MATLAB Programming NCTU Spring 2017 LAB#2 2017/10/2

Note: No loop unless noted otherwise. Use only functions mentioned in the class so far, unless noted otherwise.

Part 1: Assigning Values to the Diagonal Elements of a Square Matrix

Make `nxn` diagonal matrices (`n` given in a variable) whose diagonal values are 1 to `n`. Hint: First determine the linear indices of the diagonal elements, and then assign `1:n` to them. Example for `n`=5:

```
1  0  0  0  0
0  2  0  0  0
0  0  3  0  0
0  0  0  4  0
0  0  0  0  5
```

Part 2: Draw a filled circle

1. Make a square matrix `A` of size `nxn`. Make n an odd number.
2. Compute the "distances" of all the elements to the center element. Store these in a "distance matrix" `D`, also of size `nxn`. For this purpose, create two arrays representing the x and y "coordinates" of all the elements; these two arrays also have the size `nxn`. You can use `repmat` to create these two arrays conveniently.
3. For a given radius `r` (`r` > 0; `r` can be a floating-point number), set `A(ii,jj)` to 1 if `D(ii,jj)<r`, and 0 otherwise. Example below for `n`=7 and `r`=2.5:

```
0    0    0    0    0    0    0
0    0    1    1    1    0    0
0    1    1    1    1    1    0
0    1    1    1    1    1    0
0    1    1    1    1    1    0
0    0    1    1    1    0    0
0    0    0    0    0    0    0
```

4. [Optional] Try to utilize `fprintf` to print a more compact version, like the example below. You can use one level of loop. Better yet, you can apply `repmat` to the format string of `fprintf` and print out the whole thing without using any loop.

```
0000000
0011100
0111110
0111110
0111110
0011100
0000000
```

Part 3: Pascal Triangle

Note: You can use **one level** of loop.

For a given integer `n`>0, print out the Pascal triangle with `n` levels. Example for `n`=5:

```
1
1  1
1  2  1
1  3  3  1
1  4  6  4  1
```

Store the values of each level in a vector, which can be computed and printed in one statement.