Homework 1 Due Today
(homework solutions will be posted on the web site after class the day the assignment is due or two days after if anyone is late turning it in)

Homework 2 Due September 12

Readings for Today
Attaway Chapters 2, 7, and 8

Readings for Next Week
Attaway Chapters 3, 4, and 5
Week2.m

I will try to make the in-class Matlab code readable enough so that you can follow along the Matlab code when I am lecturing from my Keynote slides
Email list

there were too many errors and omissions in the paper email list I collected, so I will just send emails out via Blackboard

anyone sitting in who is not registered as a student will not get the emails, but in general the content of those emails will be repeated in class or really only relevant for those taking the class for credit
Scripts in Matlab
Scripts in Matlab

• The Command Window is useful for trying out commands and debugging, but we will use Matlab scripts (.m files) for most things.

• "Script" is just a series of commands. Rather than typing them, you run the script. Later we will talk about functions.

• Being an interpreted language, a complex program in Matlab is really just a more elaborate script.
Creating a script

- Change Current Folder
- Create Script
  - New ➔ Script
- Save / Save As
  - Save ...
  - Save As ...
- “Dock” / “Undock” script window
  (useful if you have multiple monitors)
Best Practices

• Use folders/directories, subfolders/subdirectories
• Make a copy of a working program before you make new edits to it.
• Keep copies of experiment programs and analysis programs with any data you collect in an experiment.
Comments in Scripts

% this is a comment

>> x = 2  % this is a comment too

Use comments to:
- briefly describe what a section of code does
- reference particular formulas or calculations
- describe variables the first time you assign them
- write comments for yourself
- write comments for others in your lab
Running a Script

Green Arrow
  Save and Run

Debugging
  setting breakpoints in code
Defining and Executing Sections

%% a comment and a start of a section

sections should be self-contained units of code

double-click (left click) to select a section

Evaluate Current Section

break points don't work when evaluating a section
Homework Assignments

Turn in .m file(s) on Blackboard; if there are multiple files, turn in as a single ZIP file

Use comments to indicate which part of a question you are answering

If I ask a question that requires a short answer, you can usually answer it in a comment within the script

For a longer answer you might need to submit a pdf file or a word file
Homework Assignments

include all files needed to run your code when you submit your solutions, even files that I have supplied

if you have more than 1 or 2 files, please submit them as a ZIP file

clearly inform me which file I should run to test your program
Homework Assignments

Unless required by the assignment to do otherwise, make sure your lines end with a ; to suppress output - in the class .m files I sometimes leave the ; off so that output is generated in the Command Window.

please use %% to separate logical sections of your code – both as a matter of style, and so I can run subsections of your code when grading.
QUIZ
Poll Everywhere
https://www.polleverywhere.com

online:
pollev.com/thomaspalmer272

free iphone / android app:
search for Poll Everywhere app

message:
text to thomaspalmer272 to 22333
\[ 16 \quad ^\wedge \quad 1/2 \]

Your poll will show here

1. Install the app from pollev.com/app
2. Make sure you are in Slide Show mode

Still not working? Get help at pollev.com/app/help or Open poll in your web browser
$\gg x = 1;$
$\gg y = 2;$
$\gg z = 3;$
$\gg (x > y) \:\mid (x > z)$
>> x = 1;
>> y = 2;
>> z = 3;
>> (x > 0) & (z > y)
```plaintext
>> x = 1;
>> y = 2;
>> z = 3;
>> (x > y) | z
```

Your poll will show here

1 Install the app from pollev.com/app
2 Make sure you are in Slide Show mode

Still not working? Get help at pollev.com/app/help
or
Open poll in your web browser
$$a = 0;$$
$$b = 2;$$
$$a = 4 | b < a$$
Strings
String Types

```matlab
>> x = 'A'
>> whos x
>> x = 'ABC'
>> whos x
```

There are lots of things you can do with strings.

We’ll talk more about strings later.

String Operations

>> strcmp(a,c)
strcmp is case sensitivity

>> strcmpi(a,c)
strcmpi is not
String Operations

```plaintext
>> a = 'abc'
>> b = 'abc '
>> strcmp(a,b)

string comparison is not intelligent - we’re used to search engines like google that take care of misspellings and extra spaces gracefully
```
Data Structures
Data Structures

one key to successful programming is using the right kind of data structure and using it the right way
Data Structures in Matlab

- Arrays, Vector, Matrices
- Structures
- Cell Arrays
- Tables
- Categorical Arrays

Matlab does not so easily support more sophisticated data structures (queues, trees, hash tables, etc.), which is why some people use Python or C++
Arrays
Why use a data structure?

Imagine we have 10 subjects and each subject answers 10 true/false questions. 100 data points.

We could create 100 variables, s1q1, s2q2, ... s2q1, s2q2, ... s10q9, s10q10.

Why do we use a data structure instead?
Why use a data structure?

Imagine we have 10 subjects and each subject answers 10 true/false questions. 100 data points.

We could create 100 variables, s1q1, s2q2, ... s2q1, s2q2, ... s10q9, s10q10.

Why do we use a data structure instead?
- access data more easily and more efficiently
- access data dynamically
- all the data is “in the same place”
Arrays

After simple variables, arrays are the most common data structure used in just about every language.

creating a one-dimensional array in Matlab

```matlab
>> a = [1 2 3 4]
```

referencing an element

```matlab
>> a(2)
```

changing the value of an element

```matlab
>> a(2) = 10
```
Arrays

remove an element from a one-dimensional array
```plaintext
>> a(2) = []
```

adding an element to an array
```plaintext
>> a(5) = 5
```

but note this
```plaintext
>> a(10) = 10
```

regular arrays, by definition are contiguous

there are advanced data structures called sparse matrices too:
One nice thing about Matlab is that it lets you dynamically add new elements to an array. Other languages force you to say how big an array is at the outset (e.g., C++) and you have to resize it explicitly to add more elements. If you go past the boundaries of the array, the program crashes.

but this can also make it harder to debug a program
Two-dimensional Arrays

\[
\text{a} = \begin{bmatrix}
1 & 2 & 3 \\
4 & 5 & 6
\end{bmatrix}
\]

\text{size(a)}

2 rows
3 columns

Now the meaning of ; changes. Within [ ] it creates a new row in an array.
Two-dimensional Arrays

\[
\begin{bmatrix}
1 & 2 \\
3 & 4 \\
5 & 6
\end{bmatrix}
\]

How many columns and rows will this have?
Two-dimensional Arrays

\[ b = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \]

How many columns and rows will this have?

\[ \text{size}(b) \]

Note: It is entirely up to you to keep straight what the rows and columns MEAN. Rows could be “subjects”. Columns could be “subjects”. It’s up to you.*

* other data structure impose more "structure"
Two-dimensional Arrays

\[
\begin{bmatrix}
1 & 2 & 3 \\
4 & 5 & 6
\end{bmatrix}
\]

What will happen here?

\[
a(4, 1)
\]

Why?
Two-dimensional Arrays

Here?

```python
>> a(4.1, 1.2)
```

Why?
Two-dimensional Arrays

Here?

```python
>> a(4.1, 1.2)
```

Why?

What if you referenced an array like this

```python
>> a(i,j)
```

You need to make sure i and j are whole numbers.
We’re already been using arrays

```matlab
>> x = 1
>> size(x)
```

1 row
1 column

*there are no true "scalars" in Matlab (unlike other languages)*
What does [ ] actually do?

It’s a concatenation operator.
concatenate = link together in a chain

\[
[ \begin{array}{c}
    \text{Red} \\
    \text{Red} \\
    \text{Red} \\
\end{array} ] = \begin{array}{c}
    \text{Red} \\
    \text{Red} \\
    \text{Red} \\
\end{array}
\]

\[
\begin{array}{c}
    \gg \ x \ = \ 1 \\
    \gg \ y \ = \ [x \ x \ x]
\end{array}
\]
What does [ ] actually do?

What do you think this will do?

```python
>>> x = 1
```

```python
>>> y = [x x x]
```

```python
>>> z = [y y]
```
What does [ ] actually do?

What do you think this will do?

```python
>>> x = 1
```

```python
>>> y = [x x x]
```

```python
>>> z = [y y]
```

```
[ x x x ] = [ x x x x x x ]
```
What does [ ] actually do?

How about this?

```plaintext
>> x = [2 ; 3]
>> y = [4 ; 5]
>> z = [6 ; 7]
>> [x y z]
```
What does [ ] actually do?

How about this?
```
>> x = [2 ; 3]
>> y = [4 ; 5]
>> z = [6 ; 7]
>> [x y z]
```

```
```

=  

```
[ [2 2] [2 3] ]
[ [4 4] [4 5] ]
[ [6 6] [6 7] ]
```

=  

```
[ [2 2 2] [2 3 3] ]
[ [4 4 4] [4 5 5] ]
[ [6 6 6] [6 7 7] ]
```
What does [ ] actually do?

This?

```python
>>> x = [1 2 3]
>>> y = [4 5]
>>> z = [6 7 8]
>>> [x y z]
```

What does [ ] actually do?

This?
```
>> x = [1 2 3]
>> y = [4 5]
>> z = [6 7 8]
>> [x y z]
```

```python
[ ]
```
What does [ ] actually do?

`>> w = [1 1 ; 1 1]`
`>> x = [2 2 ; 2 2]`
`>> y = [3 3 ; 3 3]`
`>> z = [4 4 ; 4 4]`

This?

`>> [w x y z]`
What does [ ] actually do?

```plaintext
>>> w = [1 1 ; 1 1]
>>> x = [2 2 ; 2 2]
>>> y = [3 3 ; 3 3]
>>> z = [4 4 ; 4 4]

This?
```
What does [ ] actually do?

```
>> w = [1 1 ; 1 1]
>> x = [2 2 ; 2 2]
>> y = [3 3 ; 3 3]
>> z = [4 4 ; 4 4]
```

This?

```
>> [w x ; y z]
```
What does [ ] actually do?

```
>> w = [1 1 ; 1 1]
>> x = [2 2 ; 2 2]
>> y = [3 3 ; 3 3]
>> z = [4 4 ; 4 4]
```

This?

```
>> [w x y ; z]
```
What does [ ] actually do?

This?

```
>> x = [2 ; 3]
>> y = [4 5]
>> [x y]
```
What does [ ] actually do?

This?

```python
>>> x = [2 3 ; 4 5]
```
```python
>>> y = [6 ; 7]
```
```python
>>> [x y]
```
Strings are actually character arrays

```python
>> a = 'apple'
>> size(a)
```

```python
>> b = 'banana'
>> size(b)
```

What do you think this does?

```python
>> [a b]
```
Strings are actually character arrays

```python
>> a = 'apple'
>> size(a)

>> b = 'banana'
>> size(b)

These?
>> [a ' ' b]
>> a(4)
```
How are arrays stored in memory?

Create a variable

```plaintext
>> x = 2
```

![Diagram showing memory allocation for O/S, Matlab, and Data.]
How are arrays stored in memory?

Create a variable

```plaintext
>> x = 2
```
How are arrays stored in memory?

Create a variable

```plaintext
>>> x = 2
```
How are arrays stored in memory?

Create a variable

```python
>>> x = 2
```

Length 1
How are arrays stored in memory?

Create a variable

```python
>>> x = [2, 3, 4]
```

```
x length 1
```
How are arrays stored in memory?

Create a variable

```
>>> x = [2, 3, 4]
```

$x$ length 3
How are arrays stored in memory?

Create a variable

\[
\gg \ x(4) = 5
\]
How are arrays stored in memory?

Create a variable

```plaintext
>> x(4) = 5
```

x length 4
Preallocating arrays

Creating and moving large arrays every time a new index is added can take a lot of time.

Preallocate

\[
\begin{align*}
\text{>> } & \ x = \text{zeros}(1,100) \\
\text{or} & \\
\text{>> } & \ x = \text{ones}(1,100) \\
\text{or} & \\
\text{>> } & \ x = \text{zeros}(5,20)
\end{align*}
\]
How are two-dimension arrays stored?

```
>> x = [1 2 3; 4 5 6]
```
How are two-dimensional arrays stored?

```plaintext
>> x = [1 2 3; 4 5 6]
```

x total length 6
How are two-dimension arrays stored?

```plaintext
>> x = [1 2 3; 4 5 6]  
```

How is a 2D array stored?

Column Major Order

`x` total length 6
How are two-dimension arrays stored?

```plaintext
>> x = [1 2 3; 4 5 6]
```

What will these spit out?

```plaintext
>> x(1,1)
>> x(1,2)
>> x(1,3)
>> x(1)
>> x(2)
>> x(3)
```

These?
How are two-dimension arrays stored?

```plaintext
>> x = [1 2 3; 4 5 6]
```

What will these spit out?

```plaintext
>> x(1,1)
>> x(1,2)
>> x(1,3)
```

These? can be a dangerous way to reference a multidimensional array if you are not very careful

```plaintext
>> x(1)
>> x(2)
>> x(3)
```

can be a difficult "bug" to detect as well
size vs. length of an array

```matlab
>> clear all
>> x = [1 2 3 4 5 6]
>> size(x)
>> length(x)

>> x = [1 2 ; 3 4 ; 5 6]
>> size(x)
>> length(x)
```
Extracting part of an array

```plaintext
>> a = [11 12 13 14 15 ; 21 22 23 24 25 ; ...
31 32 33 34 35 ; 41 42 43 44 45 ; ...
51 52 53 54 55];
```

**How do we extract row 2?**

```plaintext
>> [a(2,1) a(2,2) a(2,3) a(2,4) a(2,5)]
```
Extracting part of an array

```plaintext
>> a = [11 12 13 14 15 ; 21 22 23 24 25 ; ...
    31 32 33 34 35 ; 41 42 43 44 45 ; ...
    51 52 53 54 55];
```

**How do we extract row 2?**

```plaintext
>> [a(2,1) a(2,2) a(2,3) a(2,4) a(2,5)]
```

Here’s an easier way:

```plaintext
>> a(2,[1 2 3 4 5])
```
Using the colon : operator

```
>> 1:5
```
creates a sequence within an array
Using the colon : operator

```
>> 1:5
```

creates a sequence within an array

Try these.

```
>> a
```

```
>> a(2,1:5)
```

```
>> a(1:5, 2)
```
Using the colon : operator

```plaintext
>> 1:5
```
creates a sequence within an array

Try these.

```plaintext
>> a
>> a(2, 1:5)
>> a(1:5, 2)
```

**What will this give you?**

```plaintext
>> a(1:2, 1:2)
```
Using the colon : operator

: by itself within an array

```plaintext
>> a(2, :)
>> a(:, 2)
>> a(:, :)
```
Using the colon : operator

Create a 10x10 array containing the integers between 0 and 99.
Using the colon \( : \) operator

Create a \textbf{10x10} array containing the integers between 0 and 99.

\[
\begin{array}{c}
\text{a = [0:9 ; 10:19; 20:29; 30:39; \ldots} \\
\text{40:49; 50:59; 60:69; 70:79; 80:89; \ldots} \\
\text{90:99]};
\end{array}
\]
Create a 10x10 array containing the integers between 0 and 99.

```matlab
>> a = [0:9 ; 10:19; 20:29; 30:39; ... 40:49; 50:59; 60:69; 70:79; 80:89; ... 90:99];
```

```matlab
>> a = zeros(10,10);
```

```matlab
>> a = 0:99
```
Using the colon : operator

Create a 10x10 array containing the integers between 0 and 99.

```matlab
>> a = [0:9 ; 10:19; 20:29; 30:39; ... 40:49; 50:59; 60:69; 70:79; 80:89; ... 90:99];
>> a = zeros(10,10);
>> a = 0:99
>> a = reshape([0:99],10,10);
```
transpose (') – swapping rows and columns

```plaintext
>> x = [1 2 3 4 5]
>> x
>> x'
```

```plaintext
>> y = [1 2 ; 3 4 ; 5 6]
>> y'
```

(Note: Technically, if you just want to transpose rows and columns and keep the numbers exactly the same, you should use .' instead of ', but it only makes a difference if you have complex numbers.)
Multidimensional Arrays

By default, all arrays in Matlab are 2-dimensional.

You can create arrays with more than 2 dimensions.
Multidimensional Arrays

e.g., imagine a within-subjects design
rows are subjects
columns are levels along IV #1
depth are levels along IV #2

each entry \((x,y,z)\) within the array is the score for subject \(x\), along level \(y\) of IV #1 and level \(z\) of IV #2
Creating a multidimensional array

No way to create one directly using the [ ] operator.

Remember, [ ] simply concatenates. While it looks like [ ] creates a two-dimensional array from one-dimensional elements (e.g., [2 3 ; 4 5]), recall that even a number like “2” is technically a two-dimensional array (1 row, 1 column) already.

So, while [ ] can concatenate three-dimensional arrays, it cannot create them.
Creating a multidimensional array

• Preallocating
  >>> x = zeros(20,10,5)

You need to remember what the dimensions mean. And this would be impossible to understand if you saw it in someone’s Matlab script.
Creating a multidimensional array

• This is better style
  >> Nsubj = 8;
  >> NIV1 = 5;
  >> NIV2 = 3;
  >> data = zeros(Nsubj, NIV1, NIV2);
Creating a multidimensional array

• This is better style
  >> Nsubj = 8;
  >> NIV1 = 5;
  >> NIV2 = 3;
  >> data = zeros(Nsubj, NIV1, NIV2);

  for i=1:Nsubj
    for j=1:NIV1
      for k=1:NIV2
        data(i,j,k) = 100*i + 10*j + k;
      end
    end
  end
The only number (2, 125, 25.1) you should EVER see in a computer program is one that is an intrinsic part of a calculation or a formula.

e.g.,

\[
f = \frac{1}{(\text{sig}\times\sqrt{2\pi})} \times \ldots \exp\left(-\frac{(x-\text{mu})^2}{2\times\text{sig}^2}\right)
\]

for \( i = 1 : \text{size}(\text{data},1) \)

end
Best Practices

You NEVER EVER want to see “64” in a loop if you have 64 trials, or “3” in a loop if you have 3 blocks, or “2.1” in a formula if that’s some scaling factor you picked to adjust image contrast.

Make those variables with meaningful names and comment them.

Even if this is a "one-off" analysis program, you might need to go back to it in 6 months or more after a couple rounds of reviews through a journal.
In all homework assignments, I will expect that you use good programming style.
Referencing a multidimensional array

Back to our data example ...

How would you pull out the data for subject 3?
Referencing a multidimensional array

How would you pull out the data for subject 3?
Referencing a multidimensional array

How would you pull out the data for subject 3?

```matlab
>> x = data(3,:,:,:)
```

```matlab
>> size(x)
```
Referencing a multidimensional array

How would you pull out the data for subject 3?

```matlab
>> x = data(3,:,:,:)
>> size(x)
```

You might want to reduce this to two dimensions.

```matlab
>> x = squeeze(data(3,:,:,:))
```
Referencing a multidimensional array

Matlab is mind-numbingly arbitrary sometimes. Try this now ...

```matlab
>> x = data(:,:,:1)
>> size(x)
```
Removing data?

Can you remove subject three?
Removing data?

Can you remove subject three?

```plaintext
>> data(3,:,:,:) = []
```
Removing data?

Can you remove subject three?

```matlab
>> data(3,:,:,:) = [ ]
```

Can you remove level 2 of IV #1?
Removing data?

Can you remove subject three?
>> data(3,:,:,:) = []

Can you remove level 2 of IV #1?
>> data(:,2,:,:) = []
Removing data?

Can you remove subject three?

```matlab
>> data(3,:,:,:) = []
```

Can you remove level 2 of IV #1?

```matlab
>> data(:,2,:) = []
```

Now what if you wanted to remove subject four?
Removing data?

Can you remove subject three?
>> data(3,:,:,:) = []

Can you remove level 2 of IV #1?
>> data(:,2,:) = []

Now what if you wanted to remove subject four?
Is this right?
>> data(4,:,:,:) = []
Removing data?

Can you remove subject three?
```
>> data(3,:,:,:) = [ ]
```

Can you remove level 2 of IV #1?
```
>> data(:,2,:,:) = [ ]
```

Now what if you wanted to remove subject four?

Is this right?
```
>> data(4,:,:,:) = [ ]
```

Which row has subject 4′s data now?
Removing data?

Can you remove subject three?

```matlab
>> data(3,:,:,:) = []
```

Can you remove level 2 of IV #1?

```matlab
>> data(:,2,:,:) = []
```

Now what if you wanted to remove subject four?

Is this right?

```matlab
>> data(4,:,:,:) = []
```

Which row has subject 4’s data now?

```matlab
>> data(3,:,:,:) = []
```
Removing data?

Suppose you think that the data for subject 1 is suspect, but only for level 1 on IV#1 and level 3 on IV#2. **Can you remove that one data point?**
Removing data?

Suppose you think that the data for subject 1 is suspect, but only for level 1 on IV#1 and level 3 on IV#2. **Can you remove that one data point?**

```plaintext
>> data(1,1,3) = [ ]
```

Will this work?
Removing data?

Suppose you think that the data for subject 1 is suspect, but only for level 1 on IV#1 and level 3 on IV#2. **Can you remove that one data point?**

```matlab
>> data(1,1,3) = []
```

Will this work?

No. But people sometimes do this.

```matlab
>> data(1,1,3) = NaN
```
Best Practices

A common (and nasty) problem people run into is that they don’t realize they are reusing a variable, especially an array that’s already been set up.

What’s especially troubling is that they try to “restart Matlab” and the problem goes away, so they think it was just a Matlab hiccup.

Why does it go away when you restart Matlab?
A common (and nasty) problem people run into is that they don’t realize they are reusing a variable, especially an array that’s already been set up.

What’s especially troubling is that they try to “restart Matlab” and the problem goes away, so they think it was just a Matlab hiccup.

Why does it go away when you restart Matlab? The Workspace is clear.
Another example

How would we create a multidimensional array to hold the data for a mixed design where there could be unequal numbers of subjects in the between-subjects levels?

One B/S variable with 3 levels
10 subjects in level 1
12 subjects in level 2
8 subjects in level 3

One W/S variable with 2 levels

One W/S variable with 5 levels
Another example

```matlab
>> NBet = 3
>> NWith1 = 2
>> NWith2 = 5
>> NMaxSubj = 12

>> Nsubj = zeros(1,NBet)
>> Nsubj(1) = 10
>> Nsubj(2) = 12
>> Nsubj(3) = 8

>> data = ... different from how you might code data in Excel or SPSS
zeros(NBet,NMaxSubj,NWith1,NWith2)
```
Simple I/O in Matlab

For Homework Assignment
(read these notes, the book, and online help)

save, load, input, fprintf
Simple I/O in Matlab

We’ll talk about more complex I/O later ...

Files or User Input
Simple I/O in Matlab

Save data in Matlab (.mat) format:

```matlab
>> save filename.mat
``` saves all variables in the current workspace in file named filename.mat

```matlab
>> save('filename.mat', 'X', 'a', 'b')
``` saves variables X, a, and b in filename.mat
Simple I/O in Matlab

Load data in Matlab (.mat) format:

```matlab
>> load filename.mat
```
loads everything within filename.mat into workspace

You can also load specific variables

```matlab
>> load filename.mat X
```
```matlab
>> load(filename.mat', 'X')
```
Only loads variable X
Simple I/O in Matlab

User input
>> age = input('Enter age : ');
>> name = input('Enter name : ', 's');
Simple I/O in Matlab

Screen output

```matlab
>> X
```

We’ve already used this. Often we want it formatted.

`fprintf()` command (borrowed from C)
fprintf()

>> ans = 42

>> fprintf(['Answer to the Ultimate Question' ... 'of Life, the Universe, and Everything' ... 'is %d\n'], ans)

First, what am I doing with the [] and why?
fprintf()

>> ans = 42
>> fprintf(['Answer to the Ultimate Question' ...
          'of Life, the Universe, and Everything' ...
          'is %d\n'], ans)

%d inserts an integer at that location in the string
%f inserts a floating point number
%c inserts a single character
%s inserts a string
fprintf()

>> ans = 42
>> fprintf(['/\n  \'Answer to the Ultimate Question\' …
    \'of Life, the Universe, and Everything\' …
    \'is %d\n\n\n', ans)
fprintf()

What will this print?

```matlab
>> ans = 10
>> str = 'hello'
>> fprintf('Say \'%s\' and give me $%d\n', str, ans)
```

```matlab
ans = 10
str = 'hello'
fprintf('Say '%s' and give me $%d\n', str, ans)
```
fprintf()

What will this print?
>> fprintf('pi = \n', pi)
What will this print?

>> fprintf('pi = %f\n', pi)
What will this print?

>> fprintf('pi = %9.9f\n', pi)
>> fprintf('pi = %12.9f\n', pi)
>> fprintf('pi = %15.9f\n', pi)
>> fprintf('pi = %15.1f\n', pi)
What will this print?

```matlab
>> fprintf('pi = %9.9f\n', pi)
>> fprintf('pi = %12.9f\n', pi)
>> fprintf('pi = %15.9f\n', pi)
>> fprintf('pi = %15.1f\n', pi)

>> ans = 1000*pi
>> fprintf('pi = %4.4f\n', pi)
```
What will this print?

```matlab
>> ans = 12345
>> fprintf('ans = %d\n', ans)
>> fprintf('ans = %10d\n', ans)
```
fprintf()

What will this print?

```latex
\begin{verbatim}
\texttt{ans} = \input{ans}
\texttt{fprintf(\textquoteleft\texttt{ans = \%d\n\text{"}, \texttt{ans})
\texttt{fprintf(\textquoteleft\texttt{ans = \%10d\n\text{"}, \texttt{ans})
\end{verbatim}
```

Please use \texttt{fprintf()} statements where appropriate in your homework assignments