PP05: Linear Programming

CST501: Advanced Algorithms Programming Project Write-Up
Posting ID: ___6421-735_.

Pledge:

I Becca Little pledge, on my honor, that the work submitted is my own and that I have neither sought nor provided inappropriate help to any other. I understand that I may not share any part of my solution or design (even if I have a bug and want help!) with any person other than the grader or instructor.
I Becca Little further understand that I must submit this completed write-up with my name inserted into the pledge, and also a .zip file containing my source-code as presented here as two separate attachments to the course website.

Highlighted Source Code:

```
1       # -*- coding: utf-8 -*-


4       import sys # NOQA
5       from mprint import printmat

7       NAME = 'Becca Little'


10      def lu(A):
11              n = len(A)
12              LU = [[A[i][j] for j in range(n)] for i in range(n)]
13              for k in range(0, n):
14                      printmat(LU)
15                      for i in range(k + 1, n):
16                              LU[i][k] = LU[i][k] / LU[k][k]
17                      for i in range(k + 1, n):
18                              for j in range(k + 1, n):
19                                      LU[i][j] = LU[i][j] - (LU[i][k] * LU[k][j])
20              printmat(LU)
21              return LU, 'solved'
```

```
24  def lup(A):
25      n = len(A)
26      LU = [[float(A[i][j]) for j in range(n)] for i in range(n)]
27      pi = range(n)
28      for k in range(0, n):
29          printmat(LU, perm=pi)
30          p = 0
31          for i in range(k, n):
32              if (abs(LU[i][k]) > p):
33                  p = abs(LU[i][k])
34                  kprime = i
35          temp = pi[k]
36          pi[k] = pi[kprime]
37          pi[kprime] = temp
38          for i in range(0, n):
39              temp = LU[k][i]
40              LU[k][i] = LU[kprime][i]
41              LU[kprime][i] = temp
42          for i in range(k + 1, n):
43              LU[i][k] = LU[i][k] / LU[k][k]
44              for j in range(k + 1, n):
45                  LU[i][j] = LU[i][j] - (LU[i][k] * LU[k][j])
46      printmat(LU, perm=pi)
47      return LU, pi, 'solved'


50  def lupsolve(LU, pi, b):
51      n = len(b)
52      x = [float(b[pi[i]]) for i in range(n)]
53      for i in range(1, n):
54          suma = 0
55          for j in range(0, i):
56              suma += LU[i][j] * x[j]
57          x[i] = b[pi[i]] - suma
58      for i in range(n - 1, -1, -1):
59          suma = 0
60          for j in range(i + 1, n):
61              suma += LU[i][j] * x[j]
62          x[i] = (x[i] - suma) / LU[i][i]
63      return x


66  def pivot(A, row, col):
67      m = len(A)
68      n = len(A[0])
```

```
69              Ahat = [[0] * n for i in range(m)]
70              for i in range(0, n):
71                      Ahat[row][i] = A[row][i] / float(A[row][col])
72              for i in range(0, m):
73                      if (i == row):
74                              continue
75                      multiple = A[i][col] / float(Ahat[row][col])
76                      for j in range(0, n):
77                              Ahat[i][j] = A[i][j] - (Ahat[row][j] * multiple)
78              return Ahat
```

I won't include the rest, as it is the scaffolding provided and I didn't change it.

Example Output:

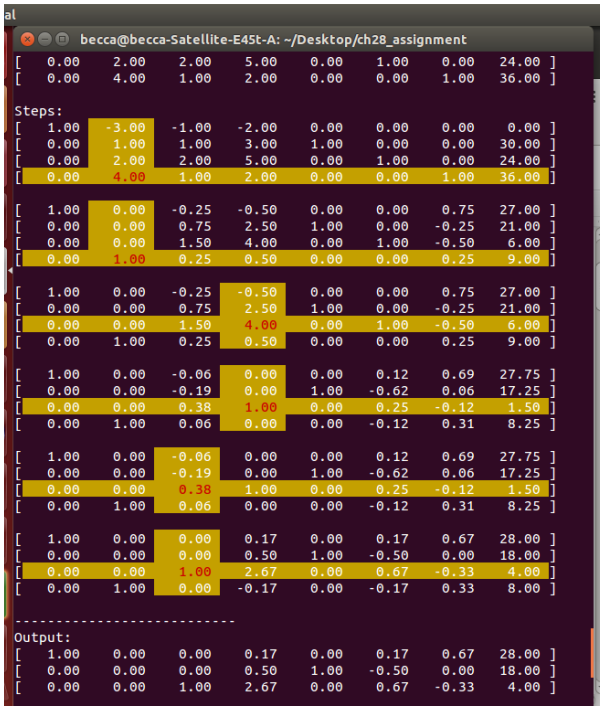
```
becca@becca-Satellite-E45t-A:~/Desktop/ch28_assignment$ flake8 --max-complexity
10 hw_ch28_fixed.py
becca@becca-Satellite-E45t-A:~/Desktop/ch28_assignment$ python hw_ch28_fixed.py
============================
Submitted by  Becca Little
LU:
Input:
[   4.00        -5.00        6.00 ]
[   8.00        -6.00        7.00 ]
[  12.00        -7.00        12.00 ]

Steps:
[   4.00        -5.00        6.00 ]
[   8.00        -6.00        7.00 ]
[  12.00        -7.00        12.00 ]

[   4.00        -5.00        6.00 ]
[   2.00        4.00         -5.00 ]
[   3.00        8.00         -6.00 ]

[   4.00        -5.00        6.00 ]
[   2.00        4.00         -5.00 ]
[   3.00        2.00         4.00 ]

[   4.00         -5.00       6.00 ]
```

```
[   2.00          4.00          -5.00 ]
[   3.00          2.00           4.00 ]

---------------------------
Output:
[   4.00         -5.00           6.00 ]
[   2.00          4.00          -5.00 ]
[   3.00          2.00           4.00 ]

result =  solved
---------------------------
===========================
Submitted by  Becca Little
LUP:
---------------------------
Input:
[   2.00          0.00    2.00    0.60 ]
[   3.00          3.00    4.00   -2.00 ]
[   5.00          5.00    4.00    2.00 ]
[  -1.00         -2.00    3.40   -1.00 ]

---------------------------
Steps:
[  0][   2.00     0.00    2.00    0.60 ]
[  1][   3.00     3.00    4.00   -2.00 ]
[  2][   5.00     5.00    4.00    2.00 ]
[  3][  -1.00    -2.00    3.40   -1.00 ]

[  2][   5.00     5.00    4.00    2.00 ]
[  1][   0.60     0.00    1.60   -3.20 ]
[  0][   0.40    -2.00    0.40   -0.20 ]
[  3][  -0.20    -1.00    4.20   -0.60 ]

[  2][   5.00     5.00    4.00    2.00 ]
[  0][   0.40    -2.00    0.40   -0.20 ]
[  1][   0.60    -0.00    1.60   -3.20 ]
[  3][  -0.20     0.50    4.00   -0.50 ]

[  2][   5.00     5.00    4.00    2.00 ]
```

```
[  0][   0.40   -2.00    0.40   -0.20 ]
[  3][  -0.20    0.50    4.00   -0.50 ]
[  1][   0.60   -0.00    0.40   -3.00 ]

[  2][   5.00    5.00    4.00    2.00 ]
[  0][   0.40   -2.00    0.40   -0.20 ]
[  3][  -0.20    0.50    4.00   -0.50 ]
[  1][   0.60   -0.00    0.40   -3.00 ]

---------------------------
Output:
[  2][   5.00    5.00    4.00    2.00 ]
[  0][   0.40   -2.00    0.40   -0.20 ]
[  3][  -0.20    0.50    4.00   -0.50 ]
[  1][   0.60   -0.00    0.40   -3.00 ]

---------------------------
result =  solved
---------------------------
============================
Submitted by  Becca Little
SIMPLEX:
Input:
[   1.00         -3.00  -1.00  -2.00    0.00    0.00    0.00    0.00 ]
[   0.00          1.00   1.00   3.00    1.00    0.00    0.00   30.00 ]
[   0.00          2.00   2.00   5.00    0.00    1.00    0.00   24.00 ]
[   0.00          4.00   1.00   2.00    0.00    0.00    1.00   36.00 ]

Steps:
[   1.00         -3.00  -1.00  -2.00    0.00    0.00    0.00    0.00 ]
[   0.00          1.00   1.00   3.00    1.00    0.00    0.00   30.00 ]
[   0.00          2.00   2.00   5.00    0.00    1.00    0.00   24.00 ]
[   0.00          4.00   1.00   2.00    0.00    0.00    1.00   36.00 ]

[   1.00          0.00  -0.25  -0.50    0.00    0.00    0.75   27.00 ]
[   0.00          0.00   0.75   2.50    1.00    0.00   -0.25   21.00 ]
[   0.00          0.00   1.50   4.00    0.00    1.00   -0.50    6.00 ]
[   0.00          1.00   0.25   0.50    0.00    0.00    0.25    9.00 ]
```

```
[   1.00        0.00   -0.25   -0.50   0.00    0.00    0.75   27.00 ]
[   0.00        0.00    0.75    2.50   1.00    0.00   -0.25   21.00 ]
[   0.00        0.00    1.50    4.00   0.00    1.00   -0.50    6.00 ]
[   0.00        1.00    0.25    0.50   0.00    0.00    0.25    9.00 ]

[   1.00        0.00   -0.06    0.00   0.00    0.12    0.69   27.75 ]
[   0.00        0.00   -0.19    0.00   1.00   -0.62    0.06   17.25 ]
[   0.00        0.00    0.38    1.00   0.00    0.25   -0.12    1.50 ]
[   0.00        1.00    0.06    0.00   0.00   -0.12    0.31    8.25 ]

[   1.00        0.00   -0.06    0.00   0.00    0.12    0.69   27.75 ]
[   0.00        0.00   -0.19    0.00   1.00   -0.62    0.06   17.25 ]
[   0.00        0.00    0.38    1.00   0.00    0.25   -0.12    1.50 ]
[   0.00        1.00    0.06    0.00   0.00   -0.12    0.31    8.25 ]

[   1.00        0.00    0.00    0.17   0.00    0.17    0.67   28.00 ]
[   0.00        0.00    0.00    0.50   1.00   -0.50    0.00   18.00 ]
[   0.00        0.00    1.00    2.67   0.00    0.67   -0.33    4.00 ]
[   0.00        1.00    0.00   -0.17   0.00   -0.17    0.33    8.00 ]

---------------------------
Output:
[   1.00        0.00    0.00    0.17   0.00    0.17    0.67   28.00 ]
[   0.00        0.00    0.00    0.50   1.00   -0.50    0.00   18.00 ]
[   0.00        0.00    1.00    2.67   0.00    0.67   -0.33    4.00 ]
[   0.00        1.00    0.00   -0.17   0.00   -0.17    0.33    8.00 ]

result =  solved
        z =  28.0
        x =  [8.0, 4.0, 0, 18.0, 0, 0]
---------------------------
===========================
Submitted by  Becca Little
LUP-SOLVE:
---------------------------
Input:
[   5.00        6.00    3.00 ]
[   0.20        0.80   -0.60 ]
[   0.60        0.50    2.50 ]
```

RHS:
[3.0, 7.0, 8.0]
---
x = [-1.4, 2.2, 0.6]



Reflection:

The LU and LUP decompositions were extremely easy to implement; they were very nearly direct mappings from the psuedocode in the book, with the exception of representing the L and U matrices more efficiently as a single LU matrix. Combined, these probably took me an hour to completely finish including testing.

Simplex took a lot more time. I initially had a lot of difficulty understanding the psuedocode in the book for the purpose of the problem set, and it was further complicated by the tableau representation Dr. Femiani implemented in the scaffolding. It took me way longer than I care to mention to decipher this representation; his tableau layout is different than most tutorials (even the one posted under lectures). However, once I got to that point the simplex function was straightforward. Next, I had to figure out how to do the pivot function. I completely understand how to pivot and the concept behind it, but the psuedocode in the book was confusing. I emailed the TA several days before the due date but he wasn't able to get back to

me in time. Once he did, his answer helped me understand how to pivot using the tableau representation (which I find much easier to visualize than the representation in the book).

My complexity levels were well below the required 10. The colorized rows and columns were helpful in debugging. Dr. Femiani's provided solutions were probably the most helpful in finishing up my solution