Due: Wednesday, February 21 at 4 PM

Create a SCRIPT FILE, called `hw4Ober.m` (where "Ober" is replaced by the first 4 letters of your last name) of the commands in order to answer the problems using MATLAB.

- This SCRIPT FILE should be formatted in such a way to be "publishable" as a webpage. Your script and function files should appear on the G-drive in the folder `www\loy1234567\ma302\html` where "1234567" is your student id number, as in the previous assignments.
- See the template file `hw4Ober.m` found on the H-drive.
  - The first 3 commands in your file should be: `clear, format, clc` (after the comments that have your name, etc. as in the template file).
  - At the bottom of your webpage there should be a link to THE SCRIPT FILE (m-file) that created the webpage AND a link to the URL of the published webpage.
  - At the very bottom of the webpage there should be the date/time the page was published.

  Note that the template file has these already formatted; you just need to change the appropriate names/filepaths/URLs for your assignment.
- The various problems should be signified with comments within the SCRIPT file (so you can easily go to the command for #3, etc.) and should be sections on the webpage. Each problem NUMBER should be its own section with labeled header. You should NOT have separate sections headers for problem 1a, problem 1b, etc. but you may want them to be their own section so they run and publish appropriately.
- You will be graded on accuracy, readability and your programming ability. For instance, if there is one command to do the task but you use 3, you may lose points. Also, you will lose points (up to 5 points!) for not following the directions. You may also lose points if extraneous output is in the command window. You shouldn't suppress the output of the calculations, but you should suppress the output for any intermediate steps (like variable assignments). All graphs should have titles that are either specified within the problem or signify the problem such as "Problem 1b." Axes should be labeled, and the domains should be appropriately defined to see the curve is not jagged unless it is supposed to be.
- You should do your own testing to make sure they run accurately and as specified. For the the script files and function files you must have links to the files within your `hw4Ober.m` file and you must upload them to Moodle.
- Anywhere the problem states something like "appear as text on the webpage" or "state your answer in the text," you should format the comments in your SCRIPT file appropriately so they appear nicely when it is published to a webpage. Any work done on paper to obtain those answers should be turned in by the due date/time. Answers should be clear, preferably in complete sentences.
- Functions should have help lines following the example syntax, including a useful H1 line (see examples on H-drive). Your name should appear as comments within the function, but it should not appear within the help lines.
- Script files should have your name appear as a comment as the first line, and the first command of the script file should be `clc`.

For the following problems you will be asked to create functions and script files. **For all of them, you should replace "Ober" in the filename to the first four letters of your last name so everyone's files are distinct.**

1. Create a function called `isZOber` that will take one input and output a **logical** true or false (value of either 0 or 1 specifying false or true, respectively). If the input is an integer (NOT TALKING ABOUT DATA TYPE INTEGER - meaning it is not π, or 3/2, etc.), then it will return a logical `true`, otherwise it will return a logical `false`. Note that this should still work if the input is a vector or matrix, and would return a vector or matrix of true/false based on whether each entry/component is an integer or not. Within the homework file, run tests of the function and create a link to the file. Also upload the file to Moodle. This function should be used on any subsequent problem that needs to check if a variable or input is an integer.

2. Consider the function $f(x) = \dfrac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}}$.
   (a) Create a function called `npdfOber.m` that takes as input x, μ, σ (your choice on the variable names within your function) and computes $y = f(x)$ making sure that the calculations can be done if x is a number, vector or matrix (just as the `sin` function works). The only error checking that will be done is that σ is positive; if not, an appropriate error message should be displayed using the `error` command. Your homework file should have a link to the function file for this part of the problem, and upload the function file to Moodle.
   (b) For $\mu = 0$ and $\sigma = 1$, use your function to graph $y = f(x)$ for $x \in [-5, 5]$.
   (c) Use your function to graph the following two functions on the same figure. The first should be of $y = f(x)$ for $\mu = 149.97$, $\sigma = 8.49$, $x \in [130, 170]$. The second should be for $y = f(x)$ for $\mu = 152.57$ and $\sigma = 9.02$, $x \in [130, 170]$. Make sure they are different colors. Have a legend for this, with "Verbal Reasoning" for the first plot and "Quantitative Reasoning" for the second plot. The overall title should be "GRE Test Scores July 1, 2013 to June 30, 2016."

3. Create a SCRIPT FILE called `pbOber` that will use the `input` command to take ask for the user, "How many Power Ball tickets?" (n). Check that n is a number (rather than a vector or matrix) and a positive integer and if not, keep asking the user "Try again. How many Power Ball tickets?" until it is correct. Then using `input` again, ask, "Power Play option? (y/n)" (PP). Likewise, if the answer to the "Power Play" option is not "Y", "y", "N", or "n" keep asking the user, "Try again. Power Play option? (y/n)" You will use the `switch/case` command for the answer to the "Power Play" question. If PP is "Y" or "y", each Powerball ticket costs \$3. If PP is "N" or "n", each Powerball ticket costs \$2. The file then displays a matrix with n rows in which each row is a Powerball ticket (as specified in Assignment #2* but can now be done in an easier way). The amount owed for the Powerball tickets is also displayed. You do not need to get fancy with displaying the tickets and price; we will work on that later.
   (a) Within the homework file, generate `randi(10)` Power Ball tickets and your choice as to whether or not the Power Play option is selected.
   (b) Create a link to the function in the homework file and upload it to Moodle.

*The first five entries of each ticket are the "white balls" and the last entry of each ticket is the "powerball" for the drawing. For each ticket, the first five numbers should be random integers from 1 to 69 **with no repeats** and not necessarily in order and the last number should be a random integer from 1 to 26.

4. Create a function `circleOber.m` that takes as inputs a positive number r, a 2D point (vector) C, and an optional positive integer n. Error checking should be done on r, C (and n if input) and appropriate messages displayed using the `error` command. This function will output vectors x and y of size n (with default value of $n = 100$) that are the parametric equations for a circle with radius r with center C. Have the output be similar to `size` and `quadratic4` (found on the H-drive) in that both are output regardless of how the user runs the function.
   (a) Use your function to plot the unit circle and another circle with radius 3 and center $(-2, 4)$.
   (b) Create another graph where you plot the unit circle but with no input for n, $n = 4$, $n = 5$, and $n = 9$.
   (c) Have a link to the function file and upload it to Moodle.

   The command `axis equal` may come in handy for these plots!

5. Finish the function `plotVecOber.m` that is found on the H-drive. This function has one or more inputs and uses `varargin`. The first input is `P`, which SHOULD be a vector with 2 or 3 elements (check for it: if not, return an error). The function will take the vector and plot the vector as a line segment from the origin to the given point (`P`). Any other input arguments are optional and should specify how the vector will be drawn (color, line width, marker type, etc.). The function will not have an output (it will only create a plot). *Hint: remember the difference between* `plot` *and* `plot3`*!*
   (a) In the homework file run it once for a 2D point and another time for a 3D point, using your choice of plot specifications (or none) on each.
   (b) Have a link to the function file and upload it to Moodle.

   EXTRA CREDIT: (up to 3 points!) expand the function to `plotVec2Ober.m` that will take a MATRIX of 2D or 3D points and connect them all. Discuss (either on paper or in the text of the homework file) the limitations or difficulties this may have in coding it and/or implementation.

6. Finish the function `plotPlaneOber.m` that is found on the H-drive. This function has 2 or more inputs and uses `varargin`. The first input is `M` which is a matrix should have exactly three rows and at least 2 columns (check for it; if not, return an error). The second input `ax` determines the domain for the plane; the domain will be from `-ax` and `ax`. Check that `ax` is a number (not a vector or matrix) and is positive; if not, return an error. (Any other inputs are optional can specify `EdgeColor`, etc. to plot the plane using the `mesh` command. *Hint: remember* `meshgrid`!)
   (a) Use the function to plot a plane $M =$`randi` - create M and display it and then use your function to plot the plane.
   (b) Have a link to the function file and upload it to Moodle.

To turn in:

1. The URL of the HTML file created by publishing your m-file submitted via Moodle. Have the URL appear in the notes part of turning in the assignment and make it a clickable link.
2. The m-files (both `hw4Ober.m` and all of the function files) uploaded to Moodle by the due date/time.
3. A paper with your written answers written neatly (if appropriate).