CS 202 – Assignment #4

Purpose: Learn class inheritance, dynamic allocation, pointers, and make utility.
Points: 100

Assignment:

In recreational mathematics, a Magic Square[1] is an arrangement of numbers (usually integers) in a square grid, where the numbers in each row, and in each column, and the numbers in the forward and backward main diagonals, all add up to the same number. A magic square has the same number of rows as it has columns, typically referred to as the *order*. A magic square that contains the integers from 1 to n^2 is called a normal magic square.
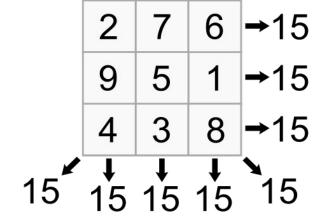

Design and implement two C++ classes to provide a normal Magic Square creation program. The classes we will implement are as follows:

- **Board Class**
  - Implement a generic board class that can be used for any type of square application. Such applications include chess, checkers, bingo, and Sudoku. This class may be used in future assignments. The class will dynamically create a square two dimensional array based on the passed order. Functions will get and set various board locations and include error checking of the indexes based on the current board order.

- **Magic Square Class**
  - Implement a class that uses the board class and creates normal magic squares based on the specified order. A magic square is-a board (square) with special rules and added functionality. The algorithms for creating normal magic squares are provided. Understanding the algorithm is more challenging that coding.

For all implementation code, points will be deducted for poor or especially inefficient solutions.

---

1   For more information, refer to: http://en.wikipedia.org/wiki/Magic_square

Development and Testing

In order to simplify development and testing, the project is split into two (2) main parts; board class and magic square class.

- The board class must be developed and fully operational before attempting the magic square class. The board class can be developed and tested independently of the other class. An independent main, *brdMain*, that uses and tests only the board class is provided and it can be built independently using the provided main file (see next section).

- The magic square class uses the board class and will implement the normal magic square algorithms based on the order (odd, singly even, or doubly even). It is suggested to to develop and test one algorithm at a time. For example, develop and test the even algorithm before attempting the other algorithms. The example output can help in checking the intermediate results.
  - The provided main, *magicSqr*, will perform a series of tests on the magic square object. If some command line arguments are provided, it just create a normal magic square of the specified order. For example;

    ```
    ./magicSqr -o 5
    ```

    will create and display a magic square of order 5. If no command line arguments are provided, it will execute a series of predefined tests (including invalid values).

Make File:

The provided make file assumes the source file names include (*brdMain.cpp*, *boardType.h*, *boardTypeImp.cpp*, *magicSqr.cpp*, *magicSquareType.h*, *magicSquareImp.cpp*). To build the provided main for board testing;

```
make brdMain
```

And to build the provided main for the magic square class;

```
make
```

Which will create the *brdMain* (for *boardType* Class) or the *magicSqr* executable.

Submission:

- It is **strongly** suggested that you submit each part of the program for testing as they are developed. For example, when the board class is completed, it can be submitted which will allow testing of that portion of the project. This will ensure it is fully working before moving on to the next part.
  - If you find any issues, they can be corrected and resubmitted for testing and unlimited number of times (before the due date/time).

- All files must compile and execute on Ubuntu and compiler with C++11.

- Submit source files
  - *Note*, do **not** submit the provided mains (we have them).

- Once you submit, the system will score the project and provide feedback.
  - If you do not get full score, you can (and should) correct and resubmit.
  - You can re-submit an unlimited number of times before the due date/time.

- Late submissions will be accepted for a period of 24 hours after the due date/time for any given lab. Late submissions will be subject to a ~2% reduction in points per an hour late. If you submit 1 minute - 1 hour late -2%, 1-2 hours late -4%, … , 23-24 hours late -50%. This means after 24 hours late submissions will receive an automatic 0.

Program Header Block

All program and header files must include your name, section number, assignment, NSHE number, input and output summary. The required format is as follows:

```
/*
Name: MY_NAME, NSHE, CLASS-SECTION, ASSIGNMENT
Description: <per assignment>
Input: <per assignment>
Output: <per assignment>
*/
```

Failure to include your name in this format will result in a loss of up to 5%.

Code Quality Checks

A C++ linter[2] is used to perform some basic checks on code quality. These checks include, but are not limited to, the following:

- Unnecessary 'else' statements should not be used.
- Identifier naming style should be either camelCase or snake_case (consistently chosen).
- Named constants should be used in place of literals.
- Correct indentation should be used.
- Redundant return/continue statements should not be used.
- Selection conditions should be in the simplest form possible.
- Function prototypes should be free of top level *const*.

Not all of these items will apply to every program. Failure to to address these guidelines will result in a loss of up to 5%.

Scoring Rubric

Scoring will include functionality, code quality, and documentation. Below is a summary of the scoring rubric for this assignment.

Criteria	Weight	Summary
Compilation	-	Failure to compile will result in a score of 0.
Program Header	5%	Must include header block in the required format (see above).
General Comments	10%	Must include an appropriate level of program documentation.
Line Length	5%	No lines should exceed more than eighty (80) characters.
Code Quality	5%	Must meet some basic code quality checks (see above)
Program Functionality		Program must meet the functional requirements as outlined in the assignment. Must be submitted on time for full score.
• Board Type	30%	
• Magic Square	45%	

---

2 For more information, refer to: https://en.wikipedia.org/wiki/Lint_(software)

Board Class

Create a board class, *boardType*, to provide basic square board functionality.

boardType
#size: int
#**board: int
-BRD_SIZE_MIN=3: static constexpr int
-BRD_SIZE_MAX=30: static constexpr int
+boardType(int)
+~boardType()
+setCell(int, int, int): void
+getCell(int, int) const: int
+getOrder() const: int
+printGrid() const: void

Your implementation and header files should be fully commented. This class may be used on a future assignment.

Function Descriptions

The following are more detailed descriptions of the required functions.

- The constructor *boardType(int)* should verify the passed size (between SIZE_MIN and SIZE_MAX, inclusive). If the size is valid, an integer two-dimensional array, *size* x *size*, should be dynamically created and all values set to 0. If the size is invalid, a message should be displayed (e.g., "Error, invalid board size." and "No board created.") and the *board* class variable set to NULL and the *bSize* set to 0. Refer to the example executions for output formatting.
- The destructor *~boardType()* should delete the dynamically allocated memory (if any). The destructor is called automatically when the object goes out of scope.
- The *getCell(int, int)* function should return the cell contents at the passed row and column location (in that order). The passed row and column must be validated to ensure invalid locations in the array are not accessed. If an invalid location is passed, an error message should be displayed (e.g., "Error, invalid board location.") and a 0 returned.
- The *setCell(int, int, int)* function should set the cell contents at the passed row and column location. The arguments are row, column, and value (in that order). The passed row and column must be validated to ensure invalid locations in the array are not accessed. If an invalid location is passed, an error message should be displayed (e.g., "Error, invalid board location.") and the current value left as is.
- The *getOrder()* function should return the board size.
- The *printGrid()* function should print the grid in a formatted manner (with each value in a small text box). An empty grid should not print anything. Refer to the example executions for output formatting.

Use the provided main, *bMain.cpp*, to demonstrate that the *boardType* class functions correctly. The main and provided makefile reference files *bMain,.cpp*, *boardTypeImp.cpp* and *boardType.h*. Your implementation and header files should be fully commented.

Magic Square Class

A magic square will require an *n*x*n* board so the *magicSquareType* class will inherit the basic grid functionality from the *boardType* class. Create a *magicSquareType* class to provide magic square creation functionality.

magicSquareType
-title: string
+magicSquareType(int, string)
+createMagicSquare(): void
+printMagicSquare() const: void
+readMagicSquare(): void
+validateMagicSquare() const: bool
+clearMagicSquare(): void
+magicNumber() const: int
+getTitle() const: string
+setTitle(string): void

Function Descriptions

The following are more detailed descriptions of the required functions.

- The *magicSquareType(int)* constructor should set the title and use the base class constructor to allow the base class to create the grid.
- The *magicNumber()* function should return the magic number or sum based on the board size or order. The magic number or sum for a normal magic square is computed as follows:

$$magicNumber = n\left(\frac{n^2+1}{2}\right)$$

- The *validateMagicSquare()* function should validate a magic square by ensuring the sum of each column, each row, and each of the two main diagonals sum to the magic number.
- The *getTitle()* function should return the current title value.
- The *setTitle(string)* function should set the title to the passed value.
- The *clearMagicSqaure()* function should reset each cell entry to 0 and clear the title string.
- The *readMagicSquare()* function should prompt the user for a title (ensuring it is not empty). The title may be multiple words. Then read integers, one row at a time (based on the board size for the current object).
- The *printMagicSquare()* function should display some header information ("CS 202 – Magic Squares", the title, order, and magic number. The headers should be displayed in bold when printed to the screen. The function should use the base class print function (and thus not repeat the same code). Refer to the example executions for output formatting.
- The *createMagicSquare()* should create a magic square based on the current order using one of the provided algorithms.

Expected Output

This section provides some additional information regarding the expected formatting.

The formatted board includes an *order* x *order* boxed board. The first line (|) should be 3 spaces from the edge. Each box should be line (|), 5 characters, and then line (|). The value should be right justified with one space after the value and the right line (|). The spaces preceding the value will depend on the size of the value. With a maximum order of 30, all values will be at most three (3) digits. Each row should have a top and bottom consisting of three (3) spaces, then order number of groups of one space and five (5) dashes. Refer to the example below.

```
ed-vm% ./magicSqr -o 5

**************************************************************

CS 202 - Magic Squares
  Title: Test
  Magic Square, order: 5
  Magic Number: 65

    _____ _____ _____ _____ _____
   |      |      |      |      |      |
   |  17  |  24  |   1  |   8  |  15  |
   |_____|_____|_____|_____|_____|
   |      |      |      |      |      |
   |  23  |   5  |   7  |  14  |  16  |
   |_____|_____|_____|_____|_____|
   |      |      |      |      |      |
   |   4  |   6  |  13  |  20  |  22  |
   |_____|_____|_____|_____|_____|
   |      |      |      |      |      |
   |  10  |  12  |  19  |  21  |   3  |
   |_____|_____|_____|_____|_____|
   |      |      |      |      |      |
   |  11  |  18  |  25  |   2  |   9  |
   |_____|_____|_____|_____|_____|


   Valid Magic Square.

   ed-vm%
```

The magic square print function uses the base class as described above with some additional headers. The colored text is performed by the provided main. The headers use some boding which can be done as follows;

```
const char* bold = "\033[1m";
const char* unbold = "\033[0m";

cout << bold << "CS 202 - Magic Squares" << endl;
cout << " Title: " << title << endl;
cout << " Magic Square, order: " << bSize << endl;
cout << " Magic Number: " << magicNumber() << unbold << endl;
```

The read magic square function prompting is done as follows;

```
cout << "Enter " << bSize << "x" <<bSize << " Magic Square."
                    << endl << endl;
cout << endl << "Enter Magic Square Values (" << bSize << "x"
                    << bSize << ")" << endl;
```

Magic Square Creation Algorithms:
There are different algorithms for creating magic squares based on the given order.

- **Odd Order → odd order (i.e., 3, 5, 7, …)**
  - Siamese method.
  - Reference: http://en.wikipedia.org/wiki/Siamese_method

  - Algorithm for constructing an *n*x*n* odd ordered Magic Square:
    - First, place a 1 in the middle of the top row.
    - After placing an integer, *k*, move up one row and one column to the right to place the next integer, *k+1*, unless the following occurs:
      - If a move takes you above the top row in the *j*th column, move to the bottom of the *j*th column and place the integer there.
      - If a move takes you outside to the right of the square in the ith row, place the integer in the *i*th row at the left side.
      - If a move takes you to an already filled square or if you move out of the square at the upper right hand corner, place *k+1* immediately below *k*.

    Starting from the central column of the first row with the number 1, the fundamental movement for filling the squares is diagonally up and right, one step at a time. If a filled square is encountered, one moves vertically down one square instead, then continuing as before. When a move would leave the square, it is wrapped around to the last row or first column, respectively.

- **Doubly Even Order → multiple of 4 (i.e., 4, 8, 12, 16, …).**
  - Diagonals Method
  - Reference: http://www.math.wichita.edu/~richardson/mathematics/magic%20squares/order4nmagicsquares.html

  - Algorithm for creating an *n*x*n* doubly even ordered Magic Square:
    - Mark all diagonals in each 4x4 sub-grid of the overall grid. For example:

- Start in the upper left corner and enter the numbers from 1 to n^2, unless the number falls on a cell marked as a diagonal. Cells marked as diagonals should be skipped, however the number count should increase anyway. For example:

	2	3			6	7	
9			12	13			16
17			20	21			24
	26	27			30	31	
	34	35			38	39	
41			44	45			48
49			52	53			56
	58	59			62	63	

- Then, begin at the lower right corner and work back starting from 1 placing the number only in the cells marked as diagonals.

64	2	3	61	60	6	7	57
9	55	54	12	13	51	50	16
17	47	46	20	21	43	42	24
40	26	27	37	36	30	31	33
32	34	35	29	28	38	39	25
41	23	22	44	45	19	18	48
49	15	14	52	53	11	10	56
8	58	59	5	4	62	63	1

- **Singly Even Order → multiple of 2, but not 4 (i.e., 6, 10, 14, 18, …).**
  - Strachey Method
  - Reference: http://en.wikipedia.org/wiki/Strachey_method_for_magic_squares
  - Reference: http://www.math.wichita.edu/~richardson/mathematics/magic%20squares/even-ordermagicsquares.html

  - Algorithm for creating an *n*x*n* singly even ordered Magic Square:
    - Divide the grid into 4 sub-grids, designated **A**, **B**, **C**, **D**. Each grid will be of board size divide by 2 (integer division). For example:

    - Using the Siamese method, complete the individual magics squares **A**, **B**, **C**, and **D** (which are of odd order). Sub-grid **A** is filled starting from 1 (to 25 in this example). Sub-grid **B** starts numbering where **A** left off (26 in this example).

Sub-grid **C** starts numbering where **B** left off (51 in this example). Sub-grid **D** starts numbering where **C** left off (76 in this example). Thus, the numbers are 1 to n^2.

17	24	1	8	15	67	74	51	58	65
23	5	7	14	16	73	55	57	64	66
4	6	13	20	22	54	56	63	70	72
10	12	19	21	3	60	62	69	71	53
11	18	23	2	9	61	68	75	52	59
92	99	76	83	90	42	49	26	33	40
98	80	82	89	91	48	30	32	39	41
79	81	88	95	97	29	31	38	45	47
85	87	94	96	78	35	37	44	46	28
86	93	100	77	84	36	43	50	27	34

- Calculate **nCols** based on the size of the sub-grid divided by 2 (integer division). In this example, that is 5/2 which is 2. Exchange the leftmost **nCols** in sub-grid **A** with the corresponding columns in sub-grid **D**. For example, the exchanged values are highlighted.

92	99	1	8	15	67	74	51	58	65
98	80	7	14	16	73	55	57	64	66
79	81	13	20	22	54	56	63	70	72
85	87	19	21	3	60	62	69	71	53
86	93	23	2	9	61	68	75	52	59
17	24	76	83	90	42	49	26	33	40
23	5	82	89	91	48	30	32	39	41
4	6	88	95	97	29	31	38	45	47
10	12	94	96	78	35	37	44	46	28
11	18	100	77	84	36	43	50	27	34

- Exchange the rightmost (**nCols-1**) in sub-grid **A** with the corresponding columns in sub-grid **D**. From the previous set that would be 2-1 which is 1. If (**nCols-1**) is 0, this step can be skipped. For example, the exchanged values are highlighted.

92	99	1	8	15	67	74	51	58	40
98	80	7	14	16	73	55	57	64	41
79	81	13	20	22	54	56	63	70	47
85	87	19	21	3	60	62	69	71	28
86	93	23	2	9	61	68	75	52	34
17	24	76	83	90	42	49	26	33	65
23	5	82	89	91	48	30	32	39	66
4	6	88	95	97	29	31	38	45	72
10	12	94	96	78	35	37	44	46	53
11	18	100	77	84	36	43	50	27	59

- Exchange the middle cell of the leftmost column of sub-grid **A** with the corresponding middle cell of sub-grid **D**. Exchange the center cell of the sub-grid **A** with the corresponding center cell of sub-grid **D**.

92	99	1	8	15	67	74	51	58	40
98	80	7	14	16	73	55	57	64	41
4	81	88	20	22	54	56	63	70	47
85	87	19	21	3	60	62	69	71	28
86	93	23	2	9	61	68	75	52	34
17	24	76	83	90	42	49	26	33	65
23	5	82	89	91	48	30	32	39	66
79	6	13	95	97	29	31	38	45	72
10	12	94	96	78	35	37	44	46	53
11	18	100	77	84	36	43	50	27	59

You may refer to the provided references for further explanation and some additional examples. All routines must be written efficiently. A point reduction will be applied for poor or especially inefficient solutions.

Some test files will be provided. Additionally, you may create your own. Refer to the example executions for output formatting. Make sure your program includes the appropriate documentation.

Example Executions:

Below is an example output for the provided board main:

```
ed-vm% ./brdMain

*** Error Testing ***********************************************************
 5 Declaration errors...

Error, invalid board size.
No board created.
Error, invalid board size.
No board created.
Error, invalid board size.
No board created.
Error, invalid board size.
No board created.
Error, invalid board size.
No board created.

*** Error Testing ***********************************************************
 4 invalid board locations...

Error, invalid board location.
Error, invalid board location.
Error, invalid board location.
Error, invalid board location.

Output badBrd3 (should be empty):


*** Board #0 ************************************************************
 3 x 3 -> All zero's


     _____ _____ _____
    |      |      |      |
    |  0   |  0   |  0   |
    |_____|_____|_____|
    |      |      |      |
    |  0   |  0   |  0   |
    |_____|_____|_____|
    |      |      |      |
    |  0   |  0   |  0   |
    |_____|_____|_____|


*** Board #1 ************************************************************
 11 x 11 -> numbered


     _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____
    |      |      |      |      |      |      |      |      |      |      |      |
    |  1   |  2   |  3   |  4   |  5   |  6   |  7   |  8   |  9   |  10  |  11  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |      |      |      |      |      |      |      |      |      |      |      |
    |  12  |  13  |  14  |  15  |  16  |  17  |  18  |  19  |  20  |  21  |  22  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |      |      |      |      |      |      |      |      |      |      |      |
    |  23  |  24  |  25  |  26  |  27  |  28  |  29  |  30  |  31  |  32  |  33  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |      |      |      |      |      |      |      |      |      |      |      |
    |  34  |  35  |  36  |  37  |  38  |  39  |  40  |  41  |  42  |  43  |  44  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |      |      |      |      |      |      |      |      |      |      |      |
    |  45  |  46  |  47  |  48  |  49  |  50  |  51  |  52  |  53  |  54  |  55  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |      |      |      |      |      |      |      |      |      |      |      |
    |  56  |  57  |  58  |  59  |  60  |  61  |  62  |  63  |  64  |  65  |  66  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |      |      |      |      |      |      |      |      |      |      |      |
    |  67  |  68  |  69  |  70  |  71  |  72  |  73  |  74  |  75  |  76  |  77  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |      |      |      |      |      |      |      |      |      |      |      |
    |  78  |  79  |  80  |  81  |  82  |  83  |  84  |  85  |  86  |  87  |  88  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |      |      |      |      |      |      |      |      |      |      |      |
    |  89  |  90  |  91  |  92  |  93  |  94  |  95  |  96  |  97  |  98  |  99  |
```

```
 |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
 |     |     |     |     |     |     |     |     |     |     |     |
 | 100 | 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 |
 |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
 |     |     |     |     |     |     |     |     |     |     |     |
 | 111 | 112 | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 |
 |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|


 *** Board #2 ************************************************************
  12 x 12 -> 1's in main diagonals, 0's otherwise.
  5 bad board location accesses...

 Error, invalid board location.
 Error, invalid board location.
 Error, invalid board location.
 Error, invalid board location.
 Error, invalid board location.

    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  1  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  1  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  1  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  1  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  0  |  1  |  0  |  0  |  0  |  0  |  0  |  0  |  1  |  0  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  0  |  0  |  1  |  0  |  0  |  0  |  0  |  1  |  0  |  0  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  0  |  0  |  0  |  1  |  0  |  0  |  1  |  0  |  0  |  0  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  0  |  0  |  0  |  0  |  1  |  1  |  0  |  0  |  0  |  0  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  0  |  0  |  0  |  0  |  1  |  1  |  0  |  0  |  0  |  0  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  0  |  0  |  0  |  1  |  0  |  0  |  1  |  0  |  0  |  0  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  0  |  0  |  1  |  0  |  0  |  0  |  0  |  1  |  0  |  0  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  0  |  1  |  0  |  0  |  0  |  0  |  0  |  0  |  1  |  0  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  0  |  1  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  1  |  0  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |     |
    |  1  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  0  |  1  |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|


    ************************************************************

    ed-vm%
```

Below is an example output for the provided main (magicSqr):

```
    ed-vm% ./magicSqr

    ************************************************************

    CS 202 - Magic Squares
      Title: Test, Order 3
      Magic Square, order: 3
      Magic Number: 15

        |_____|_____|_____|
        |     |     |     |
        |  8  |  1  |  6  |
        |_____|_____|_____|
        |     |     |     |
```

```
              |   3   |   5   |   7   |
              |_____|_____|_____|
              |       |       |       |
              |   4   |   9   |   2   |
              |_____|_____|_____|
```

Valid Magic Square.


```
***********************************************************
```

CS 202 - Magic Squares
  Title: Test, Order 4
  Magic Square, order: 4
  Magic Number: 34

```
      |_____|_____|_____|_____|
      |       |       |       |       |
      |  16   |   2   |   3   |  13   |
      |_____|_____|_____|_____|
      |       |       |       |       |
      |   5   |  11   |  10   |   8   |
      |_____|_____|_____|_____|
      |       |       |       |       |
      |   9   |   7   |   6   |  12   |
      |_____|_____|_____|_____|
      |       |       |       |       |
      |   4   |  14   |  15   |   1   |
      |_____|_____|_____|_____|
```

Valid Magic Square.


```
***********************************************************
```

CS 202 - Magic Squares
  Title: Test, Order 5
  Magic Square, order: 5
  Magic Number: 65

```
      |_____|_____|_____|_____|_____|
      |       |       |       |       |       |
      |  17   |  24   |   1   |   8   |  15   |
      |_____|_____|_____|_____|_____|
      |       |       |       |       |       |
      |  23   |   5   |   7   |  14   |  16   |
      |_____|_____|_____|_____|_____|
      |       |       |       |       |       |
      |   4   |   6   |  13   |  20   |  22   |
      |_____|_____|_____|_____|_____|
      |       |       |       |       |       |
      |  10   |  12   |  19   |  21   |   3   |
      |_____|_____|_____|_____|_____|
      |       |       |       |       |       |
      |  11   |  18   |  25   |   2   |   9   |
      |_____|_____|_____|_____|_____|
```

Valid Magic Square.


```
***********************************************************
```

CS 202 - Magic Squares
  Title: Test, Order 6
  Magic Square, order: 6
  Magic Number: 111

```
      |_____|_____|_____|_____|_____|_____|
      |       |       |       |       |       |       |
      |  35   |   1   |   6   |  26   |  19   |  24   |
      |_____|_____|_____|_____|_____|_____|
      |       |       |       |       |       |       |
      |   3   |  32   |   7   |  21   |  23   |  25   |
      |_____|_____|_____|_____|_____|_____|
      |       |       |       |       |       |       |
      |  31   |   9   |   2   |  22   |  27   |  20   |
      |_____|_____|_____|_____|_____|_____|
```

```
       _____ _____ _____ _____ _____ _____
      |     |     |     |     |     |     |
      |  8  | 28  | 33  | 17  | 10  | 15  |
      |_____|_____|_____|_____|_____|_____|
      |     |     |     |     |     |     |
      | 30  |  5  | 34  | 12  | 14  | 16  |
      |_____|_____|_____|_____|_____|_____|
      |     |     |     |     |     |     |
      |  4  | 36  | 29  | 13  | 18  | 11  |
      |_____|_____|_____|_____|_____|_____|
```

Valid Magic Square.

```
************************************************************
```

CS 202 - Magic Squares
  Title: Test, Order 8
  Magic Square, order: 8
  Magic Number: 260

```
       _____ _____ _____ _____ _____ _____ _____ _____
      |     |     |     |     |     |     |     |     |
      | 64  |  2  |  3  | 61  | 60  |  6  |  7  | 57  |
      |_____|_____|_____|_____|_____|_____|_____|_____|
      |     |     |     |     |     |     |     |     |
      |  9  | 55  | 54  | 12  | 13  | 51  | 50  | 16  |
      |_____|_____|_____|_____|_____|_____|_____|_____|
      |     |     |     |     |     |     |     |     |
      | 17  | 47  | 46  | 20  | 21  | 43  | 42  | 24  |
      |_____|_____|_____|_____|_____|_____|_____|_____|
      |     |     |     |     |     |     |     |     |
      | 40  | 26  | 27  | 37  | 36  | 30  | 31  | 33  |
      |_____|_____|_____|_____|_____|_____|_____|_____|
      |     |     |     |     |     |     |     |     |
      | 32  | 34  | 35  | 29  | 28  | 38  | 39  | 25  |
      |_____|_____|_____|_____|_____|_____|_____|_____|
      |     |     |     |     |     |     |     |     |
      | 41  | 23  | 22  | 44  | 45  | 19  | 18  | 48  |
      |_____|_____|_____|_____|_____|_____|_____|_____|
      |     |     |     |     |     |     |     |     |
      | 49  | 15  | 14  | 52  | 53  | 11  | 10  | 56  |
      |_____|_____|_____|_____|_____|_____|_____|_____|
      |     |     |     |     |     |     |     |     |
      |  8  | 58  | 59  |  5  |  4  | 62  | 63  |  1  |
      |_____|_____|_____|_____|_____|_____|_____|_____|
```

Valid Magic Square.

```
************************************************************
```

CS 202 - Magic Squares
  Title: Test, Order 9
  Magic Square, order: 9
  Magic Number: 369

```
       _____ _____ _____ _____ _____ _____ _____ _____ _____
      |     |     |     |     |     |     |     |     |     |
      | 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

Valid Magic Square.

****

CS 202 - Magic Squares
  Title: Test, Order 10
  Magic Square, order: 10
  Magic Number: 505

92	99	1	8	15	67	74	51	58	40
98	80	7	14	16	73	55	57	64	41
4	81	88	20	22	54	56	63	70	47
85	87	19	21	3	60	62	69	71	28
86	93	25	2	9	61	68	75	52	34
17	24	76	83	90	42	49	26	33	65
23	5	82	89	91	48	30	32	39	66
79	6	13	95	97	29	31	38	45	72
10	12	94	96	78	35	37	44	46	53
11	18	100	77	84	36	43	50	27	59

Valid Magic Square.

****

CS 202 - Magic Squares
  Title: Test, Order 12
  Magic Square, order: 12
  Magic Number: 870

144	2	3	141	140	6	7	137	136	10	11	133
13	131	130	16	17	127	126	20	21	123	122	24
25	119	118	28	29	115	114	32	33	111	110	36
108	38	39	105	104	42	43	101	100	46	47	97
96	50	51	93	92	54	55	89	88	58	59	85

61	83	82	64	65	79	78	68	69	75	74	72
73	71	70	76	77	67	66	80	81	63	62	84
60	86	87	57	56	90	91	53	52	94	95	49
48	98	99	45	44	102	103	41	40	106	107	37
109	35	34	112	113	31	30	116	117	27	26	120
121	23	22	124	125	19	18	128	129	15	14	132
12	134	135	9	8	138	139	5	4	142	143	1

Valid Magic Square.

****

CS 202 - Magic Squares
  Title: Test, Order 13
  Magic Square, order: 13
  Magic Number: 1105

93	108	123	138	153	168	1	16	31	46	61	76	91
107	122	137	152	167	13	15	30	45	60	75	90	92
121	136	151	166	12	14	29	44	59	74	89	104	106
135	150	165	11	26	28	43	58	73	88	103	105	120
149	164	10	25	27	42	57	72	87	102	117	119	134
163	9	24	39	41	56	71	86	101	116	118	133	148
8	23	38	40	55	70	85	100	115	130	132	147	162
22	37	52	54	69	84	99	114	129	131	146	161	7
36	51	53	68	83	98	113	128	143	145	160	6	21
50	65	67	82	97	112	127	142	144	159	5	20	35
64	66	81	96	111	126	141	156	158	4	19	34	49
78	80	95	110	125	140	155	157	3	18	33	48	63
79	94	109	124	139	154	169	2	17	32	47	62	77

Valid Magic Square.

```
************************************************************

CS 202 - Magic Squares
  Title: Test, Order 14
  Magic Square, order: 14
  Magic Number: 1379
```

177	186	195	1	10	19	28	128	137	146	99	108	68	77
185	194	154	9	18	27	29	136	145	105	107	116	76	78
193	153	155	17	26	35	37	144	104	106	115	124	84	86
5	161	163	172	34	36	45	103	112	114	123	132	85	94
160	162	171	33	42	44	4	111	113	122	131	140	93	53
168	170	179	41	43	3	12	119	121	130	139	141	52	61
169	178	187	49	2	11	20	120	129	138	147	100	60	69
30	39	48	148	157	166	175	79	88	97	50	59	117	126
38	47	7	156	165	174	176	87	96	56	58	67	125	127
46	6	8	164	173	182	184	95	55	57	66	75	133	135
152	14	16	25	181	183	192	54	63	65	74	83	134	143
13	15	24	180	189	191	151	62	64	73	82	91	142	102
21	23	32	188	190	150	159	70	72	81	90	92	101	110
22	31	40	196	149	158	167	71	80	89	98	51	109	118

Valid Magic Square.

```
************************************************************
Read Testing, 3x3...

Enter 3x3 Magic Square.

Enter Title: Test - Good

Enter Magic Square Values (3x3)
2 7 6
9 5 1
4 3 8

************************************************************

CS 202 - Magic Squares
  Title: Test - Good
  Magic Square, order: 3
  Magic Number: 15
```

2	7	6
9	5	1

```
 |_____|_____|_____|
 |     |     |     |
 |  4  |  3  |  8  |
 |_____|_____|_____|
```

Valid Magic Square.


```
*************************************************************
Enter 3x3 Magic Square.

Enter Title: Enter Title: Test - Bad

Enter Magic Square Values (3x3)
7 7 1
7 7 1
1 1 1

*************************************************************
```

CS 202 - Magic Squares
  Title: Test - Bad
  Magic Square, order: 3
  Magic Number: 15

```
     _____ _____ _____
    |     |     |     |
    |  7  |  7  |  1  |
    |_____|_____|_____|
    |     |     |     |
    |  7  |  7  |  1  |
    |_____|_____|_____|
    |     |     |     |
    |  1  |  1  |  1  |
    |_____|_____|_____|
```


Not a Magic Square Solution.


```
*************************************************************
Read Testing, 11x11...

Enter 11x11 Magic Square.

Enter Title: Enter Title: Test - 11x11

Enter Magic Square Values (11x11)
 68  81  94 107 120   1  14  27  40  53  66
 80  93 106 119  11  13  26  39  52  65  67
 92 105 118  10  12  25  38  51  64  77  79
104 117   9  22  24  37  50  63  76  78  91
116   8  21  23  36  49  62  75  88  90 103
  7  20  33  35  48  61  74  87  89 102 115
 19  32  34  47  60  73  86  99 101 114   6
 31  44  46  59  72  85  98 100 113   5  18
 43  45  58  71  84  97 110 112   4  17  30
 55  57  70  83  96 109 111   3  16  29  42
 56  69  82  95 108 121   2  15  28  41  54

*************************************************************
```

CS 202 - Magic Squares
  Title: Test - 11x11
  Magic Square, order: 11
  Magic Number: 671

```
     _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____
    |     |     |     |     |     |     |     |     |     |     |     |
    |  68 |  81 |  94 | 107 | 120 |  1  |  14 |  27 |  40 |  53 |  66 |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |
    |  80 |  93 | 106 | 119 |  11 |  13 |  26 |  39 |  52 |  65 |  67 |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
    |     |     |     |     |     |     |     |     |     |     |     |
    |  92 | 105 | 118 |  10 |  12 |  25 |  38 |  51 |  64 |  77 |  79 |
    |_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
```

```
| 104 | 117 |   9 |  22 |  24 |  37 |  50 |  63 |  76 |  78 |  91 |
| 116 |   8 |  21 |  23 |  36 |  49 |  62 |  75 |  88 |  90 | 103 |
|   7 |  20 |  33 |  35 |  48 |  61 |  74 |  87 |  89 | 102 | 115 |
|  19 |  32 |  34 |  47 |  60 |  73 |  86 |  99 | 101 | 114 |   6 |
|  31 |  44 |  46 |  59 |  72 |  85 |  98 | 100 | 113 |   5 |  18 |
|  43 |  45 |  58 |  71 |  84 |  97 | 110 | 112 |   4 |  17 |  30 |
|  55 |  57 |  70 |  83 |  96 | 109 | 111 |   3 |  16 |  29 |  42 |
|  56 |  69 |  82 |  95 | 108 | 121 |   2 |  15 |  28 |  41 |  54 |
```

Valid Magic Square.


```
************************************************************
Enter 11x11 Magic Square.

Enter Title: Enter Title: Test - 11x11 bad

Enter Magic Square Values (11x11)
6      117    46     107    36     97     26     87     16     77     6
7      57     118    47     108    37     98     27     88     17     67
68     8      58     119    48     109    38     99     28     78     18
19     69     9      59     120    49     110    39     89     29     79
80     20     70     10     60     121    50     100    40     90     30
31     81     21     71     11     61     111    51     101    41     91
92     32     82     22     72     1      62     112    52     102    42
43     93     33     83     12     73     2      63     113    53     103
104    44     94     23     84     13     74     3      64     114    54
55     105    34     95     24     85     14     75     4      65     115
116    45     106    35     96     25     86     15     76     5      66

************************************************************
```

CS 202 - Magic Squares
  Title: Test - 11x11 bad
  Magic Square, order: 11
  Magic Number: 671

```
|   6 | 117 |  46 | 107 |  36 |  97 |  26 |  87 |  16 |  77 |   6 |
|   7 |  57 | 118 |  47 | 108 |  37 |  98 |  27 |  88 |  17 |  67 |
|  68 |   8 |  58 | 119 |  48 | 109 |  38 |  99 |  28 |  78 |  18 |
|  19 |  69 |   9 |  59 | 120 |  49 | 110 |  39 |  89 |  29 |  79 |
|  80 |  20 |  70 |  10 |  60 | 121 |  50 | 100 |  40 |  90 |  30 |
|  31 |  81 |  21 |  71 |  11 |  61 | 111 |  51 | 101 |  41 |  91 |
|  92 |  32 |  82 |  22 |  72 |   1 |  62 | 112 |  52 | 102 |  42 |
|  43 |  93 |  33 |  83 |  12 |  73 |   2 |  63 | 113 |  53 | 103 |
```

```
|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |     |     |     |
| 104 | 44  | 94  | 23  | 84  | 13  | 74  |  3  | 64  | 114 | 54  |
|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |     |     |     |
| 55  | 105 | 34  | 95  | 24  | 85  | 14  | 75  |  4  | 65  | 115 |
|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |     |     |     |
| 116 | 45  | 106 | 35  | 96  | 25  | 86  | 15  | 76  |  5  | 66  |
|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|
```

Not a Magic Square Solution.


```
************************************************************
Read Testing, 6x6...

Enter 6x6 Magic Square.

Enter Title: Enter Title: Test 6x6 - good

Enter Magic Square Values (6x6)
6 32 3 34 35 1
7 11 27 28 8 30
19 14 16 15 23 24
18 20 22 21 17 13
25 29 10 9 26 12
36 5 33 4 2 31

************************************************************
```

CS 202 - Magic Squares
  Title: Test 6x6 - good
  Magic Square, order: 6
  Magic Number: 111

```
 _____ _____ _____ _____ _____ _____
|     |     |     |     |     |     |
|  6  | 32  |  3  | 34  | 35  |  1  |
|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |
|  7  | 11  | 27  | 28  |  8  | 30  |
|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |
| 19  | 14  | 16  | 15  | 23  | 24  |
|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |
| 18  | 20  | 22  | 21  | 17  | 13  |
|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |
| 25  | 29  | 10  |  9  | 26  | 12  |
|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |
| 36  |  5  | 33  |  4  |  2  | 31  |
|_____|_____|_____|_____|_____|_____|
```

Valid Magic Square.


```
************************************************************
Read Testing, 8x8...

Enter 8x8 Magic Square.

Enter Title: Enter Title: Test 8x8 - good

Enter Magic Square Values (8x8)
8 58 59 5 4 62 63 1
49 15 14 52 53 11 10 56
41 23 22 44 45 19 18 48
32 34 35 29 28 38 39 25
40 26 27 37 36 30 31 33
17 47 46 20 21 43 42 24
9 55 54 12 13 51 50 16
64 2 3 61 60 6 7 57
```

```
************************************************************
CS 202 - Magic Squares
  Title: Test 8x8 - good
  Magic Square, order: 8
  Magic Number: 260

 _____ _____ _____ _____ _____ _____ _____ _____
|     |     |     |     |     |     |     |     |
|   8 |  58 |  59 |   5 |   4 |  62 |  63 |   1 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  49 |  15 |  14 |  52 |  53 |  11 |  10 |  56 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  41 |  23 |  22 |  44 |  45 |  19 |  18 |  48 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  32 |  34 |  35 |  29 |  28 |  38 |  39 |  25 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  40 |  26 |  27 |  37 |  36 |  30 |  31 |  33 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  17 |  47 |  46 |  20 |  21 |  43 |  42 |  24 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|   9 |  55 |  54 |  12 |  13 |  51 |  50 |  16 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  64 |   2 |   3 |  61 |  60 |   6 |   7 |  57 |
|_____|_____|_____|_____|_____|_____|_____|_____|
```

Valid Magic Square.

```
************************************************************
Enter 8x8 Magic Square.

Enter Title: Enter Title: Test 8x8 - bad

Enter Magic Square Values (8x8)
7 58 59 5 4 62 63 1
49 15 14 52 53 11 10 56
41 23 22 44 45 19 18 48
32 34 35 29 28 38 39 25
40 26 27 37 36 30 31 33
17 47 46 20 21 43 42 24
9 55 54 12 13 51 50 16
64 2 3 61 60 6 8 57

************************************************************
```

CS 202 - Magic Squares
  Title: Test 8x8 - bad
  Magic Square, order: 8
  Magic Number: 260

```
 _____ _____ _____ _____ _____ _____ _____ _____
|     |     |     |     |     |     |     |     |
|   7 |  58 |  59 |   5 |   4 |  62 |  63 |   1 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  49 |  15 |  14 |  52 |  53 |  11 |  10 |  56 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  41 |  23 |  22 |  44 |  45 |  19 |  18 |  48 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  32 |  34 |  35 |  29 |  28 |  38 |  39 |  25 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  40 |  26 |  27 |  37 |  36 |  30 |  31 |  33 |
|_____|_____|_____|_____|_____|_____|_____|_____|
|     |     |     |     |     |     |     |     |
|  17 |  47 |  46 |  20 |  21 |  43 |  42 |  24 |
|_____|_____|_____|_____|_____|_____|_____|_____|
```

9	55	54	12	13	51	50	16
64	2	3	61	60	6	8	57

Not a Magic Square Solution.

ed-vm%