Assignment #12 – Multi-dimensional Arrays and Recursion **CS 218 Spring 2021**

Purpose: This MIPS program will explore the use of recursive functions and multi-dimensional arrays.

A template file is provided that outlines the required functions and includes code for the main function.

Note that all arrays in this program will store single precision float values in row major order.

MIPS and local variables. We will need to allocate a local variable in this program to hold our submatrice while recursively calling the determinant function. Use the stack to hold the values for the new array as needed. You can use the stack pointer register to refer to this local.

Required Functions:

Print Matrix
Argument 1: array address
Argument 2: integer number of rows
Argument 3: integer number of columns
Returns: Nothing
Prints out the given 2D array. All values for a row should be printed on one line together with spaces separating the values.

Generate Square Matrix

Argument 1: array address
Argument 2: integer array size
Returns: Nothing
Set each value of the given array to the following:

```
array[i][j] = i + (j x 0.5)
    row  col
```

Example:

Size 3 Square Array:

0.0	0.5	1.0
1.0	1.5	2.0
2.0	2.5	3.0

Set Element

Argument 1: array address
Argument 2: integer number of columns
Argument 3: integer row index
Argument 4: integer column index
Argument 5: floating point value to insert into the array
Returns nothing

This function will insert the new element into the 2D array at the given index values.

Calculate Determinant

Argument 1: array address
Argument 2: integer array size
Returns the determinant of the given matrix as a float

This recursive function will calculate a determinant as described below:

Determinant of a 1x1 Matrix:

Returns the value stored in the matrix.

Determinant of a NxN Matrix:

A	B	C	…	Z
a1	b1	c1	…	z1
a2	b2	c2	…	z2
…	…	…	…	…
aX	bX	cX	…	zX

Multiply each value in the first row of the matrix (A, B, C, etc above) by the determinant of the (N-1)x(N-1) matrix consisting of all elements not in that value's row or column.

```
A  B  C        +  A x determinant( E F )
                                   H I
D  E  F
               -  B x determinant( D F )
G  H  I                            G I

               +  C x determinant( D E )
                                   G H
```

Alternate between adding and subtracting each value.

Store the submatrix for each recursive call in a local variable.

Example 3x3 Determinant Calculation

```
1   4   2

1   3   2

0   1   4

1 x det( 3 2 ) - 4 x det( 1 2 ) + 2 x det( 1 3 )
         1 4               0 4              0 1

det ( 3 2 ) = 3 x det(4) – 2 x det(1)
      1 4
             = 3 x 4 – 2 x 1
            = 12 – 2
            = 10

det ( 1 2 ) = 1 x det(4) – 2 x det(0)
      0 4
             = 1 x 4 – 2 x 0
              = 4

det ( 1 3 ) = 1 x det(1) – 3 x det(0)
      0 1
             = 1 x 1 – 3 x 0
            = 1

1 x 10 – 4 x 4 + 2 x 1
10 – 16 + 2
-4
```

When creating your submatrix for the recursive call, use a local variable to store it. Copy all the values from the original array minus the first row and the column that the top row value is in.

Submission

Once completed, upload the MIPS assembly source code file (.asm) to the class website.

Example Execution 1x1

```
Select a matrix size (1-6): 1

Generating a 1x1 Matrix

0.00000000

Would you like to change a value (y/n): y

Column: 0

Row: 0

Change to: 4.5

4.50000000

Would you like to change a value (y/n): n

Determinant: 4.50000000
```

Example Execution 2x2

```
Select a matrix size (1-6): 2

Generating a 2x2 Matrix

0.00000000 0.50000000
1.00000000 1.50000000

Would you like to change a value (y/n): y

Column: 0

Row: 0

Change to: 4.56

4.55999994 0.50000000
1.00000000 1.50000000

Would you like to change a value (y/n): n

Determinant: 6.34000015
```

Example Execution 6x6

```
Select a matrix size (1-6): 6


Generating a 6x6 Matrix


0.00000000 0.50000000 1.00000000 1.50000000 2.00000000 2.50000000
1.00000000 1.50000000 2.00000000 2.50000000 3.00000000 3.50000000
2.00000000 2.50000000 3.00000000 3.50000000 4.00000000 4.50000000
3.00000000 3.50000000 4.00000000 4.50000000 5.00000000 5.50000000
4.00000000 4.50000000 5.00000000 5.50000000 6.00000000 6.50000000
5.00000000 5.50000000 6.00000000 6.50000000 7.00000000 7.50000000

Would you like to change a value (y/n): n

Determinant: 0.00000000
```

Example Execution Error Checking

```
Select a matrix size (1-6): 0

Invalid size.

Select a matrix size (1-6): 10

Invalid size.

Select a matrix size (1-6): 3

Generating a 3x3 Matrix

0.00000000 0.50000000 1.00000000
1.00000000 1.50000000 2.00000000
2.00000000 2.50000000 3.00000000

Would you like to change a value (y/n): y

Column: -1

Invalid column index

Column: 2

Row: 5

Invalid row index

Row: 3
```

```
Invalid row index

Row: 2

Change to: 0

0.00000000 0.50000000 1.00000000
1.00000000 1.50000000 2.00000000
2.00000000 2.50000000 0.00000000

Would you like to change a value (y/n): d

Would you like to change a value (y/n): a

Would you like to change a value (y/n): f

Would you like to change a value (y/n): d

Would you like to change a value (y/n): n

Determinant: 1.50000000
```