get_env_feedback(current_state, action):
    """
    这个feedback并不是q_table里面的值
    :param state:
    :param action:
    :return: 返回在current_state下采取action后获得的reword， 以及下一步的state
    """
    x_axis = current_state[0]
    y_axis = current_state[1]
    if action == "up":
        # 往上走
        y_axis += 1
    elif action == "down":
        # 往下走
        y_axis -= 1
    elif action == "left":
        # 往左走
```

```
        x_axis -= 1
    else:
        # 往右走
        x_axis += 1
    next_state = (x_axis, y_axis)
    if next_state not in STATES:
        # 撞到墙壁，状态不变
        return 0, current_state
    else:
        # 没有撞到墙壁
        if is_terminal(next_state):
            return GOLD_REWORD, next_state
        else:
            return 0, next_state


def is_terminal(the_state):
    """判断此时是不是终点"""
    if the_state == GOLD_STATE:
        return True
    else:
        return False


def max_action_val(state_action):
    """
    选择出这个dict中Reword最大的action
    :param q_table:
    :param current_state:
    :return: 最大的值对应的action, 以及相应的val
    """
    max_key = 0
    max_val = 0
    for key, val in state_action.items():
        if not max_key:
            max_key = key
            max_val = val
            continue
        if val > max_val:
            max_key = key
            max_val = val
    return max_key, max_val


def all_zero(state_action):
    """
    :param state_action: 传入当前的state的对应的q_table ,形式如下
        {
            "top" : 0.00,
```

```
            "down" : 0.00,
            "left" : 0.00,
            "right" : 0.00
        }
    :return: true(全0) or false(非全0)
    """
    for val in state_action.values():
        if val:
            return False
    return True


def rl():
    """
    主循环函数
    :return:
    """
    q_table = init_q_table(STATES, ACTIONS)
    for episode in range(MAX_EPISODES):
        s = INIT_STATE  # 初始化状态
        step = 0
        while not is_terminal(s):
            # 在还不是终点的时候
            action = choose_action(q_table, s)
            R, next_s = get_env_feedback(s, action)
            ac, val = max_action_val(q_table[next_s])
            # 更新Q(s, a)值
            gap = R + GAMMA * val - q_table[s][ac]
            q_table[s][action] += ALPHA * gap
            s = next_s
            step += 1
        print("episode : "+str(episode)+" reach gold steps: " + str(step))
    return q_table


def get_final_path(q_table):
    """
    根据最后的q_table, 得出最后的path
    :param q_table:
    :return:
    """
    s = INIT_STATE
    print(s)
    while not is_terminal(s):
        action = choose_action(q_table, s, 1)
        R, next_s = get_env_feedback(s, action)
        s = next_s
        print(s)
```

```
if __name__ == "__main__":
    q_table = rl()
    print("the final path is: ")
    get_final_path(q_table)
```

```
"""
output:
trainng times : 200
"""
episode : 0 reach gold steps: 1569
episode : 1 reach gold steps: 182
episode : 2 reach gold steps: 434
episode : 3 reach gold steps: 275
episode : 4 reach gold steps: 22
episode : 5 reach gold steps: 69
episode : 6 reach gold steps: 11
episode : 7 reach gold steps: 22
episode : 8 reach gold steps: 20
episode : 9 reach gold steps: 12
episode : 10 reach gold steps: 9
episode : 11 reach gold steps: 9
episode : 12 reach gold steps: 9
episode : 13 reach gold steps: 10
episode : 14 reach gold steps: 11
episode : 15 reach gold steps: 9
episode : 16 reach gold steps: 10
episode : 17 reach gold steps: 12
episode : 18 reach gold steps: 9
episode : 19 reach gold steps: 9
episode : 20 reach gold steps: 9
episode : 21 reach gold steps: 9
episode : 22 reach gold steps: 11
episode : 23 reach gold steps: 10
episode : 24 reach gold steps: 9
episode : 25 reach gold steps: 11
episode : 26 reach gold steps: 9
episode : 27 reach gold steps: 11
episode : 28 reach gold steps: 9
episode : 29 reach gold steps: 11
episode : 30 reach gold steps: 9
episode : 31 reach gold steps: 10
episode : 32 reach gold steps: 9
episode : 33 reach gold steps: 12
episode : 34 reach gold steps: 9
episode : 35 reach gold steps: 9
```

```
episode : 36 reach gold steps: 9
episode : 37 reach gold steps: 10
episode : 38 reach gold steps: 9
episode : 39 reach gold steps: 9
episode : 40 reach gold steps: 12
episode : 41 reach gold steps: 12
episode : 42 reach gold steps: 10
episode : 43 reach gold steps: 9
episode : 44 reach gold steps: 9
episode : 45 reach gold steps: 9
episode : 46 reach gold steps: 9
episode : 47 reach gold steps: 9
episode : 48 reach gold steps: 9
episode : 49 reach gold steps: 9
episode : 50 reach gold steps: 11
episode : 51 reach gold steps: 9
episode : 52 reach gold steps: 9
episode : 53 reach gold steps: 10
episode : 54 reach gold steps: 10
episode : 55 reach gold steps: 9
episode : 56 reach gold steps: 9
episode : 57 reach gold steps: 12
episode : 58 reach gold steps: 9
episode : 59 reach gold steps: 12
episode : 60 reach gold steps: 11
episode : 61 reach gold steps: 10
episode : 62 reach gold steps: 9
episode : 63 reach gold steps: 10
episode : 64 reach gold steps: 9
episode : 65 reach gold steps: 11
episode : 66 reach gold steps: 9
episode : 67 reach gold steps: 9
episode : 68 reach gold steps: 14
episode : 69 reach gold steps: 9
episode : 70 reach gold steps: 10
episode : 71 reach gold steps: 9
episode : 72 reach gold steps: 9
episode : 73 reach gold steps: 12
episode : 74 reach gold steps: 9
episode : 75 reach gold steps: 10
episode : 76 reach gold steps: 12
episode : 77 reach gold steps: 10
episode : 78 reach gold steps: 10
episode : 79 reach gold steps: 9
episode : 80 reach gold steps: 9
episode : 81 reach gold steps: 9
episode : 82 reach gold steps: 9
episode : 83 reach gold steps: 10
episode : 84 reach gold steps: 10
```

```
episode : 85 reach gold steps: 11
episode : 86 reach gold steps: 10
episode : 87 reach gold steps: 9
episode : 88 reach gold steps: 11
episode : 89 reach gold steps: 9
episode : 90 reach gold steps: 14
episode : 91 reach gold steps: 11
episode : 92 reach gold steps: 12
episode : 93 reach gold steps: 9
episode : 94 reach gold steps: 10
episode : 95 reach gold steps: 10
episode : 96 reach gold steps: 9
episode : 97 reach gold steps: 9
episode : 98 reach gold steps: 9
episode : 99 reach gold steps: 9
episode : 100 reach gold steps: 9
episode : 101 reach gold steps: 11
episode : 102 reach gold steps: 9
episode : 103 reach gold steps: 10
episode : 104 reach gold steps: 11
episode : 105 reach gold steps: 13
episode : 106 reach gold steps: 11
episode : 107 reach gold steps: 10
episode : 108 reach gold steps: 9
episode : 109 reach gold steps: 13
episode : 110 reach gold steps: 9
episode : 111 reach gold steps: 9
episode : 112 reach gold steps: 11
episode : 113 reach gold steps: 10
episode : 114 reach gold steps: 9
episode : 115 reach gold steps: 9
episode : 116 reach gold steps: 10
episode : 117 reach gold steps: 11
episode : 118 reach gold steps: 9
episode : 119 reach gold steps: 9
episode : 120 reach gold steps: 9
episode : 121 reach gold steps: 11
episode : 122 reach gold steps: 11
episode : 123 reach gold steps: 11
episode : 124 reach gold steps: 11
episode : 125 reach gold steps: 9
episode : 126 reach gold steps: 15
episode : 127 reach gold steps: 9
episode : 128 reach gold steps: 9
episode : 129 reach gold steps: 10
episode : 130 reach gold steps: 11
episode : 131 reach gold steps: 10
episode : 132 reach gold steps: 9
episode : 133 reach gold steps: 9
```

```
episode : 134 reach gold steps: 9
episode : 135 reach gold steps: 10
episode : 136 reach gold steps: 11
episode : 137 reach gold steps: 10
episode : 138 reach gold steps: 10
episode : 139 reach gold steps: 9
episode : 140 reach gold steps: 9
episode : 141 reach gold steps: 9
episode : 142 reach gold steps: 11
episode : 143 reach gold steps: 10
episode : 144 reach gold steps: 9
episode : 145 reach gold steps: 9
episode : 146 reach gold steps: 11
episode : 147 reach gold steps: 9
episode : 148 reach gold steps: 10
episode : 149 reach gold steps: 11
episode : 150 reach gold steps: 11
episode : 151 reach gold steps: 10
episode : 152 reach gold steps: 9
episode : 153 reach gold steps: 10
episode : 154 reach gold steps: 9
episode : 155 reach gold steps: 9
episode : 156 reach gold steps: 10
episode : 157 reach gold steps: 9
episode : 158 reach gold steps: 10
episode : 159 reach gold steps: 9
episode : 160 reach gold steps: 11
episode : 161 reach gold steps: 9
episode : 162 reach gold steps: 11
episode : 163 reach gold steps: 11
episode : 164 reach gold steps: 11
episode : 165 reach gold steps: 11
episode : 166 reach gold steps: 10
episode : 167 reach gold steps: 10
episode : 168 reach gold steps: 9
episode : 169 reach gold steps: 9
episode : 170 reach gold steps: 9
episode : 171 reach gold steps: 9
episode : 172 reach gold steps: 9
episode : 173 reach gold steps: 9
episode : 174 reach gold steps: 9
episode : 175 reach gold steps: 11
episode : 176 reach gold steps: 9
episode : 177 reach gold steps: 10
episode : 178 reach gold steps: 9
episode : 179 reach gold steps: 9
episode : 180 reach gold steps: 9
episode : 181 reach gold steps: 11
episode : 182 reach gold steps: 11
```

```
episode : 183 reach gold steps: 11
episode : 184 reach gold steps: 9
episode : 185 reach gold steps: 13
episode : 186 reach gold steps: 11
episode : 187 reach gold steps: 9
episode : 188 reach gold steps: 11
episode : 189 reach gold steps: 9
episode : 190 reach gold steps: 9
episode : 191 reach gold steps: 10
episode : 192 reach gold steps: 12
episode : 193 reach gold steps: 9
episode : 194 reach gold steps: 12
episode : 195 reach gold steps: 10
episode : 196 reach gold steps: 10
episode : 197 reach gold steps: 9
episode : 198 reach gold steps: 11
episode : 199 reach gold steps: 11
the final path is:
(7, 3)
(7, 4)
(7, 5)
(7, 6)
(6, 6)
(5, 6)
(4, 6)
(4, 5)
(4, 4)
(5, 4)
```

Question 2 (MDP)

Q2_MDP.py

```
"""
a Markov System with action
FangZhou 2018124048
"""
STATE_TYPE = ['S1', 'S2', 'S3', 'S4', 'S5']     # 状态的类型
REWORD = {
    "S1":  1.0,
    "S2": -1.0,
    "S3":  0.0,
    "S4":  3.0,
```

```
    "S5":  1.0
}
ACTION = ["A", "B", "C", "D"]   # 事件名
PROBABILITY = {                 # 转移概率
    "S1": {
        "A": {
            "S1": 0.2,
            "S2": 0.8,
            "S3": 0,
            "S4": 0,
            "S5": 0
        },
        "B": {
            "S1": 0,
            "S2": 0,
            "S3": 1,
            "S4": 0,
            "S5": 0,
        },
        "C": {
            "S1": 1,
            "S2": 0,
            "S3": 0,
            "S4": 0,
            "S5": 0,
        },
        "D": {
            "S1": 0.8,
            "S2": 0,
            "S3": 0.2,
            "S4": 0,
            "S5": 0,
        }
    },
    "S2": {
        "A": {
            "S1": 0.7,
            "S2": 0,
            "S3": 0,
            "S4": 0.3,
            "S5": 0
        },
        "B": {
            "S1": 0,
            "S2": 0.1,
            "S3": 0,
            "S4": 0.9,
            "S5": 0,
        },
```

```
        "C": {
            "S1": 0,
            "S2": 0,
            "S3": 0.8,
            "S4": 0.2,
            "S5": 0,
        },
        "D": {
            "S1": 0.8,
            "S2": 0.2,
            "S3": 0,
            "S4": 0,
            "S5": 0,
        }
    },
    "S3": {
        "A": {
            "S1": 0,
            "S2": 0.5,
            "S3": 0.5,
            "S4": 0,
            "S5": 0
        },
        "B": {
            "S1": 0,
            "S2": 0.5,
            "S3": 0.5,
            "S4": 0,
            "S5": 0,
        },
        "C": {
            "S1": 0,
            "S2": 0,
            "S3": 1,
            "S4": 0,
            "S5": 0,
        },
        "D": {
            "S1": 0.5,
            "S2": 0,
            "S3": 0.5,
            "S4": 0,
            "S5": 0,
        }
    },
    "S4": {
        "A": {
            "S1": 0,
            "S2": 0,
```

```
            "S3": 0,
            "S4": 1,
            "S5": 0
        },
        "B": {
            "S1": 0,
            "S2": 0.3,
            "S3": 0,
            "S4": 0.7,
            "S5": 0,
        },
        "C": {
            "S1": 0,
            "S2": 0,
            "S3": 0,
            "S4": 0,
            "S5": 1,
        },
        "D": {
            "S1": 0,
            "S2": 0,
            "S3": 0,
            "S4": 0.5,
            "S5": 0.5,
        }
    },
    "S5": {
        "A": {
            "S1": 0,
            "S2": 0,
            "S3": 0.3,
            "S4": 0,
            "S5": 0.7
        },
        "B": {
            "S1": 0,
            "S2": 0,
            "S3": 0,
            "S4": 0.6,
            "S5": 0.4,
        },
        "C": {
            "S1": 0,
            "S2": 0,
            "S3": 0,
            "S4": 0.9,
            "S5": 0.1,
        },
        "D": {
```

```
            "S1": 0.8,
            "S2": 0,
            "S3": 0,
            "S4": 0,
            "S5": 0.2,
        }
    }
}
EPSILON = 0.0001    # 差值
GAMMA = 0.9         # 衰减

def markov():
    """
    :return: 一个list形式的J向量 e.g
    [
        {
            's1' : xxx,
            's2' : xxx,
            ...
        },
        {
            's1' : xxx,
            's2' : xxx,
            ...
        }
        ...
    ]
    """
    J = list()          # answer
    J.append(REWORD)    # 添加初始状态
    gap = 1             # 初始化n和n+1年之间的J的差值,gap 应该为几个状态间的最大差
值
    while gap > EPSILON:
        temp_map = dict()
        last_J = J[-1:][0]     # 上一个J状态reword向量, 用于更新此时的J状态
        for each_state in STATE_TYPE:
            # 按state更新J
            max_j_with_action = -10000
            for each_action in ACTION:
                tmp_j_with_action = 0
                for sta, pro in PROBABILITY[each_state]
[each_action].items():
                    tmp_j_with_action += last_J[sta] * pro
                if tmp_j_with_action > max_j_with_action:
                    max_j_with_action = tmp_j_with_action
            update_num = max_j_with_action * GAMMA + REWORD[each_state]
            if abs(update_num - last_J[each_state]) > gap:
                gap = abs(update_num - last_J[each_state])   # 更新gap
            temp_map[each_state] = update_num
```

```
        gap = get_gap(temp_map, last_J)
        J.append(temp_map)
    return J


def get_gap(next_dict, last_dict):
    """
    获取next_dict和last_dict之间的最大gap
    :param next_dict:
    :param last_dict:
    :return:
    """
    gap_reword = list()
    for state, reword in next_dict.items():
        gap_reword.append(abs(last_dict[state] - reword))
    return max(gap_reword)


if __name__ == "__main__":
    ans_J = markov()
    for item in ans_J:
        print(item)
```

```
"""
output
J vector in epsilon = 0.0001
"""
{'S1': 1.0, 'S2': -1.0, 'S3': 0.0, 'S4': 3.0, 'S5': 1.0}
{'S1': 1.9, 'S2': 1.3400000000000003, 'S3': 0.45, 'S4': 5.7, 'S5':
3.5200000000000005}
{'S1': 2.71, 'S2': 3.7376000000000005, 'S3': 1.0575, 'S4':
8.129999999999999, 'S5': 5.933800000000001}
{'S1': 4.178872, 'S2': 5.921683999999999, 'S3': 2.157795, 'S4': 10.317,
'S5': 8.119342}
{'S1': 6.01580944, 'S2': 7.889721560000002, 'S3': 3.63576555, 'S4':
12.285300000000001, 'S5': 10.08751078}
{'S1': 7.763445222400002, 'S2': 9.661167940400002, 'S3': 5.186469199500001,
'S4': 14.056770000000002, 'S5': 11.858968970200001}
{'S1': 9.353461057120002, 'S2': 11.255488814636001, 'S3':
6.681436712955001, 'S4': 15.651093000000001, 'S5': 13.453290907318001}
{'S1': 10.78757493681952, 'S2': 12.690379323317242, 'S3': 8.07161648741595,
'S4': 17.0859837, 'S5': 14.888181511658622}
{'S1': 12.078836601415928, 'S2': 13.981780936098554, 'S3':
9.342898114829936, 'S4': 18.377385330000003, 'S5': 16.179583133049277}
```

```
{'S1': 13.241072862245828, 'S2': 15.144042401548877, 'S3':
10.49610557291782, 'S4': 19.539646797000003, 'S5': 17.341844599274438}
{'S1': 14.287103644319442, 'S2': 16.190077721709404, 'S3':
11.538066588510013, 'S4': 20.585682117300003, 'S5': 18.3878799195047}
{'S1': 15.228534615608272, 'S2': 17.13150950996685, 'S3':
12.477664939598737, 'S4': 21.527113905570005, 'S5': 19.329311707768426}
{'S1': 16.07582307798562, 'S2': 17.978798119408722, 'S3':
13.324128502304514, 'S4': 22.374402515013006, 'S5': 20.176600317210866}
{'S1': 16.838382800011694, 'S2': 18.741357867907322, 'S3':
14.086316979770956, 'S4': 23.136962263511705, 'S5': 20.939160065709512}
{'S1': 17.524686568895376, 'S2': 19.427661641556142, 'S3':
14.772453681455227, 'S4': 23.823266037160536, 'S5': 21.62546383935834}
{'S1': 18.14235996432159, 'S2': 20.04533503784009, 'S3':
15.390051895355116, 'S4': 24.440939433444484, 'S5': 22.243137235642287}
{'S1': 18.69826602082275, 'S2': 20.60124109449564, 'S3':
15.945924119937843, 'S4': 24.996845490100036, 'S5': 22.79904329229784}
{'S1': 19.19858147178496, 'S2': 21.10155654548564, 'S3':
16.446224346495068, 'S4': 25.497160941090034, 'S5': 23.299358743287836}
{'S1': 19.648865377670955, 'S2': 21.551840451376638, 'S3':
16.89650140139132, 'S4': 25.947444846981032, 'S5': 23.749642649178835}
{'S1': 20.05412089297195, 'S2': 21.957095966678533, 'S3':
17.30175383374558, 'S4': 26.352700362282928, 'S5': 24.15489816448073}
{'S1': 20.418850856743497, 'S2': 22.32182593045024, 'S3':
17.666482410190852, 'S4': 26.717430326054636, 'S5': 24.51962812825244}
{'S1': 20.747107824138006, 'S2': 22.65008289784478, 'S3':
17.994738753288495, 'S4': 27.045687293449173, 'S5': 24.847885095646976}
{'S1': 21.042539094793085, 'S2': 22.945514168499862, 'S3':
18.290169743009972, 'S4': 27.341118564104256, 'S5': 25.14331636630206}
{'S1': 21.30842723838266, 'S2': 23.211402312089437, 'S3':
18.556057760179428, 'S4': 27.60700670769383, 'S5': 25.409204509891634}
{'S1': 21.547726567613275, 'S2': 23.450701641320055, 'S3':
18.79535703252099, 'S4': 27.84630603692445, 'S5': 25.648503839122252}
{'S1': 21.76309596392083, 'S2': 23.666071037627606, 'S3':
19.01072640322847, 'S4': 28.061675433232004, 'S5': 25.863873235429807}
{'S1': 21.956928420597627, 'S2': 23.85990349430441, 'S3':
19.204558848385236, 'S4': 28.255507889908806, 'S5': 26.05770569210661}
{'S1': 22.13137763160675, 'S2': 24.034352705313527, 'S3':
19.379008054210345, 'S4': 28.429957100917925, 'S5': 26.232154903115728}
{'S1': 22.288381921514958, 'S2': 24.19135699522174, 'S3':
19.536012341785742, 'S4': 28.586961390826133, 'S5': 26.389159193023936}
{'S1': 22.429685782432347, 'S2': 24.33266085613912 5, 'S3':
19.677316201653365, 'S4': 28.72826525174352, 'S5': 26.53046305394132}
{'S1': 22.556859257257994, 'S2': 24.459834330964775, 'S3':
19.80448967600662, 'S4': 28.85543872656917, 'S5': 26.65763652876697}
{'S1': 22.67131538460108, 'S2': 24.574290458307857, 'S3':
19.918945803137127, 'S4': 28.96989485391225, 'S5': 26.772092656110054}
{'S1': 22.774325899209853, 'S2': 24.677300972916633, 'S3':
20.021956317650243, 'S4': 29.072905368521027, 'S5': 26.87510317071883}
```

```
{'S1': 22.867035362357754, 'S2': 24.77001043606453, 'S3':
20.114665780755097, 'S4': 29.165614831668925, 'S5': 26.96781263386673}
{'S1': 22.95047387919086, 'S2': 24.85344895289764, 'S3':
20.198104297568833, 'S4': 29.249053348502034, 'S5': 27.051251150699837}
{'S1': 23.025568544340658, 'S2': 24.92854361804744, 'S3':
20.273198962709913, 'S4': 29.324148013651833, 'S5': 27.126345815849636}
{'S1': 23.093153742975474, 'S2': 24.99612881668225, 'S3':
20.34078416134081, 'S4': 29.39173321228665, 'S5': 27.193931014484452}
{'S1': 23.15398042174681, 'S2': 25.056955495453586, 'S3':
20.40161084011038, 'S4': 29.452559891057984, 'S5': 27.254757693255787}
{'S1': 23.20872443264101, 'S2': 25.11169950634779, 'S3':
20.456354851003784, 'S4': 29.507303901952184, 'S5': 27.309501704149987}
{'S1': 23.25799404244579, 'S2': 25.16096911615257, 'S3':
20.505624460808207, 'S4': 29.556573511756966, 'S5': 27.35877131395477}
{'S1': 23.302336691270096, 'S2': 25.205311764976877, 'S3':
20.54996710963235, 'S4': 29.60091616058127, 'S5': 27.403113962779074}
{'S1': 23.34224507521197, 'S2': 25.24522014891875, 'S3':
20.589875493574155, 'S4': 29.640824544523145, 'S5': 27.443022346720948}
{'S1': 23.37816262075966, 'S2': 25.281137694466437, 'S3':
20.625793039121806, 'S4': 29.67674209007083, 'S5': 27.478939892268635}
{'S1': 23.410488411752578, 'S2': 25.313463485459355, 'S3':
20.65811883011471, 'S4': 29.70906788106375, 'S5': 27.511265683261552}
{'S1': 23.439581623646202, 'S2': 25.34255669735298, 'S3':
20.68721204200833, 'S4': 29.738161092957373, 'S5': 27.540358895155176}
{'S1': 23.465765514350466, 'S2': 25.368740588057243, 'S3':
20.71339593271259, 'S4': 29.764344983661637, 'S5': 27.56654278585944}
{'S1': 23.489331015984302, 'S2': 25.39230608969108, 'S3':
20.736961434346426, 'S4': 29.787910485295473, 'S5': 27.590108287493276}
{'S1': 23.51053996745475, 'S2': 25.413515041161528, 'S3':
20.758170385816875, 'S4': 29.809119436765926, 'S5': 27.61131723896373}
{'S1': 23.52962802377816, 'S2': 25.43260309748494, 'S3': 20.77725844214028,
'S4': 29.828207493089334, 'S5': 27.630405295287137}
{'S1': 23.546807274469227, 'S2': 25.449782348176008, 'S3':
20.79443769283135, 'S4': 29.845386743780402, 'S5': 27.647584545978205}
{'S1': 23.562268600091187, 'S2': 25.465243673797968, 'S3':
20.80989901845331, 'S4': 29.860848069402362, 'S5': 27.663045871600165}
{'S1': 23.576183793150953, 'S2': 25.47915886685773, 'S3':
20.823814211513078, 'S4': 29.874763262462125, 'S5': 27.676961064659928}
{'S1': 23.58870746690474, 'S2': 25.491682540611517, 'S3':
20.836337885266865, 'S4': 29.88728693621591, 'S5': 27.689484738413714}
{'S1': 23.599978773283148, 'S2': 25.502953846989925, 'S3':
20.847609191645272, 'S4': 29.89855824259432, 'S5': 27.700756044792122}
{'S1': 23.610122949023715, 'S2': 25.513098022730492, 'S3':
20.85775336738584, 'S4': 29.908702418334887, 'S5': 27.71090022053269}
{'S1': 23.619252707190228, 'S2': 25.522227780897005, 'S3':
20.86688312555235, 'S4': 29.9178321765014, 'S5': 27.720029978699202}
{'S1': 23.627469489540086, 'S2': 25.530444563246867, 'S3':
20.87509990790221, 'S4': 29.92604895885126, 'S5': 27.728246761049064}
```

{'S1': 23.63486459365496, 'S2': 25.53783966736174, 'S3': 20.882495012017085, 'S4': 29.933444062966135, 'S5': 27.735641865163938}
{'S1': 23.64152018735835, 'S2': 25.544495261065126, 'S3': 20.889150605720474, 'S4': 29.94009965666952, 'S5': 27.742297458867323}
{'S1': 23.647510221691398, 'S2': 25.550485295398175, 'S3': 20.89514064005352, 'S4': 29.94608969100257, 'S5': 27.748287493200372}
{'S1': 23.65290125259114, 'S2': 25.55587632629792, 'S3': 20.900531670953264, 'S4': 29.951480721902314, 'S5': 27.753678524100117}
{'S1': 23.65775318040091, 'S2': 25.56072825410769, 'S3': 20.905383598763034, 'S4': 29.956332649712085, 'S5': 27.758530451909888}
{'S1': 23.662119915429706, 'S2': 25.565094989136483, 'S3': 20.909750333791827, 'S4': 29.960699384740877, 'S5': 27.76289718693868}
{'S1': 23.666049976955616, 'S2': 25.569025050662397, 'S3': 20.91368039531774, 'S4': 29.96462944626679, 'S5': 27.766827248464594}
{'S1': 23.66958703232894, 'S2': 25.57256210603572, 'S3': 20.917217450691062, 'S4': 29.968166501640113, 'S5': 27.770364303837916}
{'S1': 23.67277038216493, 'S2': 25.575745455871708, 'S3': 20.92040080052705, 'S4': 29.971349851476102, 'S5': 27.773547653673905}
{'S1': 23.675635397017317, 'S2': 25.578610470724094, 'S3': 20.92326581537944, 'S4': 29.97421486632849, 'S5': 27.776412668526294}
{'S1': 23.678213910384468, 'S2': 25.58118898409125, 'S3': 20.925844328746592, 'S4': 29.976793379695643, 'S5': 27.778991181893446}
{'S1': 23.680534572414906, 'S2': 25.583509646121684, 'S3': 20.928164990777027, 'S4': 29.979114041726078, 'S5': 27.78131184392388}
{'S1': 23.682623168242298, 'S2': 25.585598241949075, 'S3': 20.93025358660442, 'S4': 29.98120263755347, 'S5': 27.783400439751272}
{'S1': 23.68450290448695, 'S2': 25.58747797819373, 'S3': 20.932133322849072, 'S4': 29.983082373798123, 'S5': 27.785280175995926}
{'S1': 23.686194667107138, 'S2': 25.589169740813915, 'S3': 20.933825085469262, 'S4': 29.98477413641831, 'S5': 27.786971938616112}
{'S1': 23.687717253465305, 'S2': 25.590692327172086, 'S3': 20.935347671827433, 'S4': 29.98629672277648, 'S5': 27.788494524974283}
{'S1': 23.689087581187657, 'S2': 25.592062654894438, 'S3': 20.936717999549785, 'S4': 29.987667050498832, 'S5': 27.789864852696635}
{'S1': 23.690320876137775, 'S2': 25.593295949844556, 'S3': 20.9379512944999, 'S4': 29.98890034544895, 'S5': 27.791098147646753}
{'S1': 23.691430841592883, 'S2': 25.59440591529966, 'S3': 20.939061259955007, 'S4': 29.990010310904054, 'S5': 27.792208113101857}
{'S1': 23.692429810502478, 'S2': 25.595404884209255, 'S3': 20.940060228864603, 'S4': 29.99100927981365, 'S5': 27.793207082011453}
{'S1': 23.69332888252111, 'S2': 25.596303956227892, 'S3': 20.940959300883236, 'S4': 29.991908351832286, 'S5': 27.79410615403009}
{'S1': 23.694138047337887, 'S2': 25.597113121044664, 'S3': 20.941768465700008, 'S4': 29.99271751664906, 'S5': 27.79491531884686}
{'S1': 23.69486629567298, 'S2': 25.597841369379758, 'S3': 20.9424967140351, 'S4': 29.993445764984152, 'S5': 27.795643567181955}
{'S1': 23.695521719174565, 'S2': 25.598496792881342, 'S3': 20.943152137536686, 'S4': 29.994101188485736, 'S5': 27.79629899068354}

```
{'S1': 23.69611160032599, 'S2': 25.59908667403277, 'S3':
20.943742018688113, 'S4': 29.994691069637163, 'S5': 27.796888871834966}
{'S1': 23.696642493362276, 'S2': 25.599617567069053, 'S3':
20.944272911724397, 'S4': 29.995221962673448, 'S5': 27.79741976487125}
{'S1': 23.69712029709493, 'S2': 25.60009537080171, 'S3':
20.944750715457054, 'S4': 29.995699766406105, 'S5': 27.797897568603908}
{'S1': 23.697550320454322, 'S2': 25.6005253941611, 'S3':
20.945180738816443, 'S4': 29.996129789765494, 'S5': 27.798327591963297}
{'S1': 23.697937341477772, 'S2': 25.60091241518455, 'S3':
20.945567759839896, 'S4': 29.996516810788943, 'S5': 27.798714612986746}
{'S1': 23.698285660398874, 'S2': 25.60126073410565, 'S3':
20.945916078760998, 'S4': 29.99686512971005, 'S5': 27.79906293190785}
{'S1': 23.698599147427867, 'S2': 25.60157422113465, 'S3':
20.946229565789995, 'S4': 29.997178616739046, 'S5': 27.79937641893685}
{'S1': 23.698881285753966, 'S2': 25.601856359460747, 'S3':
20.94651170411609, 'S4': 29.99746075506514, 'S5': 27.799658557262944}
{'S1': 23.699135210247455, 'S2': 25.602110283954232, 'S3':
20.946765628609576, 'S4': 29.997714679558626, 'S5': 27.79991248175643}
{'S1': 23.69936374229159, 'S2': 25.60233881599837, 'S3':
20.946994160653713, 'S4': 29.997943211602763, 'S5': 27.800141013800566}
{'S1': 23.69956942113131, 'S2': 25.60254449483809, 'S3':
20.947199839493436, 'S4': 29.998148890442486, 'S5': 27.80034669264029}
{'S1': 23.69975453208706, 'S2': 25.602729605793844, 'S3':
20.947384950449184, 'S4': 29.998334001398238, 'S5': 27.80053180359604}
{'S1': 23.69992113194724, 'S2': 25.602896205654016, 'S3':
20.947551550309363, 'S4': 29.998500601258414, 'S5': 27.800698403456217}
{'S1': 23.700071071821398, 'S2': 25.60304614552818, 'S3':
20.94770149018352, 'S4': 29.998650541132573, 'S5': 27.800848343330376}
{'S1': 23.70020601770814, 'S2': 25.603181091414918, 'S3':
20.947836436070265, 'S4': 29.998785487019315, 'S5': 27.80098328921712}
{'S1': 23.700327469006208, 'S2': 25.60330254271299, 'S3':
20.947957887368336, 'S4': 29.998906938317383, 'S5': 27.801104740515186}
{'S1': 23.700436775174474, 'S2': 25.60341184888125, 'S3': 20.9480671935366,
'S4': 29.999016244485645, 'S5': 27.801214046683448}
{'S1': 23.70053515072591, 'S2': 25.603510224432686, 'S3':
20.948165569088033, 'S4': 29.99911462003708, 'S5': 27.801312422234883}
```