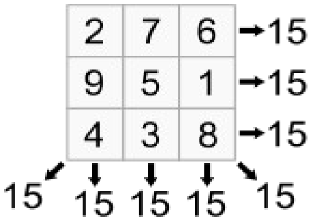
CS 218 – Assignment #3

Purpose: Become familiar with the MIPS Instruction Set, and the MIPS function calling convention, and indexing for multiple dimension arrays.
Points: 100

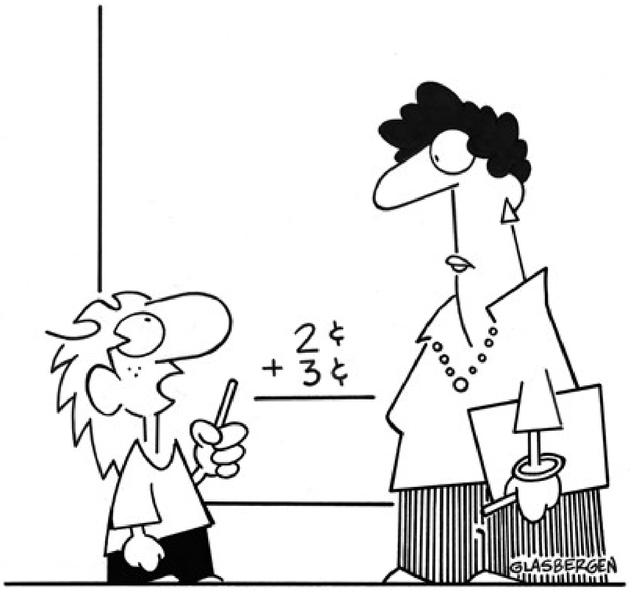
Assignment:

Write a simple assembly language function to check if a two-dimensional array is a *magic square*[1]. The provided main calls the following functions as follows:

- Write a void function, ***chkMagicSqr(arr, order)***, that will check an (***n*** by ***n***) two-dimensional array to see if it is a magic square. In recreational mathematics, a magic square of order ***n*** is an arrangement of n^2 numbers, usually integers, in a square stored as a two-dimensional array, such that the ***n*** numbers in all rows, all columns, and both diagonals sum to the same value. A normal magic square contains the integers from 1 to n^2. This function must call the ***prtMsg()*** function to display the sums (each row, each column, and each diagonal).
- Write a void function, ***prtMsg(str, num, sum)***, to display the row, column, or diagonal message, the row/col/diagonal number, and sum. Refer to the example execution for output formating.
- Write a void function, ***prtSquare(arr, order)***, to display an (***n*** by ***n***) two-dimensional matrix. The numbers should be printed in a two-dimensional format (see example output). All numbers must be right justified (i.e., lined up on right side). This can is done by printing spaces based on the range of the number.



	2	7	6	→15
	9	5	1	→15
	4	3	8	→15
15	15	15	15	15



"If I'll be paying off college loans someday, we should probably start using bigger numbers."

Array Implementation:

At the machine level, multi-dimension arrays are implemented as a large single dimension array. The formula for calculating two-dimensional array indexing is:

```
addr[row,col] = baseAddress + (row * colDimension + col) * elementSize
```

You must use the formula to access matrix elements. **No score** will be provided for submissions that do not use this formula. The `colDimension` is the number of columns the array was originally created with. The `elementSize` would be 4 for word, 2 for halfwords and 1 for bytes.

[1] For more information, refer to: https://en.wikipedia.org/wiki/Magic_square

Submission:

- All source files must assemble and execute with QtSpim/SPIM MIPS simulator.
- Submit source file
  - Submit a copy of the program source file via the on-line submission
- 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 (at a maximum rate of 5 submissions per hour).
- 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 source files must include your name, section number, assignment, NSHE number, and program description. The required format is as follows:

```
#  Name: <your name>
#  NSHE ID: <your id>
#  Section: <section>
#  Assignment: <assignment number>
#  Description: <short description of program goes here>
```

Failure to include your name in this format will result in a reduction of points.

Scoring Rubric

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

Criteria	Weight	Summary
Assemble	-	Failure to assemble will result in a score of 0.
Program Header	3%	Must include header block in the required format (see above).
General Comments	7%	Must include an appropriate level of program documentation.
Program Functionality (and on-time)	90%	Program must meet the functional requirements as outlined in the assignment. Must be submitted on time for full score.

Example Output:

The following is the example output for the first data set:

```
MIPS Assignment #3
Program to check a Magic Square.

--------------------------------------------
Possible Magic Square #1

    2     7     6
    9     5     1
    4     3     8

   Row  #0   Sum: 15
   Row  #1   Sum: 15
   Row  #2   Sum: 15
   Col  #0   Sum: 15
   Col  #1   Sum: 15
   Col  #2   Sum: 15
   Diag #1   Sum: 15
   Diag #2   Sum: 15

IS a Magic Square.

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

   [ … output truncated for space … ]

--------------------------------------------
Possible Magic Square #3

   16     3     2    13
    5    10    11     8
    9     5     7    12
    4    15    14     1

   Row  #0   Sum: 34
   Row  #1   Sum: 34
   Row  #2   Sum: 33
   Row  #3   Sum: 34
   Col  #0   Sum: 34
   Col  #1   Sum: 33
   Col  #2   Sum: 34
   Col  #3   Sum: 34
   Diag #1   Sum: 34
   Diag #2   Sum: 33

NOT a Magic Square.

   [ … output truncated for space … ]
```

Note, not all data sets not shown.