**DM 1:** Write a small Matlab function `largest(a,b)` that returns the largest of the two values. Test that your function works by computing `largest(1,2)`, `largest(0,-1)` and `largest(5,5)`.

**Solution:**
Code:

```
Your code here!
```

Output:

```
Your output here!
```

**DM 2:** Write a small Matlab function `nextprime(x)` that takes a positive integer argument and returns the smallest prime number at least as large as `x`. Your function should use a `while` loop and take advantage of the `isprime` function in Matlab. Test that your function works by computing `nextprime(5)`, `nextprime(6)`, `nextprime(-1)` and `nextprime(100)`.

**Solution:**
Code:

```
Your code here!
```

Output:

```
Your output here!
```

**DM 3:** Define a sequence of numbers by $x_1 = 1$ and $x_{k+1} = \frac{1}{2}x_k + 1$. Write a Matlab function `buildseq(N)` that returns an array with the first `N` elements of the sequence in it. For example, `buildseq(2)` should return $[1, 1.5]$. Test that your function works by computing the first four sequence elements by hand, and then verifying that your function computes them correctly. You may wish to take advantage of the Matlab command `zeros`.

**Solution:**
Code:

```
Your code here!
```

Expected Output:

Your explanation here.

Output:

```
Your output here!
```

**DM 4:** Write a function `HW2bisect` that does the following:

1. Takes the following input:
   - `f`, the name of a function
   - Numbers `a` and `b` that (supposedly) bound an interval containing a root of `f`.
   - `delta`, a number determining the accuracy of the sollution.

Your function should approximate a root of $f$ in the interval $[a, b]$ by bisection. The approximation $x$ should be within $\delta$ of a root

The function should return:

1. `x`, the approximate root
2. A $n \times 2$ matrix `history` that contains a list of the interval endpoints at each stage, starting with the initial guess provided.

$$\texttt{hist} = \begin{pmatrix} a_0 & b_0 \\ a_1 & b_1 \\ \vdots & \vdots \\ a_n & b_n \end{pmatrix} \tag{1}$$

Your function should have help and should produce error messages appropriately when the function fails.

Test your function works by applying it to the function $f(x) = x^2 - 4$ with starting interval $[0, 5]$.

**Chapter 4: 2(a)** (You should use the code developed in the previous problem.)**:**

**Chapter 4: 18:** Produce a set of four plots of the first four Taylor polynomials of the function $f(x) = e^{1-x^2}$, expanded around the point $x = 1$.