2 Q2

(a) By using scipy.optimize.linprog in Python, we finally get two minimax profiles (NEs) from given matrix, namely, ((2/15, 17/30, 0, 3/10, 0), (1/3, 0, 1/3, 0,1/3)) and ((0, 1/2, 0, 5/18, 2/9), (1/3, 0, 1/3, 0, 1/3)), with the same minimax value 14/3, which is shown in Table 3.

Interestingly, by using different linear programming methods, such as 'simplex' and 'revised simplex', and the results reuturned from code are shown in Figure 1 and Figure 2. We found that player 1 has different optimal solutions, but with the same payoff value. After consulting the documentation, we found that 'scipy.optimize.linprog' can only return one result, which is the same as Matlab 'linprog'. If the inequality constraints in matrix A given in a different order, the algorithm will return different results if there are multiple optimal solutions.

Below is the code part of using linprog in Python.

```
from scipy.optimize import linprog
import numpy as np

print('Prime ---------------------------------')
A = [[3,3,9,6,2],
     [7,8,4,5,3],
     [1,2,5,6,4],
     [1,4,4,5,9],
     [4,7,7,8,3]]

A = np.array(A)
A_ub = np.hstack((0-A.T, np.ones((5, 1))))
A_eq = np.array([[1, 1, 1, 1, 1, 0]])
b_ub = np.zeros(5)
b_eq = [1]
c = [0,0,0,0,0,-1]
x_bounds = (0, 1)
x6_bounds = (None, None)
res = linprog(c, A_ub=A_ub, b_ub=b_ub, A_eq=A_eq, b_eq=b_eq, \
              bounds=[x_bounds, x_bounds, x_bounds, x_bounds, x_bounds, x6_bounds], \
              method='simplex')
print(res)

print('Another solution------------------------------')

res = linprog(c, A_ub=A_ub, b_ub=b_ub, A_eq=A_eq, b_eq=b_eq, \
              bounds=[x_bounds, x_bounds, x_bounds, x_bounds, x_bounds, x6_bounds], \
              method='revised simplex')
print(res)

print('Dual ---------------------------------')
A = np.array(A)
A_ub = np.hstack((A, -np.ones((5, 1))))
A_eq = np.array([[1, 1, 1, 1, 1, 0]])
b_ub = np.zeros(5)
b_eq = [1]
c = [0,0,0,0,0,1]
x_bounds = (0, 1)
x6_bounds = (None, None)
res = linprog(c, A_ub=A_ub, b_ub=b_ub, A_eq=A_eq, b_eq=b_eq, \
              bounds=[x_bounds, x_bounds,x_bounds,x_bounds,x_bounds, x6_bounds], \
              method='revised simplex')
print(res)
```

```
Prime ----------------------
     con: array([1.11022302e-16])
     fun: -4.666666666666666
 message: 'Optimization terminated successfully.'
     nit: 6
   slack: array([0.        , 1.46666667, 0.        , 0.46666667, 0.        ])
  status: 0
 success: True
       x: array([0.13333333, 0.56666667, 0.        , 0.3       , 0.        ,
       4.66666667])
Dual ----------------------
     con: array([-2.22044605e-16])
     fun: 4.666666666666667
 message: 'Optimization terminated successfully.'
     nit: 7
   slack: array([ 0.00000000e+00,  4.44089210e-16,  1.33333333e+00, -6.66133815e-16,
        0.00000000e+00])
  status: 0
 success: True
       x: array([0.33333333, 0.        , 0.33333333, 0.        , 0.33333333,
       4.66666667])
```

Figure 1: scipy.optimize.linprog with method = 'revised simplex'.

```
Prime ----------------------
     con: array([-2.22044605e-16])
     fun: -4.666666666666668
 message: 'Optimization terminated successfully.'
     nit: 13
   slack: array([ 8.8817842e-16,  2.0000000e+00,  0.0000000e+00,  1.0000000e+00,
       -8.8817842e-16])
  status: 0
 success: True
       x: array([0.        , 0.5       , 0.        , 0.27777778, 0.22222222,
       4.66666667])
Dual ----------------------
     con: array([-2.22044605e-16])
     fun: 4.666666666666667
 message: 'Optimization terminated successfully.'
     nit: 7
   slack: array([ 0.00000000e+00,  4.44089210e-16,  1.33333333e+00, -6.66133815e-16,
        0.00000000e+00])
  status: 0
 success: True
       x: array([0.33333333, 0.        , 0.33333333, 0.        , 0.33333333,
       4.66666667])
```

Figure 2: scipy.optimize.linprog with method = 'revised simplex'.

player 1's strategy	player 2's strategy	player 1's payoff	player 2's payoff	minimax value
(2/15, 17/30, 0, 3/10, 0)	(1/3, 0, 1/3, 0,1/3)	14/3	-14/3	14/3
(0, 1/2, 0, 5/18, 2/9)	(1/3, 0, 1/3, 0, 1/3)	14/3	-14/3	14/3

Table 3: The NEs in game G.

(b) Let $w = (y^*, x^*, z)$ be a maxminimizer strategy for player 2 in the game G.Note that the value of any symmetric 2-player zero-sum game must be equal to zero. This implies, by the minimax theorem, that $Bw <= 0$, which implies,

$$Ax^* - bz <= 0 \tag{1}$$

$$-A^T y^* + cz <= 0 \tag{2}$$

$$b^T y^* - c^T x* <= 0 \tag{3}$$

Assuming z = 0, we get $Ax^* <= 0$ and $A^T y^* >= 0$. Then if $y^*! = 0$, $(y^*)^T (Ax' - b) < 0$, because $y^* > 0$ and there exists x' that holds $(Ax' - b) < 0$. Likewise, if $x^*! = 0$, $(x^*)^T (Ay' - c) > 0$, because $x^* > 0$ and there exists y' that holds $(Ay' - c) > 0$. By inequality transformation and weak duality, we can get

$$(y^*)^T Ax' < (y^*)^T b <= (x^*)^T c < (x^*)^T Ay' \tag{4}$$

However, because $(y^*)^T A >= 0$, $(x^*)^T A <= 0$, $x' >= 0$, and $y' >= 0$, we can get $(x^*)^T Ay' <= 0 <= (y^*)^T Ax'$, which violates the the inequality (4). Thus, we can prove that if z = 0, there will be contradiction with our assumptions.

Because $0 <= z <= 1$, we then discuss the availability of remaining valid range $z > 0$. For $z > 0$ we can times z on both size of inequality $Ax' < b$ and $A^T y' > c$ and make $x'' = x'z$, $x^{**} = x^* z$, $y'' = y'z$, and $y^{**} = y^* z$, then we get

$$Ax'' < bz \tag{5}$$

$$A^T y'' > cz \tag{6}$$

With the weak duality, we can get

$$(y^{**})^T Ax' < (y^{**})^T b <= (x^{**})^T c < (x^{**})^T Ay' \tag{7}$$

And finally we can get the below inequalities, the equality holds if and only if (x^{**}, y^{**}) is the minimax profile of this game.

$$Ax^{**} - bz <= 0 \tag{8}$$

$$-A^T y^{**} + cz <= 0 \tag{9}$$

$$b^T y^{**} - c^T x^{**} <= 0 \tag{10}$$

In conlusion, we proved that when z = 0, there are conflicts with known assumptions, while $z > 0$, we can get $Bw <= 0$ from conditions given and weak duality. Thus, for the game G, every minmaximizer strategy $w = (y^*, x^*, z)$ for player 1 (and hence also every maxminimizer strategy for player 2, since the game is symmetric) has the property that $z > 0$.