```
#Jack Amos
#CS2300-002
#11/4/2019
#Project 3
#Python 3.7



def printSysProd(sys,prod,col_round):

	count = len(sys)
	row_string = ""

	#prints matrix heading
	if col_round > 0:
		print("\nColumn %d"%(col_round))
	else:
		print("\nSystem matrix:")

	#prints matrix
	while count != 0:

		for n in sys:

			for m in n:
				index = sys.index(n)
				row_string+=str("{:.4f}".format(m))+"  "
			row_string+="x  "+str("{:.4f}".format(prod[index]))+"\n"
			print(row_string)
			row_string = ""
			count-=1



def sysCheck(sys):

	num_zeros = 0
	count = len(sys)-1

	while count != 0:
		num_zeros+=count
		count-=1


	#sets completed rows corresponding index to 0
	zero_count = 0


	for n in sys:
```

```
            for m in n:
                if m == 0.0:
                    zero_count+=1

        if zero_count < num_zeros:

            return False

        return True



def solveForX(sys,prod):

        #sys
        with open(sys,'r') as file:
            contents = file.read()

        #get each value in string an convert it to float
        sys_values = [float(x) for x in contents.split()]

        #prod
        with open(prod,'r') as file:
            contents = file.read()

        #get each value in string an convert it to float
        prod_values = [float(x) for x in contents.split()]


        #determine matrix size
        size = int(sys_values[0])

        #remove size data from lists to make operations easier
        sys_values.remove(size)
        prod_values.remove(size)

        #making sys into 2d array so it can be processed easier
        count = size
        iterator = size
        start = 0
        end = size
        temp = []
        sysFinal = []

        while count != 0:

            temp = sys_values[start:end]
            sysFinal.append(temp)
            temp = []
```

```
            count-=1
            start+=iterator
            end+=iterator


        #make necessary 0's
        check = False
        col_round = 1
        top = 0
        top_value = 0.0
        current = 0
        current_value = 0.0

        printSysProd(sysFinal,prod_values,0)

        while check == False:

            #gaussian solving logic
            top_value = sysFinal[top][top]
            current = top + 1

            while current < len(sysFinal):

                current_value = sysFinal[current][top]

                if current_value != 0.0:

                    row_length = 0
                    prod_values[current] = (prod_values[current] * top_value * -1) +
(prod_values[top] * current_value)

                    while row_length < len(sysFinal):

                        sysFinal[current][row_length] = (sysFinal[current][row_length] *
top_value * -1) + (sysFinal[top][row_length] * current_value)

                        row_length+=1


                current+=1

            top+=1


            #set value to 0 if below or at minimum of .0000001, -.0000001, -0, and keeps values
from becoming too small
            for n in sysFinal:
                for m in n:
```

```
                        if m <= .0000001 and m > 0.0:
                                n[n.index(m)] = 0.0
                        elif m >= -.0000001 and m < 0.0:
                                n[n.index(m)] = 0.0
                        elif m == -0.0:
                                n[n.index(m)] = 0.0

                for n in prod_values:

                        if n <= .0000001 and n > 0.0:
                                n = 0.0
                        elif n >= -.0000001 and n < 0.0:
                                n = 0.0
                        elif n == -0.0:
                                n = 0.0


                #reduces size of floating point values without removing them
                big_count = 0
                current_array = []

                for n in sysFinal:

                        current_array = sysFinal[sysFinal.index(n)]

                        if prod_values[sysFinal.index(n)] > 100 or prod_values[sysFinal.index(n)] < -
100:

                                prod_values[sysFinal.index(n)] = prod_values[sysFinal.index(n)]/100

                                while big_count < len(sysFinal):

                                        current_array[big_count] = current_array[big_count]/100

                                        big_count+=1

                        big_count = 0


                printSysProd(sysFinal,prod_values,col_round)
                col_round+=1

                check = sysCheck(sysFinal)

        #solve for x values, starting at bottom
        x_values = []
        i = 0

        #make array of x's based on matrix size
```

```
        while i < len(sysFinal):

                x_values.append(1)
                i+=1

        i-=1
        solveX = 0.0
        f = len(sysFinal)-1
        current_array = sysFinal[len(sysFinal)-1]
        divVal = 0.0

        #gets sum of all current x's and row values then subtracts non important x positions and divides
product value by important x
        while f > -1:

                while i > -1:

                        if i != f:
                                solveX += x_values[i] * current_array[i]
                        elif i == f:
                                divVal = current_array[i]
                        i-=1

                x_values[f] = (prod_values[f] - solveX)/divVal
                f-=1
                i = len(sysFinal)-1
                solveX = 0.0
                current_array = sysFinal[f]


        #print x values
        print("\nSolution Vector:")

        for n in x_values:

                print("%.4f"%n)



solveForX("sysmat1.txt","prodvec1.txt")
solveForX("sysmat2.txt","prodvec2.txt")
```