Engineering Analysis I, Fall 2017

Midterm 1

Name ______________________________

Section number ______________

Section number	Discussion time	Instructor
30	9:00 a.m.	Ilya Mikhelson
31	10:00 a.m.	Ilya Mikhelson
32	10:00 a.m.	Iman Hassani
33	11:00 a.m.	Iman Hassani
34	12:00 noon	Randy Berry

This exam is closed-book and closed-notes. Calculators, computers, phones, or other computing/communication devices are not allowed.

Students should skip this page—it is only for graders.

Question	Points	Score
1	28	
2	20	
3	25	
4	27	
Total:	100	

Answer each question in the space provided. There are 4 questions for a total of 100 points.

1. (a) [8 points] Which of the following MATLAB statements will *not* generate an error message?

☐ `x = [1 2 3].*[4 5]`

☐ `x = rand(4,5) < 0.3`

☐ `x = (5<6) && (6>7)`

☐ `x = sin(1:5).^4`

☐ `x = [2 3]*[4;5]`

☐ `x = eye(3)*[4;5]`

☐
```
if 6 = 1 + 5
    x = 2
end
```

☐
```
while k = 1:3
    x = k + 1
end
```

(b) [8 points] Which of the following eight blocks of code will triple each element of an existing (and possibly non-square) matrix `A`?

☐
```
3*A = A;
```

☐
```
A = A*3
```

☐
```
A = 3*A;
```

☐
```
A = A.^3
```

☐
```
A = A + 2*A;
```

☐
```
for ii = 1:size(A)
    A(ii,ii) = 3*A(ii,ii);
end
```

☐
```
ii = 1;
x = size(A,1);
while ii <= x
     A(:,ii) = 3*A(:,ii);
     ii = ii + 1;
end
```

☐
```
for ii = 1:size(A,1)
   for jj = 1:size(A,2)
      A(ii,jj) = 3*A(ii,jj);
   end
end
```

(c) [2 points] A system of linear equations in non-standard form is $\mathbf{My} + \mathbf{Nz} = \mathbf{b}$, where $\mathbf{M}$ and $\mathbf{N}$ are known matrices, $\mathbf{b}$ is a known vector, and $\mathbf{y}$ and $\mathbf{z}$ are unknown vectors. To bring this system into the standard form $\mathbf{Ax} = \mathbf{b}$, we define the unknown vector $\mathbf{x} = \begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix}$, that is, $\mathbf{x}$ is the vertical concatenation of the vectors $\mathbf{y}$ and $\mathbf{z}$. How would you combine the matrices $\mathbf{M}$ and $\mathbf{N}$ to obtain the matrix $\mathbf{A}$ for the standard form? **(There is one and only one correct answer.)**

- ☐ $\mathbf{A} = \mathbf{M} + \mathbf{N}$
- ☐ $\mathbf{A} = \begin{bmatrix} \mathbf{M} \\ \mathbf{N} \end{bmatrix}$ (vertical concatenation)
- ☐ $\mathbf{A} = \begin{bmatrix} \mathbf{M} & \mathbf{N} \end{bmatrix}$ (horizontal concatenation)

(d) [6 points] Suppose the MATLAB variable `A` contains an array of real numbers. Which of the following statements will create a variable `B` containing an array the same size as `A` whose entries are equal to -1 whenever the corresponding entry of `A` is negative and equal to 0 otherwise?

☐	`B = (A>=0) - 1;`	☐	`B = A - 1;`
☐	`B = (A>=0);`	☐	`B = (A>=1);`
☐	`B = -(A<0);`	☐	`B = -(~(A>=0));`

(e) [4 points] Given the assignments

```
a = [4 3 8 2; 1 2 4 5; 4 7 2 3];
b = a([1 3], [2 2])
```

what does MATLAB return for `b`?

2. Suppose each section of code below is run in MATLAB. If MATLAB generates an error message for the given code section, write "error" on the associated line. Otherwise, write the value that the variable **x** will have after the code section is run.

(a) [4 points]

```
clear all;
x = 2;
if ~(x^3 == 8) || (x ~= x)
    x = x - 2;
end
```

(a) ______________

(b) [4 points]

```
clear all;
x = 10;
for k = 9.5:-3:0.5
    x = x - 1;
end
```

(b) ______________

(c) [4 points]

```
clear all;
x = 4;
for k=3:2:115
    if k < x
       x = x + 1;
    elseif k == x
       x = x + 2;
    else
       x = 0;
    end
end
```

(c) ______________

(d) [4 points]

```
clear all;
x = 15;
while x > y
    x = x - 1;
end
```

(d) ______________

(e) [4 points]

```
clear all;
y = 16;
x = 2;
while (y >= x) && (x > 0)
    if x >= y
       x = x - y
    else
       x = x + 2
    end
end
```

(e) ______________

3. [25 points] For this question, we will first generate a random matrix of integers (between 2 and 10^9), with random dimensions (between 1 and 10). Our goal is to find, for each element, the highest number whose factorial* is less than that number. For example, if an element of `A` is 1000, the corresponding element in `B` would be 6, since $6! = 720$ (and $7! = 5040 > 1000$). You will create a matrix of the same size as the original, where each element is this highest factorial. For example, consider the following input matrix `A` and output `B`.

$$\texttt{A} = \begin{bmatrix} 1000 & 3000 \\ 50 & 121 \\ 100000 & 4000000 \end{bmatrix} \qquad \texttt{B} = \begin{bmatrix} 6 & 6 \\ 4 & 5 \\ 8 & 10 \end{bmatrix}$$

Fill in the blanks below to complete this task.

Note: The `factorial` function in MATLAB computes the factorial of a number. For example, `factorial(6)` computes 6!.

*The "factorial", denoted by an exclamation point, is defined as:

$$n! = n \cdot (n-1) \cdot (n-2) \cdots 1$$

```
% Create the input random matrix.
A = randi([2 1000000000],randi(10),randi(10));

% Find dimensions of A (where m is the number of rows and n is
% the number of columns.
m =  ________________________________________________
n =  ________________________________________________

% Create an output matrix B filled with zeros.
B =  ________________________________________________

for ii = 1:m
    for jj =  ___________________________________________
        my_fact =  _______________________________________

        % Iterate as stated in the introduction.
        while  __________________________________________
            my_fact = my_fact + 1;
        end

        % Fill in the corresponding value in the B matrix.
        B(ii,jj) =  _____________________________________
    end
end
```

4. [27 points] For this question, we will estimate pi using an iterative method. Pi can be computed using the Gregory-Leibniz Series as:

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \cdots = \sum_{k=1}^{\infty} (-1)^{k-1} \frac{4}{2k-1}$$

Your goal is to keep adding terms until your subsequent estimates are within 0.0000001 of each other. **Fill in the missing code below so that the script performs this calculation.**

```
% Create and initialize variables to hold your adjacent estimates of pi.
pi_new = 4;
pi_old =  ____________________________________________________________

% Create and initialize a variable to keep track of the number of terms
% in your summation
k =  _________________________________________________________________

% Iterate as stated in the introduction
while  _______________________________________________________________
    pi_old =  ________________________________________________________
    pi_new =  ________________________________________________________
    k =  _____________________________________________________________
end

% Display your value of pi to 5 decimal places and the number of terms
% you used (e.g "Pi: 3.14159, Terms: 30000").
fprintf(  ____________________________________________________________
```