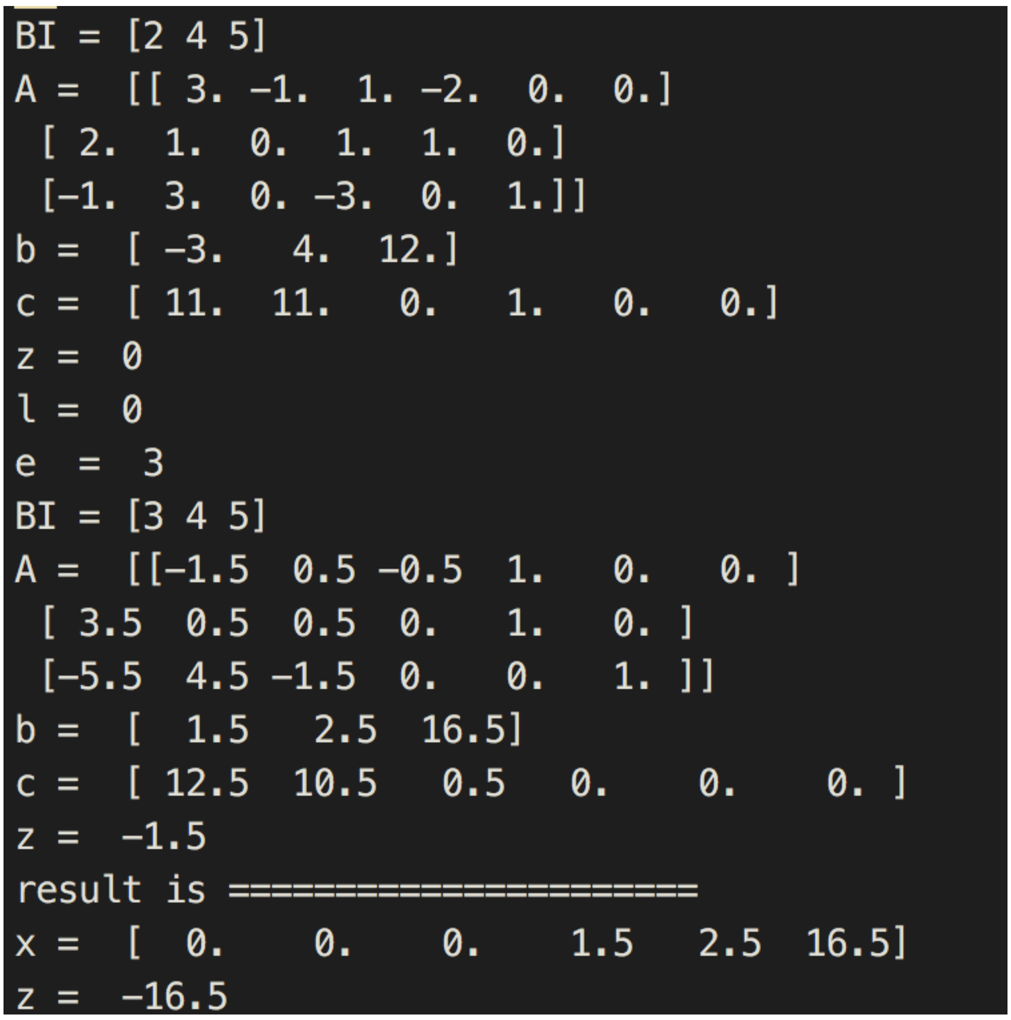
第 7 题代码

```
#!/usr/bin/env python3
#-*- coding: utf-8 -*-

'''
@author: 金晔俊
@data: 2017-12-21
'''

import numpy as np


def calculate_x(BI, A, b, c):
    '''
    assign non-basic variables with 0, and assign basic varaibles
    with corresponding bi;
    '''

    x = np.zeros(A.shape[1])
    for j in BI:
        for i in range(A.shape[0]):
            if  A[i, j] == 1:
                x[j] = b[i]
    return x # return the corresponding vector


def pivot(BI, A, b, c, z, e, l):

    # scaling the l-th line
    b[l] /= A[l, e]

    for j in range(A.shape[1]):
        A[l, j] /= A[l, e]

    # Gauss elimination
    for i in range(A.shape[0]):
        if i == l:
            continue
        b[i] -= A[i, e] * b[l]
        for j in range(A.shape[1]):
            A[i, j] -= A[i, e] * A[l, j]

    z -= b[l] * c[e]

    for j in range(A.shape[1]):
        c[j] -= c[e] * A[l, j]

    # BI = BI - {l} U {e}
    for i, v in np.ndenumerate(BI):
        if A[i, e] == 1:
            BI[i] = e
            break

    return BI, A, b, c, z


def dual_simplex(BI, z, A, b, c):

    '''
    Dual simplex starts with a dual feasible basis.
    Here Bi contains the indices of the baisc variables.
    '''

    while True:
        print('BI =', BI)
        print('A = ', A)
        print('b = ', b)
        print('c = ', c)
        print('z = ', z)

        if np.all(b >= 0):
            x = calculate_x(BI, A, b, c)
            return (x, z)

        # Get the smallest b for all b < 0
        l = np.argmin(b)

        print('l = ' , l)
        e = -1
        det_e = np.inf
        # choose an index e that minimizes det_j
        for j in range(A.shape[1]):
            if A[l, j] < 0:
                det_j = - c[j] / A[l, j]
                det_e, e = (det_j, j) if det_j < det_e else (det_e, e)

        if det_e == np.inf:
            print('No feasiable solution')
            return None # Here None means no feasible solution

        print('e  = ' , e)

        BI, A, b, c, z = pivot(BI, A, b, c, z, e, l)


def _main():
    A = np.array([
                [3, -1, 1, -2, 0, 0],
                [2, 1, 0, 1, 1, 0],
                [-1, 3, 0, -3, 0, 1],
              ], dtype=np.float)

    b = np.array([ -3, 4, 12], dtype=np.float)
    z = 0
    c = np.array([ 11, 11, 0, 1, 0, 0], dtype=np.float)

    # find BI
    BI = []
    eye3 = np.eye(3)
    for j in range(A.shape[1]):
        for i in range(eye3.shape[1]):
            if np.all(A[:, j] == eye3[:, i]):
                BI.append(j)
                break
    BI = np.array(BI)

    x, z = dual_simplex(BI, z, A, b, c)
    print("result is ======================")
    print("x = ", x)
    print("z = ", -z - 18)


if __name__ == '__main__':
    _main()
```

运行结果：



```
BI = [2 4 5]
A =  [[ 3. -1.  1. -2.  0.  0.]
 [ 2.  1.  0.  1.  1.  0.]
 [-1.  3.  0. -3.  0.  1.]]
b =  [ -3.   4.  12.]
c =  [ 11.  11.   0.   1.   0.   0.]
z =  0
l =  0
e  =  3
BI = [3 4 5]
A =  [[-1.5  0.5 -0.5  1.   0.   0. ]
 [ 3.5  0.5  0.5  0.   1.   0. ]
 [-5.5  4.5 -1.5  0.   0.   1. ]]
b =  [  1.5   2.5  16.5]
c =  [ 12.5  10.5   0.5   0.    0.    0. ]
z =  -1.5
result is ======================
x =  [  0.    0.    0.    1.5   2.5  16.5]
z =  -16.5
```