```python
# -*- coding: utf-8 -*-
"""
Created on Thu Jun 10 21:50:10 2021

@author: SESWARAN
"""

#%% Library Import
import numpy as np
from math import log10,floor
import sys
#%% Significant digit round up
def round_sig(x, sig):
    return round(x, sig-int(floor(log10(abs(x))))-1)


#%% Get User Input & Generate Random 2X2 Matrix
print('Number of unknowns = 2. i.e. 2X2 matrix')
unknowns = 2
elementDtype =  int(input('Enter number of data type for martix elements (int = 1/ float = 2):'))
sigDigits = int(input('Enter the number of significant digits (d):'))

# Generate Randon martix
if elementDtype == 2:
    matrixA = np.random.rand(unknowns,unknowns+1)
    # significant digit reduction
    for i in range(unknowns):
        for j in range (unknowns+1):
            matrixA[i][j] = round_sig(matrixA[i][j],sigDigits)
    del i,j

elif elementDtype == 1:
    rangeRandom = int(input('Enter max value range for array values (e.g. 10):'))
    matrixA = np.random.randint(rangeRandom,size=(unknowns,unknowns+1))
else:
    print('Incorrect Data type entered. Enter Correct Data type!!')

# Print Matrix
print('Linear system equations are:')
print(f'{matrixA[0][0]}x + {matrixA[0][1]}y = {matrixA[0][2]}')
print(f'{matrixA[1][0]}x + {matrixA[1][1]}y = {matrixA[1][2]}')

#%% Applying Gauss Elimination without Pivot i.e Back Substitution
x = np.zeros(unknowns)
for i in range(unknowns):
    if matrixA[i][i] == 0.0:
        sys.exit('Divide by zero detected!')

    for j in range(i+1, unknowns):
        ratio = matrixA[j][i]/matrixA[i][i]

        for k in range(unknowns+1):
            matrixA[j][k] = matrixA[j][k] - ratio * matrixA[i][k]

# Back Substitution
x[unknowns-1] = matrixA[unknowns-1][unknowns]/matrixA[unknowns-1][unknowns-1]

for i in range(unknowns-2,-1,-1):
    x[i] = matrixA[i][unknowns]

    for j in range(i+1,unknowns):
        x[i] = x[i] - matrixA[i][j]*x[j]

    x[i] = x[i]/matrixA[i][i]

# Displaying solution
print('\n Values for unknowns are using elimination method: ')
print(f'x ={round_sig(x[0],sigDigits)} ')
print(f'y ={round_sig(x[1],sigDigits)}')

#%%  Applying Gauss Elimination with Pivot
```

```
72  for i in range(0,unknowns-2):      # Loop through the columns of the matrix
73
74      if np.abs(matrixA[i,i])==0:
75          for k in range(i+1,unknowns-1):
76              if np.abs(matrixA[k,i])>np.abs(matrixA[i,i]):
77                  matrixA[[i,k]]=matrixA[[k,i]]             # Swaps ith and kth rows
                    to each other
78                  break
79
80      for j in range(i+1,unknowns-1):      # Loop through rows below diagonal for each
        column
81          m = matrixA[j,i]/matrixA[i,i]
82          matrixA[j,:] = matrixA[j,:] - m*matrixA[i,:]
83
84  y = np.zeros(unknowns)
85  # Back Substitution
86  y[unknowns-1] = matrixA[unknowns-1][unknowns]/matrixA[unknowns-1][unknowns-1]
87
88  for i in range(unknowns-2,-1,-1):
89      y[i] = matrixA[i][unknowns]
90
91      for j in range(i+1,unknowns):
92          y[i] = y[i] - matrixA[i][j]*x[j]
93
94      y[i] = y[i]/matrixA[i][i]
95
96  # Displaying solution
97  print('\n Values for unknowns are using Pivoting method: ')
98  print(f'x ={round_sig(y[0],sigDigits)} ')
99  print(f'y ={round_sig(y[1],sigDigits)}')
```