Due: Friday, April 6 at 4 PM

- For this assignment you will create 5 files; `hw7Ober.m`, `isnnintOber.m`, `ln2Ober`, `nln2Ober`, and `mycosOber.m` where "Ober" is replaced by the first four letters of your last name.
- The SCRIPT FILE `hw7Ober.m` should be formatted in such a way to be "publishable" as a webpage. Your webpage should appear on the G-drive in the same folders as your previous assignments. Within a browser, copy and paste the URL for your page into Moodle by the due date/time.
- All other directions are the same as the previous assignments.

TIP: Remember that the output of commands aren't shown until you have a `%%`, so you may want to have lines of just `%%` between some of the commands so it publishes a set of commands or comments, then creates output, publishes the next set of commands or comments, then output, etc.

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 `hw7Ober.m` and all of the function files) uploaded to Moodle by the due date/time.

---

1. The first function `isnnintOber` will have one input, n, and the output will be a logical true/-false as to whether n is a nonnegative integer. Note that no error messages are displayed with this function; it returns either `true` or `false`. Within `hw7Ober`, put a link to the function file `isnnintOber.m` and upload it to Moodle.
2. This problem looks at the Alternating Harmonic Series

$$\sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k} = \ln(2).$$

In Calculus or Analysis, using Taylor (Maclaurin) Series on $f(x) = \ln(1 + x)$ it can be shown that the Taylor Series converges when $x = 1$ and you get the above equation. Therefore, the n-th partial sum of the series can be used to estimate ln(2). Your `ln2Ober` function will do the following:

- The function will have one input n which should be a positive integer. Check for it using `isnnintOber`; if not, an appropriate error message is displayed using the `error` command.
- The function will have **two** outputs. The first output is the n-th partial sum of the series:

$$\sum_{k=1}^{n} \frac{(-1)^{k+1}}{k}.$$

A loop can be used for this calculation but it may be fastest if you use vectorized code instead. The second output will be the estimated error of this partial sum using the Alternating Series Estimation Theorem.

Within `hw7Ober`, put a link to the function file `ln2Ober.m`.

3. The next function `nln2Ober` will not calculate partial sums, but instead tell you the minimum n should equal in order to use the `ln2Ober` function to approximate $\ln(2)$ to a certain degree of accuracy ("tolerance level").

   - The `nln2Ober` function has *one* input: a "tolerance level" ε for using the partial sum.
   - Your function should check that $\varepsilon > 0$; if not, an appropriate error message is displayed using the `error` command.
   - The function figures out the minimum value of n (determined by the Alternating Series Estimation Theorem) for the n-th partial sum to have an error less than or equal to ε. The output of the function is this value n.

   Within `hw7Ober` file, put a link to the function `nln2Ober.m`.

4. This problem is looking at Taylor/Maclaurin Series and Taylor Polynomials for $f(x) = \cos(x)$.

$$\cos(x) = \sum_{k=0}^{\infty} \frac{(-1)^k \, x^{2k}}{(2k)!}$$

   Create a function called `mycosOber.m` that calculates the n-degree Taylor Polynomial for $f(x) = \cos(x)$ using the above Maclaurin Series for $\cos(x)$. Make sure that n is a **nonnegative integer** using your function `isnnintOber`; if not, an appropriate error message is displayed using the `error` command. The calculations should be such that x can be a number, vector, or matrix. The input n is the **highest degree term that will appear in the polynomial**. Thus if $n = 8$, the degree of the polynomial will be no bigger than 8. Notice that if we enter $n = 9$, we should get the same as $n = 8$. Careful: if we want $n = 8$, what should k equal for the last term in the sum? If we input $n = 9$, what should happen? Using `floor`, `ceil`, or `fix` may be useful here.

   (a) Within `hw7Ober` file, put a link to this function `mycosOber.m`.

   (b) Plot the Taylor Polynomials for $x \in [-10, 10]$ for $n = 4$, $n = j$, $n = k$, and $n = 24$ using the above function `mycosOber` along with $y = \cos(x)$. Graph them all on the same figure.

   - The values of j and k will be random integers between 6 and 12 determined using the `randi` command. Use a `while` loop to make sure $k \neq j$.
   - Have $y = \cos(x)$ dotted and in black, and have the others in different colors and/or shapes (dotted, dashed, etc.). Use your own judgement on `LineWidth`, etc. to make the graphs clear.
   - Have an appropriate legend and title. (`sprintf` may be useful to put the values of n in the legend!)
   - The vertical axis should only be between -2 and 2.

5. **Extra Credit** (up to 4 points) Use your functions `nln2Ober` and `ln2Ober`to estimate $\ln(8)$ accurate to 4 decimal places. Careful! What would the error tolerance need to be within that many decimal places, and how do you adjust the error for $\ln(2)$ to get the estimate for $\ln(8)$? Use the value of n you get from `nln2Ober` in your function `ln2Ober` to estimate $\ln(8)$.