CS1112 Exercise 9 Name: _____ NetID: _____

You have until *Sunday, April 12, at 9:00 PM EDT* to submit problems 2 and 3 to MATLAB Grader.

1 Virtual discussion

By discussing with your lab partners in your breakout room, determine which two of you were the farthest apart at some instant in time since Cornell announced it was going virtual. Be ready to let the consultant/TA know what you figured out and how!

- Date and time (EDT):
- Person 1 name and location:
- Person 2 name and location:
- Approximate (geodesic) distance apart (for reference, Earth's circumference is 40,000 km):

2 Random walk

> This question requires that you *design* a solution, subject to a few constraints in the interest of time. Take some time to do the planning—think about what values you need to keep track of and choose "appropriately-shaped" variables to store them.

A random walk that starts from the center of a 21×21 grid ends when a boundary is reached. On average which "square" or grid point is visited most often? Function `RandomWalk2D` (discussed in lecture) is shown below for your reference; you should call it from your solution, but do not modify its implementation.

```
function [x, y] = RandomWalk2D(N)
% Simulate a 2D random walk in an (2N+1)-by-(2N+1) grid.
% N is a positive integer.
% Walk starts from the middle and continues until the an edge, abs(N),
% is reached.
% x and y are row vectors with the property that (x(k),y(k)) is the
% location of the token after k hops, k=1:length(x).

% Initializations...
k=0;  xc=0;  yc=0;

% In general, (xc,yc) is the location after k hops.
while abs(xc)<N && abs(yc)< N
    % Standing at (xc,yc), randomly select a step
    r= rand(1);
    if r < .25
        yc= yc + 1;  % north
    elseif r < .5
        xc= xc + 1;  % east
    elseif r < .75
        yc= yc -1;   % south
    else
        xc= xc -1;   % west
    end
    % Record location...
    k= k + 1;  x(k)= xc;  y(k)= yc;
end
```

Once you've designed your solution, you should submit it to MATLAB Grader. Grader imposes some constraints on your program design, so your solution should take the form of a function, `mostVisited()`, that calls `RandomWalk2D()` to simulate a single random walk (you should not provide this function yourself).

3 Two-dimensional interpolation

When you enlarge an image, you are actually adding data points among the existing data (pixels). How do you get the additional data points? One way is to interpolate from the neighboring points—take the average value. First, consider a simple case of one-dimensional interpolation, we add a data point between neighboring pairs of existing data points by taking the simple average. For example,

`2.0 1.0 1.0 2.0` becomes `2.0 1.5 1.0 1.0 1.0 1.5 2.0`.

In 2-d interpolation, work with one dimension at a time. For example, given a matrix

```
2.0  1.0  1.0  2.0
6.0  5.0  4.0  3.0
5.0  5.0  5.0  4.0
```

First we can add a column between two neighboring columns, so the matrix becomes 3×7:

```
2.0  1.5  1.0  1.0  1.0  1.5  2.0
6.0  5.5  5.0  4.5  4.0  3.5  3.0
5.0  5.0  5.0  5.0  5.0  4.5  4.0
```

Then add a row between neighboring rows, so the final matrix will be 5×7:

```
2.0  1.5  1.0  1.0  1.0  1.5  2.0
4.0  3.5  3.0  2.8  2.5  2.5  2.5
6.0  5.5  5.0  4.5  4.0  3.5  3.0
5.5  5.2  5.0  4.8  4.5  4.0  3.5
5.0  5.0  5.0  5.0  5.0  4.5  4.0
```

Write two functions for doing 2-d interpolation as discussed above: (a) `interpolate2d_nv` uses non-vectorized code; (b) `interpolate2d_v` uses vectorized code (work with whole columns one at a time; then whole rows). Do not use built-in functions `mean`, `sum`, and `linspace`.

```
function newM = interpolate2d_nv(M)
% Perform 2-d interpolation on the real-valued data in nr-by-nc matrix M
%   where nr>1 and nc>1.
% The interpolated data are added between existing data points so newM is
%   (2*nr-1)-by-(2*nc-1).  Use the simple average as the interpolated  value.
% Do NOT use built-in functions mean, sum, and linspace.
% NON-VECTORIZED implementation.
```

The vectorized version has the name `interpolate2d_v` but the same specification as above other than using vectorized code. In addition to submitting your code on MATLAB Grader, ask a member of the course staff to look over your vectorized solution to ensure that you are using vectorized code effectively.