Problem on Sudoku (Magic Squares)

--------------------------------------------------------------------------------------------------

Construct a 9 x 9 matrix and assign numbers sequentially from 1 through 81, so that the row totals are the same, the column totals are the same, and the diagonals also add up to the same number.

My answer:

Sum of the first n natural numbers = n \* (n + 1) /2

Sum of all the number in the matrix = 81 \* (81 + 1) / 2.   = 81 \* 41  =  3321.

Each Row total = 3321 / 9 = 369

Each column total = 369

Each diagonal total along the main diagonals = 369

My algorithm:

*1. Start by assigning 1 to the middle column of the topmost row ie. (n, (n + 1)//2), and then assign  
 sequential numbers until n \* n (including it) as given below in steps 2 through 5.  
2. Assign the next number to position with 1 row above and 1 column above.  
3. If row exceeds n, then go to row 1 (wrap around)  
4. If column exceeds n, then go to column 1 (wrap around)  
5. If the number is divisible by n, then the next number is placed immediately below it.*

Here is my program:

*# MagicSquaresWithDictionary.py  
  
'''  
 Program: Construct a matrix of size n x n where n is odd and n = 3, 5, 7, or 9  
 with sequential numbers from 1 to n \* n such that the row totals, column totals and the diagonal totals  
 are the same.  
  
 Algorithm:  
 1. Start by assigning 1 to the middle column of the topmost row ie. (n, (n + 1)//2), and then assign  
 sequential numbers until n \* n (including it) as given below in steps 2 through 5.  
 2. Assign the next number to position with 1 row above and 1 column above.  
 3. If row exceeds n, then go to row 1 (wrap around)  
 4. If column exceeds n, then go to column 1 (wrap around)  
 5. If the number is divisible by n, then the next number is placed immediately below it.  
'''***def** createMatrix(n):  
 *'''  
 This function creates a matrix with rows numbers 1 through n from bottom to top, and columns numbered 1 through  
 n from left to right.  
 Note: I used a dictionary implementation since using a list (or an array) in python would have 0 -index.  
 I now have a matrix in the dictionary with rows 1 through n and columns 1 through n.* **:param** *n: the number of rows in the matrix (also the number of columns in the matrix)* **:return***: matrix which is list of lists.  
 '''* mydict = {} *# Initialize a dictionary with n x n tuples to 0.* i = n  
 j = (n + 1)//2  
 k = 1  
 mydict[(i, j)] = k  
 k += 1  
 print(**"mydict = "**, mydict)  
 **while** k <= n \* n:  
 **if** (k - 1) % n == 0: *# previous number is divisible by n* i -= 1  
 j = j  
 **else**:  
 i += 1  
 j += 1  
 **if** i > n:  
 i = 1  
 **if** j > n:  
 j = 1  
 mydict[(i, j)] = k  
 print(**"i = "**, i, **"j = "**, j, **"k = "**, k, **"mydict[(i, j)] = "**, mydict[(i, j)])  
 k += 1  
  
 print(**"mydict after populating number = "**)  
 print(mydict)  
 print(**"Create matrix"**)  
 matrix = []  
 **for** i **in** range(1, n + 1):  
 row = []  
 **for** j **in** range(1, n + 1):  
 row.append(mydict[(i,j)])  
 matrix.append(row)  
 matrix = matrix[::-1]  
 print(**"Printing matrix"**)  
 **for** row **in** matrix:  
 print(row)  
 print(**"---------------------"**)  
 **return** matrix  
  
**def** main():  
 n = 9  
 matrix = createMatrix(n)  
 print(**"In main, n = "**, n)  
 **for** i **in** range(n):  
 **for** j **in** range(n):  
 print(matrix[i][j], end = **' '**)  
 print()  
 print(**"-----------------------------------"**)  
  
 print(**"Row Totals = "**)  
 **for** row **in** matrix:  
 print(sum(row))  
 print(**"-----------------------------------"**)  
 print(**"Column Totals = "**)  
 **for** i **in** range(n):  
 columnTotal = 0  
 **for** j **in** range(n):  
 columnTotal += matrix[i][j]  
 print(columnTotal)  
 print(**"-----------------------------------"**)  
 print(**"Main Diagonal Total "**)  
 mainDiagTotal = 0  
 **for** i **in** range(n):  
 mainDiagTotal += matrix[i][i]  
 print(**"Main Diagonal Total = "**, mainDiagTotal)  
 print(**"-----------------------------------"**)  
 print(**"Secondary Diagonal Total"**)  
 secDiagTotal = 0  
 **for** i **in** range(n):  
 secDiagTotal += matrix[n - i - 1][i]  
 print(**"Secondary Diagonal Total = "**, secDiagTotal)  
 print(**"---------------------------------"**)  
 print(**"Grand Total"**)  
 grandTotal = 0  
 **for** i **in** range(n):  
 **for** j **in** range(n):  
 grandTotal += matrix[i][j]  
 print(**"Grand Total = "**, grandTotal)  
 print(**"---------------------------------"**)  
  
  
  
**if** \_\_name\_\_==**'\_\_main\_\_'**:  
 main()

################################################

# Run Output

/usr/local/bin/python3.7 /Users/jayashrijagannathan/PycharmProjects/DataStruct\_Alg/MagicSquaresWithDictionary.py  
mydict =  {(9, 5): 1}  
i =  1 j =  6 k =  2 mydict[(i, j)] =  2  
i =  2 j =  7 k =  3 mydict[(i, j)] =  3  
i =  3 j =  8 k =  4 mydict[(i, j)] =  4  
i =  4 j =  9 k =  5 mydict[(i, j)] =  5  
i =  5 j =  1 k =  6 mydict[(i, j)] =  6  
i =  6 j =  2 k =  7 mydict[(i, j)] =  7  
i =  7 j =  3 k =  8 mydict[(i, j)] =  8  
i =  8 j =  4 k =  9 mydict[(i, j)] =  9  
i =  7 j =  4 k =  10 mydict[(i, j)] =  10  
i =  8 j =  5 k =  11 mydict[(i, j)] =  11  
i =  9 j =  6 k =  12 mydict[(i, j)] =  12  
i =  1 j =  7 k =  13 mydict[(i, j)] =  13  
i =  2 j =  8 k =  14 mydict[(i, j)] =  14  
i =  3 j =  9 k =  15 mydict[(i, j)] =  15  
i =  4 j =  1 k =  16 mydict[(i, j)] =  16  
i =  5 j =  2 k =  17 mydict[(i, j)] =  17  
i =  6 j =  3 k =  18 mydict[(i, j)] =  18  
i =  5 j =  3 k =  19 mydict[(i, j)] =  19  
i =  6 j =  4 k =  20 mydict[(i, j)] =  20  
i =  7 j =  5 k =  21 mydict[(i, j)] =  21  
i =  8 j =  6 k =  22 mydict[(i, j)] =  22  
i =  9 j =  7 k =  23 mydict[(i, j)] =  23  
i =  1 j =  8 k =  24 mydict[(i, j)] =  24  
i =  2 j =  9 k =  25 mydict[(i, j)] =  25  
i =  3 j =  1 k =  26 mydict[(i, j)] =  26  
i =  4 j =  2 k =  27 mydict[(i, j)] =  27  
i =  3 j =  2 k =  28 mydict[(i, j)] =  28  
i =  4 j =  3 k =  29 mydict[(i, j)] =  29  
i =  5 j =  4 k =  30 mydict[(i, j)] =  30  
i =  6 j =  5 k =  31 mydict[(i, j)] =  31  
i =  7 j =  6 k =  32 mydict[(i, j)] =  32  
i =  8 j =  7 k =  33 mydict[(i, j)] =  33  
i =  9 j =  8 k =  34 mydict[(i, j)] =  34  
i =  1 j =  9 k =  35 mydict[(i, j)] =  35  
i =  2 j =  1 k =  36 mydict[(i, j)] =  36  
i =  1 j =  1 k =  37 mydict[(i, j)] =  37  
i =  2 j =  2 k =  38 mydict[(i, j)] =  38  
i =  3 j =  3 k =  39 mydict[(i, j)] =  39  
i =  4 j =  4 k =  40 mydict[(i, j)] =  40  
i =  5 j =  5 k =  41 mydict[(i, j)] =  41  
i =  6 j =  6 k =  42 mydict[(i, j)] =  42  
i =  7 j =  7 k =  43 mydict[(i, j)] =  43  
i =  8 j =  8 k =  44 mydict[(i, j)] =  44  
i =  9 j =  9 k =  45 mydict[(i, j)] =  45  
i =  8 j =  9 k =  46 mydict[(i, j)] =  46  
i =  9 j =  1 k =  47 mydict[(i, j)] =  47  
i =  1 j =  2 k =  48 mydict[(i, j)] =  48  
i =  2 j =  3 k =  49 mydict[(i, j)] =  49  
i =  3 j =  4 k =  50 mydict[(i, j)] =  50  
i =  4 j =  5 k =  51 mydict[(i, j)] =  51  
i =  5 j =  6 k =  52 mydict[(i, j)] =  52  
i =  6 j =  7 k =  53 mydict[(i, j)] =  53  
i =  7 j =  8 k =  54 mydict[(i, j)] =  54  
i =  6 j =  8 k =  55 mydict[(i, j)] =  55  
i =  7 j =  9 k =  56 mydict[(i, j)] =  56  
i =  8 j =  1 k =  57 mydict[(i, j)] =  57  
i =  9 j =  2 k =  58 mydict[(i, j)] =  58  
i =  1 j =  3 k =  59 mydict[(i, j)] =  59  
i =  2 j =  4 k =  60 mydict[(i, j)] =  60  
i =  3 j =  5 k =  61 mydict[(i, j)] =  61  
i =  4 j =  6 k =  62 mydict[(i, j)] =  62  
i =  5 j =  7 k =  63 mydict[(i, j)] =  63  
i =  4 j =  7 k =  64 mydict[(i, j)] =  64  
i =  5 j =  8 k =  65 mydict[(i, j)] =  65  
i =  6 j =  9 k =  66 mydict[(i, j)] =  66  
i =  7 j =  1 k =  67 mydict[(i, j)] =  67  
i =  8 j =  2 k =  68 mydict[(i, j)] =  68  
i =  9 j =  3 k =  69 mydict[(i, j)] =  69  
i =  1 j =  4 k =  70 mydict[(i, j)] =  70  
i =  2 j =  5 k =  71 mydict[(i, j)] =  71  
i =  3 j =  6 k =  72 mydict[(i, j)] =  72  
i =  2 j =  6 k =  73 mydict[(i, j)] =  73  
i =  3 j =  7 k =  74 mydict[(i, j)] =  74  
i =  4 j =  8 k =  75 mydict[(i, j)] =  75  
i =  5 j =  9 k =  76 mydict[(i, j)] =  76  
i =  6 j =  1 k =  77 mydict[(i, j)] =  77  
i =  7 j =  2 k =  78 mydict[(i, j)] =  78  
i =  8 j =  3 k =  79 mydict[(i, j)] =  79  
i =  9 j =  4 k =  80 mydict[(i, j)] =  80  
i =  1 j =  5 k =  81 mydict[(i, j)] =  81  
mydict  after populating number =   
{(9, 5): 1, (1, 6): 2, (2, 7): 3, (3, 8): 4, (4, 9): 5, (5, 1): 6, (6, 2): 7, (7, 3): 8, (8, 4): 9, (7, 4): 10, (8, 5): 11, (9, 6): 12, (1, 7): 13, (2, 8): 14, (3, 9): 15, (4, 1): 16, (5, 2): 17, (6, 3): 18, (5, 3): 19, (6, 4): 20, (7, 5): 21, (8, 6): 22, (9, 7): 23, (1, 8): 24, (2, 9): 25, (3, 1): 26, (4, 2): 27, (3, 2): 28, (4, 3): 29, (5, 4): 30, (6, 5): 31, (7, 6): 32, (8, 7): 33, (9, 8): 34, (1, 9): 35, (2, 1): 36, (1, 1): 37, (2, 2): 38, (3, 3): 39, (4, 4): 40, (5, 5): 41, (6, 6): 42, (7, 7): 43, (8, 8): 44, (9, 9): 45, (8, 9): 46, (9, 1): 47, (1, 2): 48, (2, 3): 49, (3, 4): 50, (4, 5): 51, (5, 6): 52, (6, 7): 53, (7, 8): 54, (6, 8): 55, (7, 9): 56, (8, 1): 57, (9, 2): 58, (1, 3): 59, (2, 4): 60, (3, 5): 61, (4, 6): 62, (5, 7): 63, (4, 7): 64, (5, 8): 65, (6, 9): 66, (7, 1): 67, (8, 2): 68, (9, 3): 69, (1, 4): 70, (2, 5): 71, (3, 6): 72, (2, 6): 73, (3, 7): 74, (4, 8): 75, (5, 9): 76, (6, 1): 77, (7, 2): 78, (8, 3): 79, (9, 4): 80, (1, 5): 81}  
Create matrix  
Printing matrix  
[47, 58, 69, 80, 1, 12, 23, 34, 45]  
[57, 68, 79, 9, 11, 22, 33, 44, 46]  
[67, 78, 8, 10, 21, 32, 43, 54, 56]  
[77, 7, 18, 20, 31, 42, 53, 55, 66]  
[6, 17, 19, 30, 41, 52, 63, 65, 76]  
[16, 27, 29, 40, 51, 62, 64, 75, 5]  
[26, 28, 39, 50, 61, 72, 74, 4, 15]  
[36, 38, 49, 60, 71, 73, 3, 14, 25]  
[37, 48, 59, 70, 81, 2, 13, 24, 35]  
---------------------  
In main,  n =  9  
47    58    69    80    1    12    23    34    45      
57    68    79    9    11    22    33    44    46      
67    78    8    10    21    32    43    54    56      
77    7    18    20    31    42    53    55    66      
6    17    19    30    41    52    63    65    76      
16    27    29    40    51    62    64    75    5      
26    28    39    50    61    72    74    4    15      
36    38    49    60    71    73    3    14    25      
37    48    59    70    81    2    13    24    35      
-----------------------------------  
Row Totals =   
369  
369  
369  
369  
369  
369  
369  
369  
369  
-----------------------------------  
Column Totals =   
369  
369  
369  
369  
369  
369  
369  
369  
369  
-----------------------------------  
Main Diagonal Total   
Main Diagonal Total =  369  
-----------------------------------  
Secondary Diagonal Total  
Secondary Diagonal Total =  369  
---------------------------------  
Grand Total  
Grand Total =  3321  
---------------------------------  
  
Process finished with exit code 0

========================================================