MAGIC SQUARE, MINI PRESENTATION

19CSE100 - PROBLEM SOLVING AND ALGORITHMIC THINKING MINI PRESENTATION

Presented by ANASWARA SURESH . M . K
CB.EN.U4CYS22007
TIFAC-CORE in Cyber Security
Amrita Vishwa Vidyapeetham, Coimbatore Campus

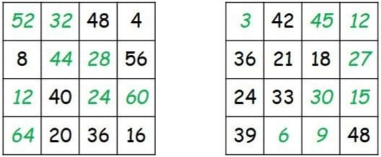
MARCH 1, 2023

TABLE OF CONTENTS

- WHAT IS A MAGIC SQUARE ?
- EXAMPLES OF MAGIC SQUARE .
- PROBLEM ANALYSIS .
- PROBLEM DISSECTION .
- ALGORITHM
- LIMITATIONS OF FLOWGORITHM FOR MAGIC SQUARE .

1.WHAT IS A MAGIC SQUARE ?

A magic square is a square grid of numbers in which the numbers in each row, column, and diagonal add up to the same number. The numbers in the square can be any integers, but are often used with consecutive numbers starting with 1. The most famous magic square is the Lo Shu Square, which is a 3x3 square and is considered to have ancient Chinese origin. Magic squares have been studied for centuries and have been found to have many mathematical properties and applications..



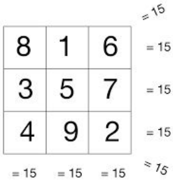
52	*32*	48	4
8	*44*	*28*	56
12	40	*24*	*60*
64	20	36	16

3	42	*45*	*12*
36	21	18	*27*
24	33	*30*	*15*
39	*6*	*9*	48

Figure: Examples of 4x4 magic squares

EXAMPLES OF MAGIC SQUARE

One example of a 3x3 magic square is:



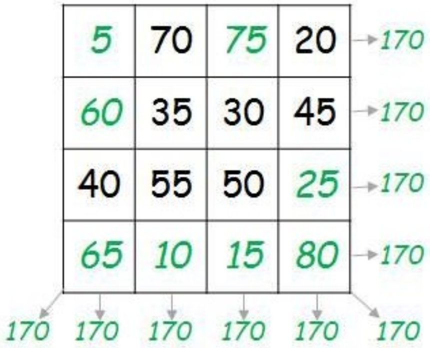
8	1	6
3	5	7
4	9	2

Figure: 3x3 magic square

In this square, the sum of the numbers in each row, column, and diagonal is 15. For example, the sum of the numbers in the first row is 8 + 1 + 6 = 15, and the sum of the numbers in the first column is 8 + 3 + 4 = 15. Similarly, the sum of the numbers in the diagonal from top left to bottom right is 8 + 5 + 2 = 15, and the sum of the numbers in the diagonal from top right to bottom left is 6 + 5 + 4 = 15.

EXAMPLES OF MAGIC SQUARE

One example of a 4x4 magic square is:



5	70	75	20	170
60	35	30	45	170
40	55	50	25	170
65	10	15	80	170

170
170
170
170
170
170

Figure: 4x4 magic square

PROBLEM ANALYSIS

Algorithm complexity

Depending on the method used to generate the magic square, the algorithm can become quite complex. This can make it difficult to write, debug, and maintain the code.

Handling large square sizes

As the size of the square increases, the number of possible permutations and combinations also increases, which can significantly slow down the algorithm.

Memory management

Generating a magic square can require a large amount of memory, particularly for large square sizes. This can cause issues with memory management and can lead to performance bottlenecks.

Symmetry and rotations

Some methods for generating magic squares, such as the Siamese Method, involve rotating and reflecting the square to generate all other squares in the same class. This can add complexity to the algorithm and make it more difficult to implement in a program.

Error handling

It is important to handle errors such as input validation, unexpected input, and unexpected output, as this can cause issues with the program and make it more difficult to debug.

Efficiency

Depending on the method used to generate the magic square, the algorithm may be computationally expensive. This can make it difficult to generate large squares in a reasonable amount of time.

Input validation

Some methods may require specific inputs such as odd size, prime number, and othe specific constraint, it is important to validate the input and handle any errors that ma occur.

PROBLEM DISSECTION

Initialization

Start by initializing a 2D square grid with the desired size, typically filled with zeroes or a placeholder value

Positioning:

Determine the starting position for filling in the numbers, typically the middle of the first row.

Number generation

Generate the numbers to fill in the square, typically starting with the number 1 and incrementing by 1 for each subsequent square.

Number placement

Place the generated numbers in the appropriate squares according to the rules of a magic square (where the sum of each row, column, and diagonal is the same). This typically involves moving diagonally up and to the right, wrapping around to the other side of the square if you reach the edge, and moving down one square instead if the square you are moving to is already filled.

Validation

Verify that the resulting square is indeed a magic square by checking that the sum of each row, column, and diagonal is the same.

Output

Output the magic square in the desired format.

Optimization

Optimize the algorithm to be more efficient and/or handle larger square sizes.

ALGORITHM

Here is a simple algorithm for generating a magic square using the Siamese method in pseudo code:

- Create an n x n square with the numbers 1 through n^2
- Place the number 1 in the middle column of the top row
- For each remaining number, i:
  1. Move one square diagonally up and to the right
  2. If the new position is outside the square, move to the opposite side of the square
  3. If the new position is already occupied, move one square down
  4. Place the number i in the new position
- Repeat steps 3a-3d for all remaining numbers
- Use rotations and reflections to generate all other magic squares in the same class

LIMITATIONS OF FLOWGORITHM FOR MAGIC SQUARE

1. Creating a magic square is a complex task,if the square is large. This may make it difficult to represent the logic and calculations in a flowchart.

2. Flowgorithm is a visual representation of a program's logic and is not intended for writing code. Some of the more complex mathematical operations

3. Flowgorithm does not provide debugging tools, making it difficult to identify and fix errors in the logic of the program.

4. Flowgorithm is not designed for handling large amounts of data, so it may not be suitable for creating a magic square for larger squares.

THANK YOU !!

Submitted By

- **ANASWARA SURESH . M . K**

  CB.EN.U4CYS22007

- **ANURAG REDDY**

  CB.EN.U4CYS22069

- **VISHAL R S**

  CB.EN.U4CYS22073

References

Ramaguru, R., Sindhu, M., Sethumadhavan, M. (2019). Blockchain for the Internet of Vehicles. In: Singh, M., Gupta, P., Tyagi, V., Flusser, J., Ören, T., Kashyap, R. (eds) Advances in Computing and Data Sciences. ICACDS 2019. Communications in Computer and Information Science, vol 1045. Springer, Singapore. https://doi.org/10.1007/978-981-13-9939-8_37