ENGR 121: Computation Lab I (Midterm Exam)

The exam is due Monday, November 2, 2015, 11:59 pm. It comprises four questions worth ten points each. Please read the *Submission Instructions* section towards the end of this document that describes how to package your exam and submit it on-line using both the BBLearn and Cody systems. **You must work on this exam by yourself.**

1. The harmonic mean of the n values in a data set $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ is defined as

$$\frac{n}{1/x_1 + 1/x_2 + 1/x_3 + \cdots + 1/x_n}.$$

Write a function called `h_mean` that takes as input the data set or vector `x` and returns the harmonic mean value as the output. Your code should be vectorized using MATLAB's built-in vector operations, that is there is no need to use loops. An example of correct function behavior is as follows:

```
>> x = ones(1, 10);
>> m = h_mean(x)
m =
     1
>> x = randi([1, 10], 1, 10);
>> m = h_mean(x)
m =
    2.9412
```

2. The Maclaurin series expansion for the cosine function up to some order n is given by

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \cdots, \frac{x^n}{n!},$$

where x is the angle in radians and $n!$ denotes the factorial of n. So, the cosine approximations for $n = 4$, $n = 6$, and $n = 10$ are as follows:

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!}$$
$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!}$$
$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \frac{x^{10}}{10!}$$

Write a function `cosineApprox` that takes scalar input arguments `x` and `n`, and returns the approximate value of $\cos(x)$. So, the function will be invoked as follows:

```
approx = cosineApprox(x, n);
```

Use the built-in MATLAB function **factorial** when computing the series expansion. Assume that `x` is in radians. Examples of correct function behavior for various values of x and n follow:

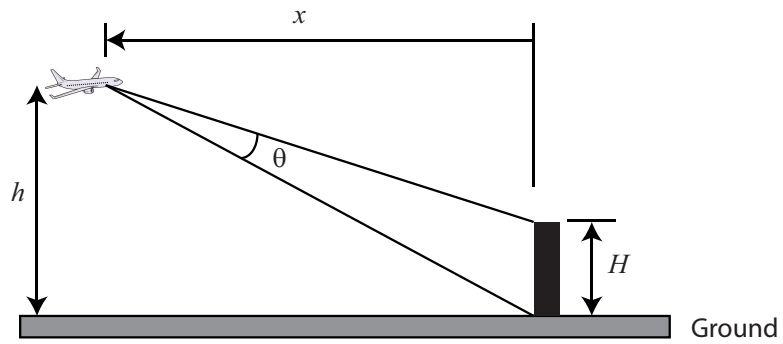
```
>> x = pi/3; n = 4;
>> approx = cosineApprox(x, n)
approx =
0.5018
>> n = 8;
>> approx = cosineApprox(x, n)
approx =
0.5000

>> x = 2/3*pi; n = 4;
>> approx = cosineApprox(x, n)
approx =
-0.3915
>> n = 12;
>> approx = cosineApprox(x, n)
approx =
-0.5000
```

Finally, note that your code should use MATLAB's built-in vector operations to solve this problem. You must not use `for` and `while` loops.

3. Write a function `makemat` that will receive two row vectors as input arguments, and from them create and return a matrix with two rows. You may not assume that the length of the vectors is known *a priori*. Also, the vectors may be of different lengths. If that is the case, add 0s to the end of one vector first, that is pad it, to make it as long as the other. For example, calls to the function might be:

```
>> thismat = makemat([1:5], [2:9])
thismat =
     1     2     3     4     5     0     0     0
     2     3     4     5     6     7     8     9
>> thismat = makemat([2:6], [1:3])
thismat =
     2     3     4     5     6
     1     2     3     0     0
>> thismat = makemat([2:6], [10:14])
thismat =
     2     3     4     5     6
    10    11    12    13    14
```

4. In the figure below, an airplane is flying at a height of h ft while observing a target that is H ft tall. The best view of the target is when θ is maximum.



Write a MATLAB function `maxObservability` that determines the distance x at which θ is maximum. Define the vector `x` with elements ranging from 50 to 1500 with spacing of 0.5. The function is invoked as follows:

```
best_x = maxObservability(x, h, H)
```

Finally, note that your solution should work correctly for any given values of h and H. For example, if the plane is flying at a height of $h = 900$ ft and the height of the target is $H = 70$ ft, then the correct function behavior is:

```
>> x = 50:0.5:1500;
>> h = 900; H = 70;
>> m = maxObservability(x, h, H)
m =
864.5000

>> h = 500; H = 70;
>> m = maxObservability(x, h, H)
m =
463.5000
```

Submission Instructions

- Submit these files via BBLearn under the link called 'Midterm exam' by November 2, 2015, 11:59 pm: `h_mean.m`, `cosineApprox.m`, `makemat.m`, and `maxObservability.m`. Note that you may upload your work more than once; the latest version will be evaluated.
- In addition, you must submit and test all your answers on Cody by November 2, 2015, 11:59 pm. The submission site will be up and running by Saturday afternoon at the latest. we will send out an email to the class once the site is up.