
CircuitLab and Matlab

ECE 211 Circuits I

Lecture #0

Please visit [Bison Academy](#) for corresponding
lecture notes, homework sets, and solutions

Circuit design and analysis is usually a three step process:

- First, you design your circuit on paper.
- Next, you simulate your design to make sure it works.
- Then you build your circuit in lab

The tools we'll be using in this class to do this are:

- Matlab: a tool that makes circuit design and analysis much easier. Think of Matlab as a calculator which can solve 50 equations for 50 unknowns and graph the results.
 - CircuitLab: a very friendly circuit simulator which allows you to build a circuit using drag and drop. Once built, you can find the voltages or run a time simulation to see how the voltages change over time (useful for AC analysis).
 - Breadboards, oscilloscopes, and multimeters: used in lab to allow you to connect components together to build a circuit.
-

CircuitLab (www.CircuitLab.com)

- CircuitLab is a circuit simulator
- Very friendly to use
- Fairly intuitive (can jump right in)
- Multiple tutorials on YouTube are available

Starting Out:

- Create an account
- Use your NDSU email address
 - It's free to NDSU students
 - `student.name@ndsu.edu`
- NDSU ECE pays of university license
 - `student.name@ndus` does *not* work
 - NDUS includes UND (hiss)
- \$34/year for personal license (2026 prices)

CIRCUIT LAB Search My Workbench Electronics Q&A Textbook jacob.glower inbox (0) · settings · log out

Circuit simulation and schematics.

Build and simulate circuits right in your browser.

- Design with our easy-to-use schematic editor.
- Analog & digital circuit simulations in seconds.
- Professional schematic PDFs, wiring diagrams, and plots.
- No installation required! Launch it instantly with [one click](#).

► **Launch CircuitLab** or watch a quick demo video →

Watch on YouTube

Interactive Electronics Textbook NEW

Master the analysis and design of electronic systems with CircuitLab's free, interactive, online electronics textbook.

Open: [Ultimate Electronics: Practical Circuit Design and Analysis](#)

Electronics Questions and Answers
from the CircuitLab Community

Which Courses Use CircuitLab?

Circuit Analysis

- RLC, Sources, Op-Amps
- ECE 111 Intro to ECE
- ECE 211 Circuits I
- ECE 311 Circuits II
- ECE 331 Energy Conversion

Electronics

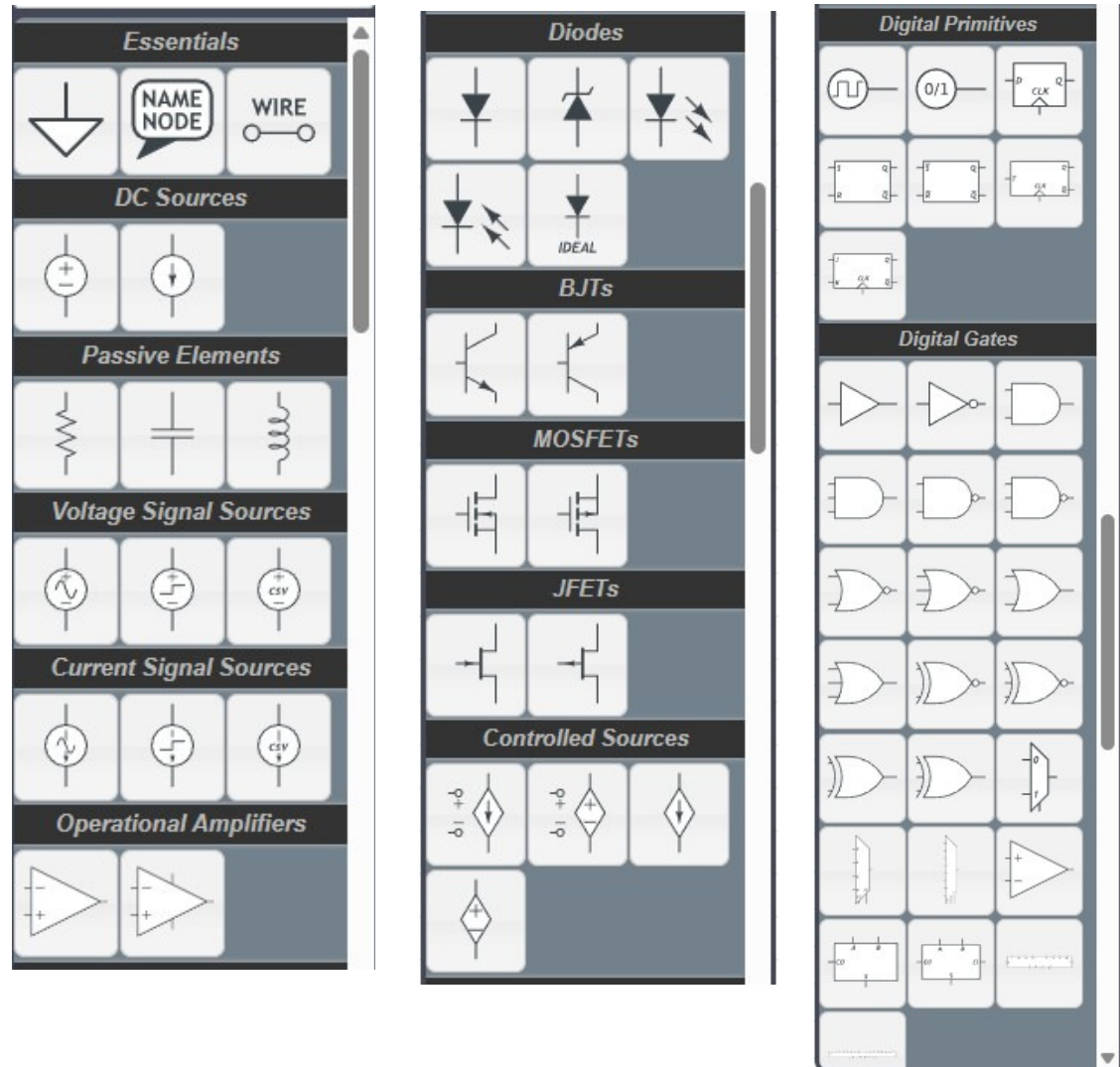
- Diodes / Transistors / Mosfets
- ECE 320 Electronics
- ECE 437 Power Electronics

Digital Systems

- Flip-Flops, Logic Gates
- ECE 270 Digital System Design

Signals

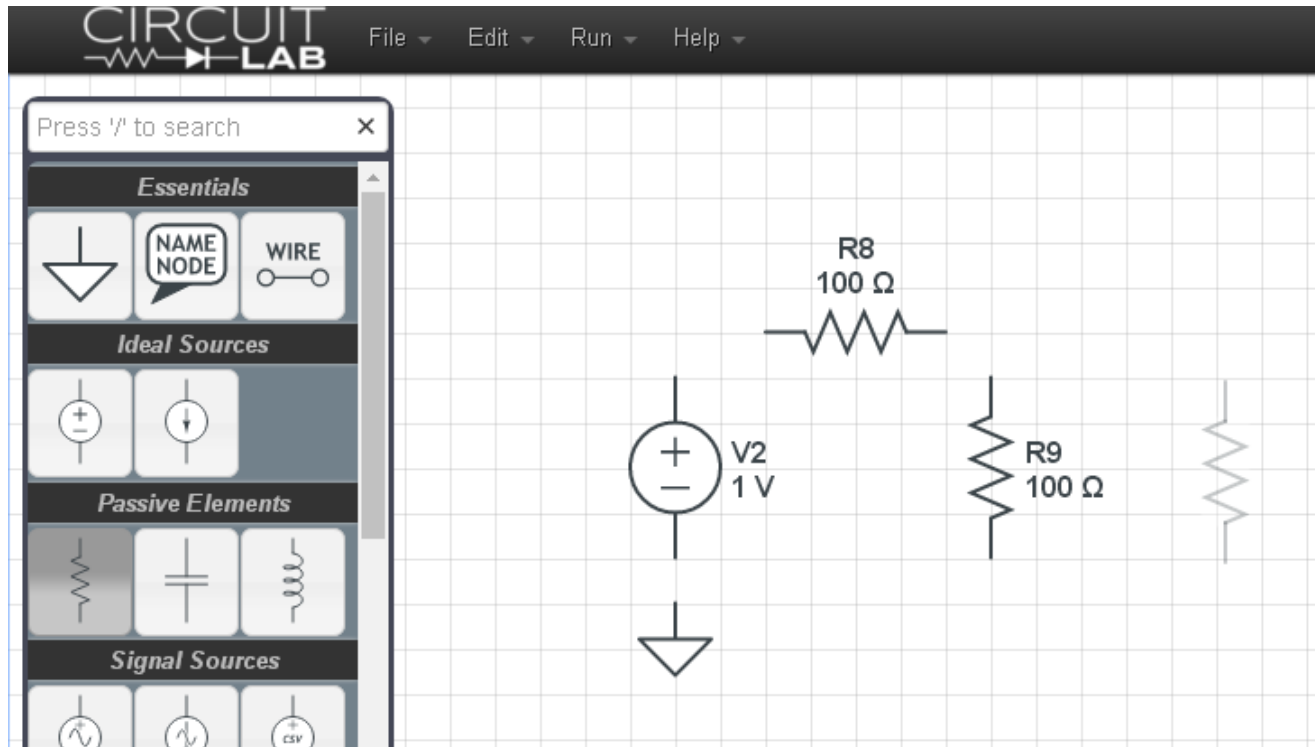
- ECE 343 Signals & Systems
- ECE 461 Controls Systems



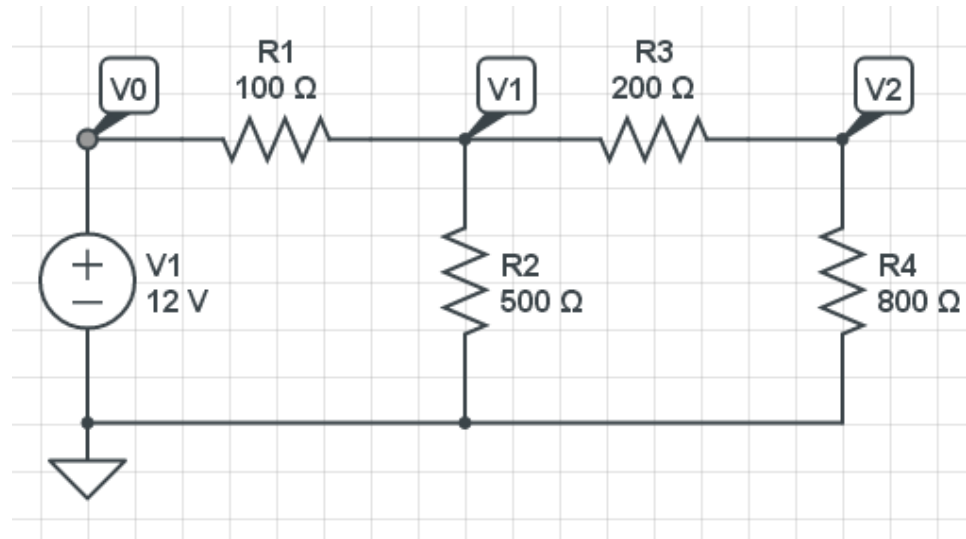
Starting CircuitLab

To build a circuit, drag and drop components from the left (we'll talk about what each one is later in this course)

- R rotates the component
- Double click to change the value

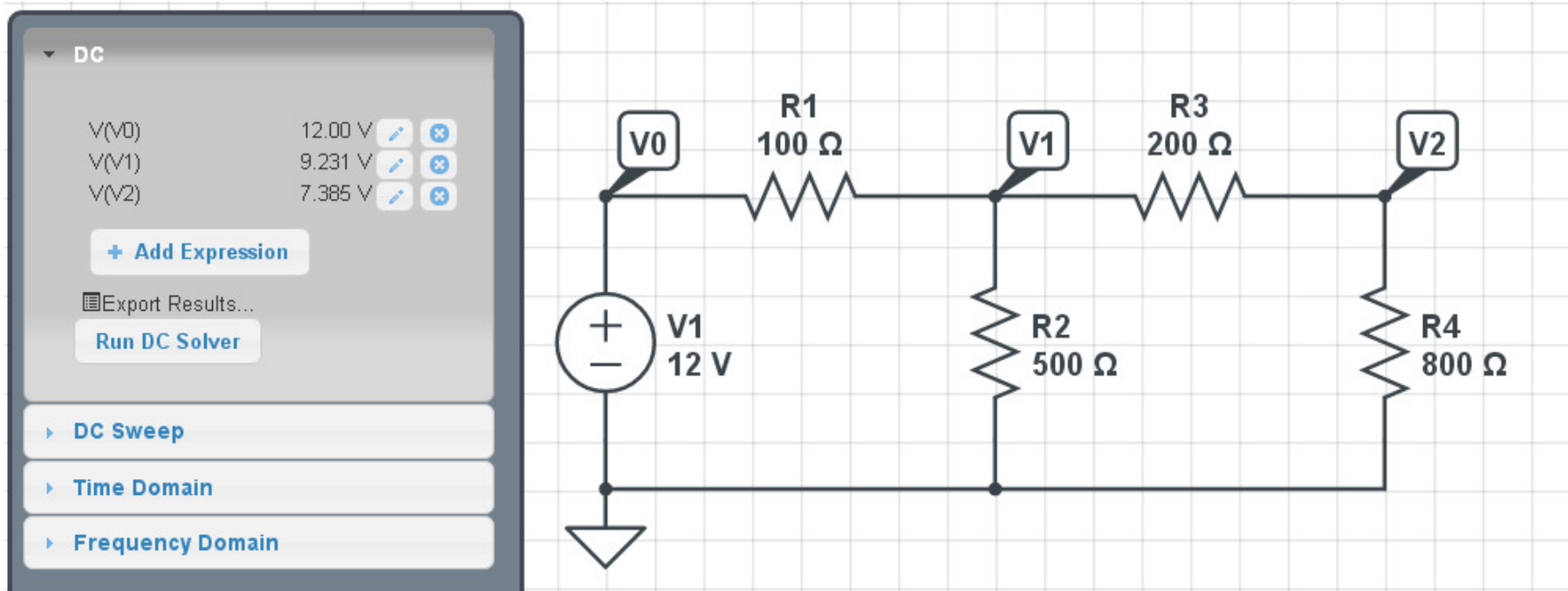


Click and drag to connect the wires together. Adding labels allow you to look at different voltages

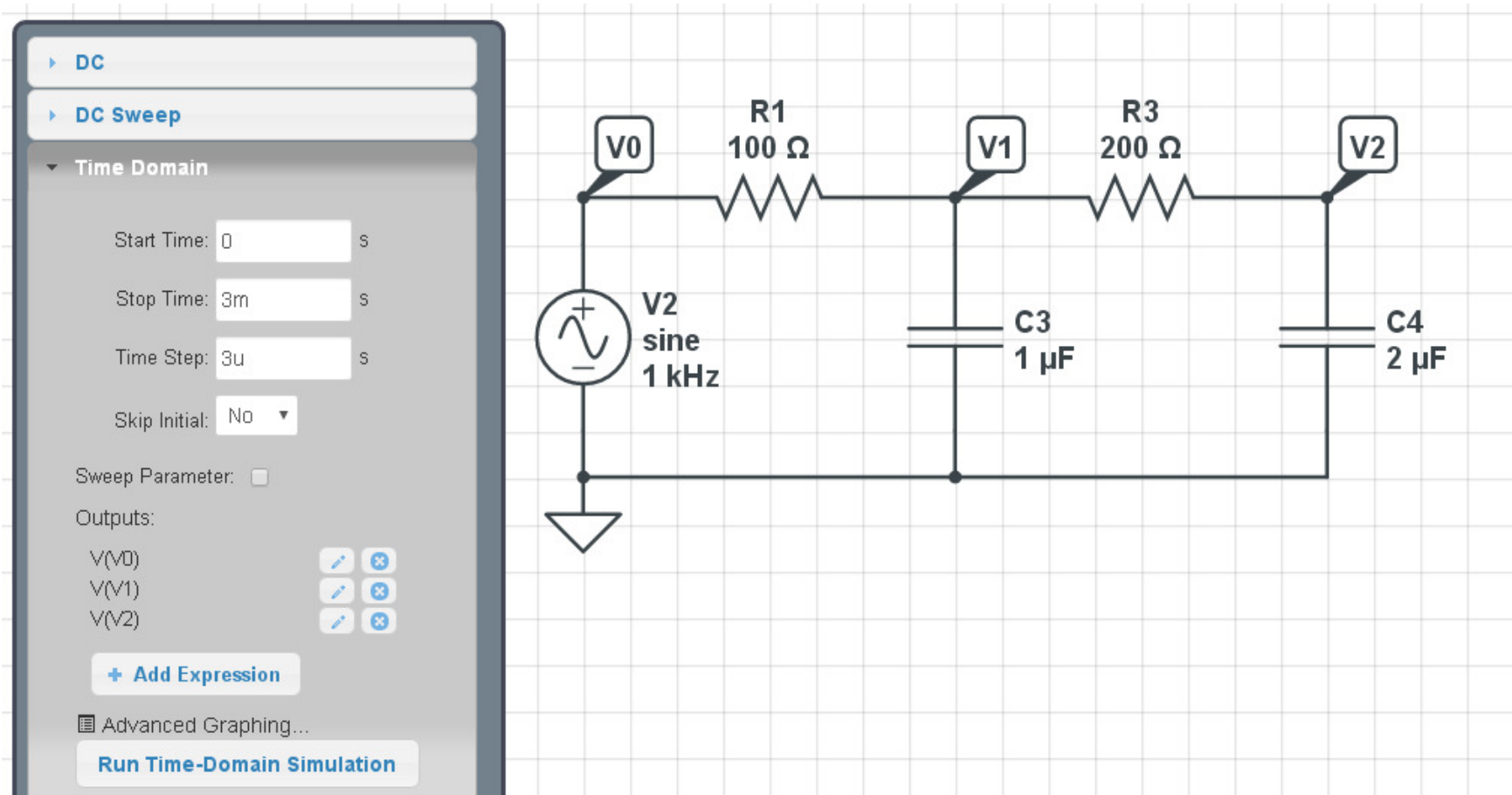


To find the voltages, click on

- Simulate
- + Add Expression (Select V0, V1, and V2)
- Run DC Solver
- Voltages should match what you calculated with Matlab and measured in lab

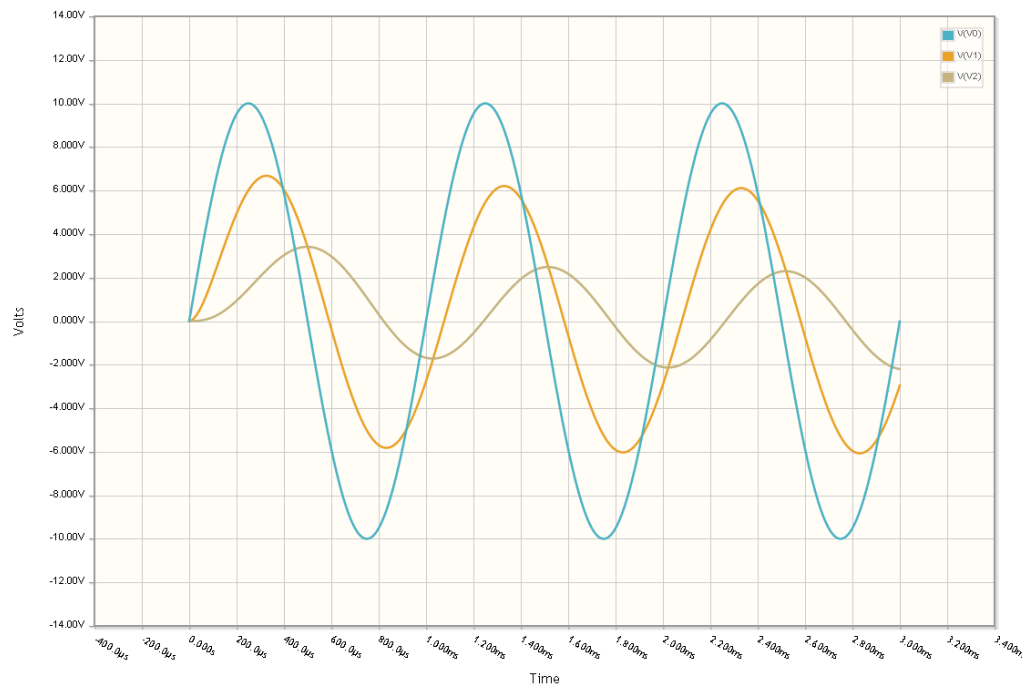


You can also look at transient simulation. Change the source to an AC signal and change R2 and R4 to capacitors



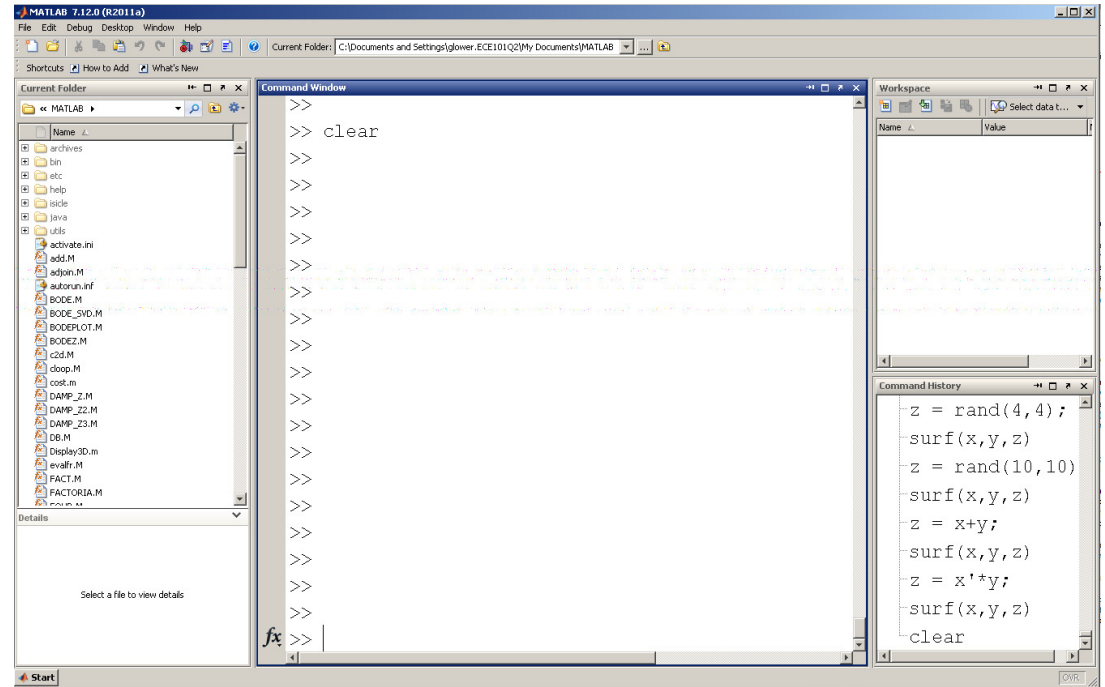
Now run a Timer Domain simulation

- The input is a 1kHz sine wave. Run the simulation for a few cycles (3 cycles or 3ms)
- Make the time step 100 to 1000x smaller (3 μ s gives 1000 points on the following graph). If you make the time step too small, it will take some time to finish the simulation.
- + Add expression to see V0, V1, and V2
- Click on Run Time Domain Simulation
- Should be the same as you measure in lab with an oscilloscope



Becoming familiar with MATLAB

- The console
- The editor
- Scripts
- Functions
- Graphics Window



General environment and the console

When you start up Matlab, you get several windows

I usually close everything doen except the command window.

The command window is like a calculator:

```
>> (2 + 3) * 5
```

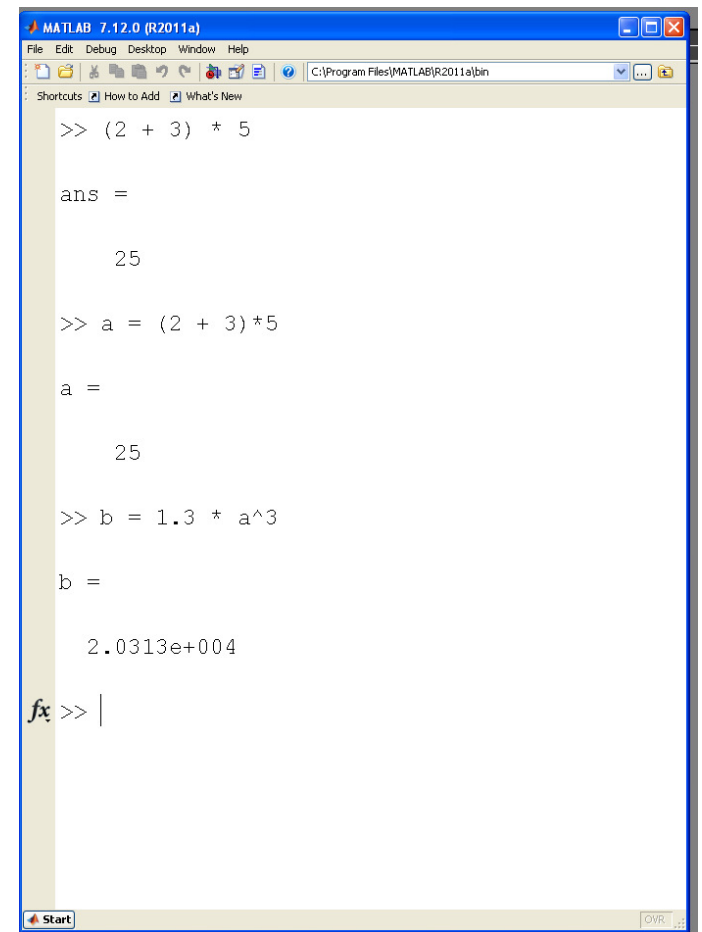
```
ans =    25
```

```
>> a = (2+3) * 5
```

```
a =    25
```

```
>> b = 1.3 * a^3
```

```
b = 2.0313e+004
```



Matrices in Matlab

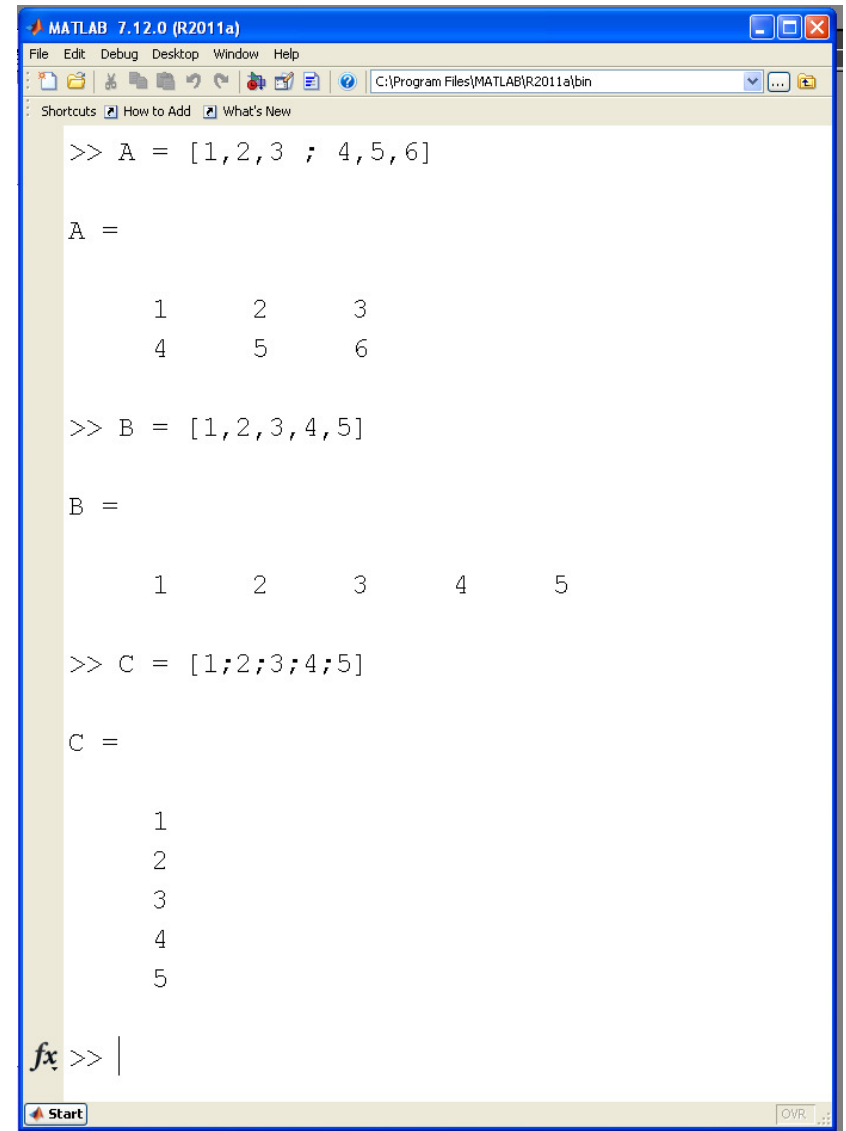
The syntax to input a matrix are:

```
[      start of a matrix
,      next column (a space also works)
;      next row
]      end of matrix.
```

Example:

```
A = [1,2,3 ; 4,5,6]
```

1	2	3
4	5	6



```
MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Program Files\MATLAB\R2011a\bin
Shortcuts How to Add What's New

>> A = [1,2,3 ; 4,5,6]

A =

     1     2     3
     4     5     6

>> B = [1,2,3,4,5]

B =

     1     2     3     4     5

>> C = [1;2;3;4;5]

C =

     1
     2
     3
     4
     5

fx >> |
```

Matrix Operations

Addition & Multiplication:

- Dimensions have to be the same

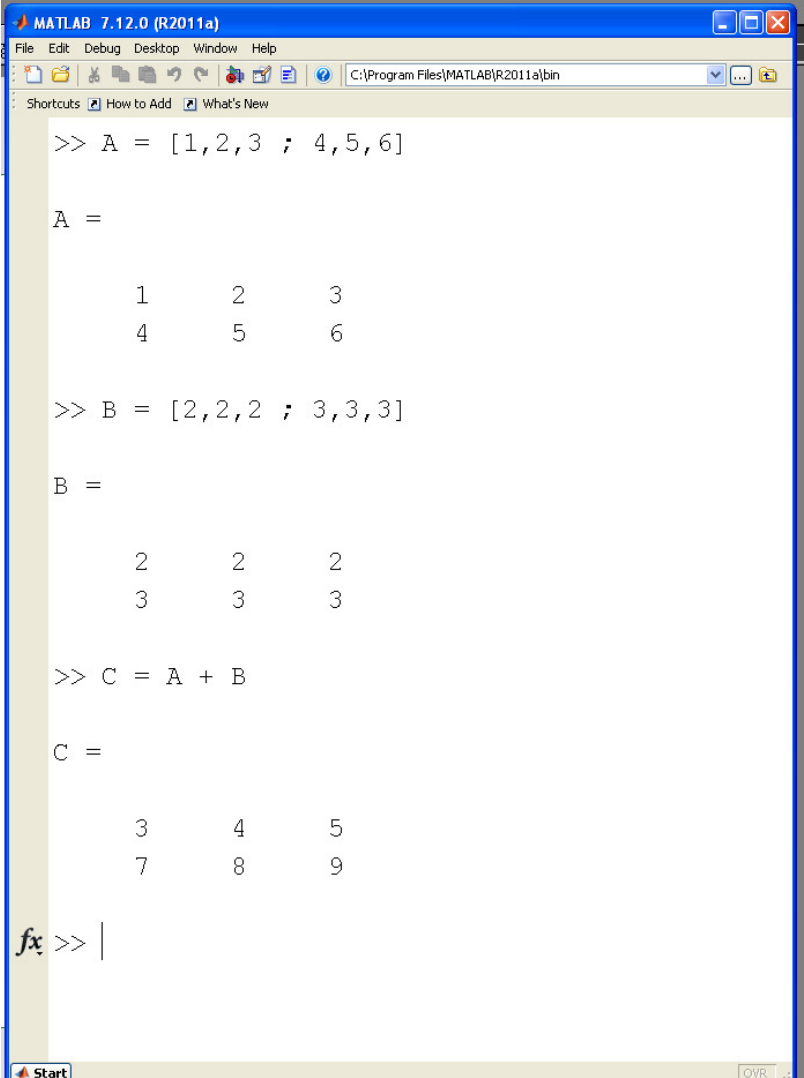
Multiplication:

- Inner dimensions must match

Matrix Inverse

- Needs to be an NxN matrix

Error message if not



The image shows a screenshot of the MATLAB 7.12.0 (R2011a) command window. The window has a blue title bar and a menu bar with options: File, Edit, Debug, Desktop, Window, Help. Below the menu bar is a toolbar with icons for file operations and a command line. The command line shows the following sequence of commands and their outputs:

```
>> A = [1,2,3 ; 4,5,6]

A =

     1     2     3
     4     5     6

>> B = [2,2,2 ; 3,3,3]

B =

     2     2     2
     3     3     3

>> C = A + B

C =

     3     4     5
     7     8     9

fx >> |
```

The window also shows a status bar at the bottom with a 'Start' button and a 'OVR' indicator.

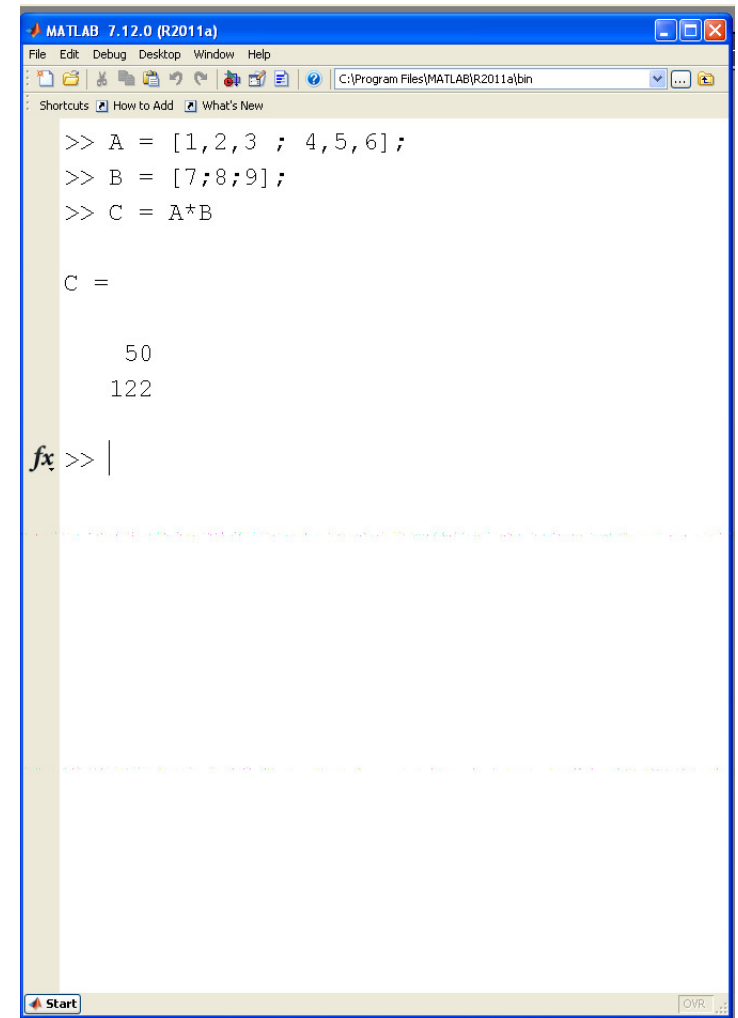
Semi-Colon

- Exclude: Evaluate and display the result
- Include: Evaluate and do *not* display the result

When debugging my code, I usually remove semi-colons

- Check that the code is working correctly

Once it works, insert semi-colons



The image shows a screenshot of the MATLAB 7.12.0 (R2011a) interface. The Command Window displays the following code and its output:

```
>> A = [1,2,3 ; 4,5,6] ;  
>> B = [7;8;9] ;  
>> C = A*B  
  
C =  
  
    50  
    122
```

The code is executed line by line, and the results are displayed in the Command Window. The output for C is a 3x1 column vector with values 50 and 122. The Command Window also shows a prompt for the next line of code, indicating that the execution is ongoing.

Display Settings

```
format short  
pi
```

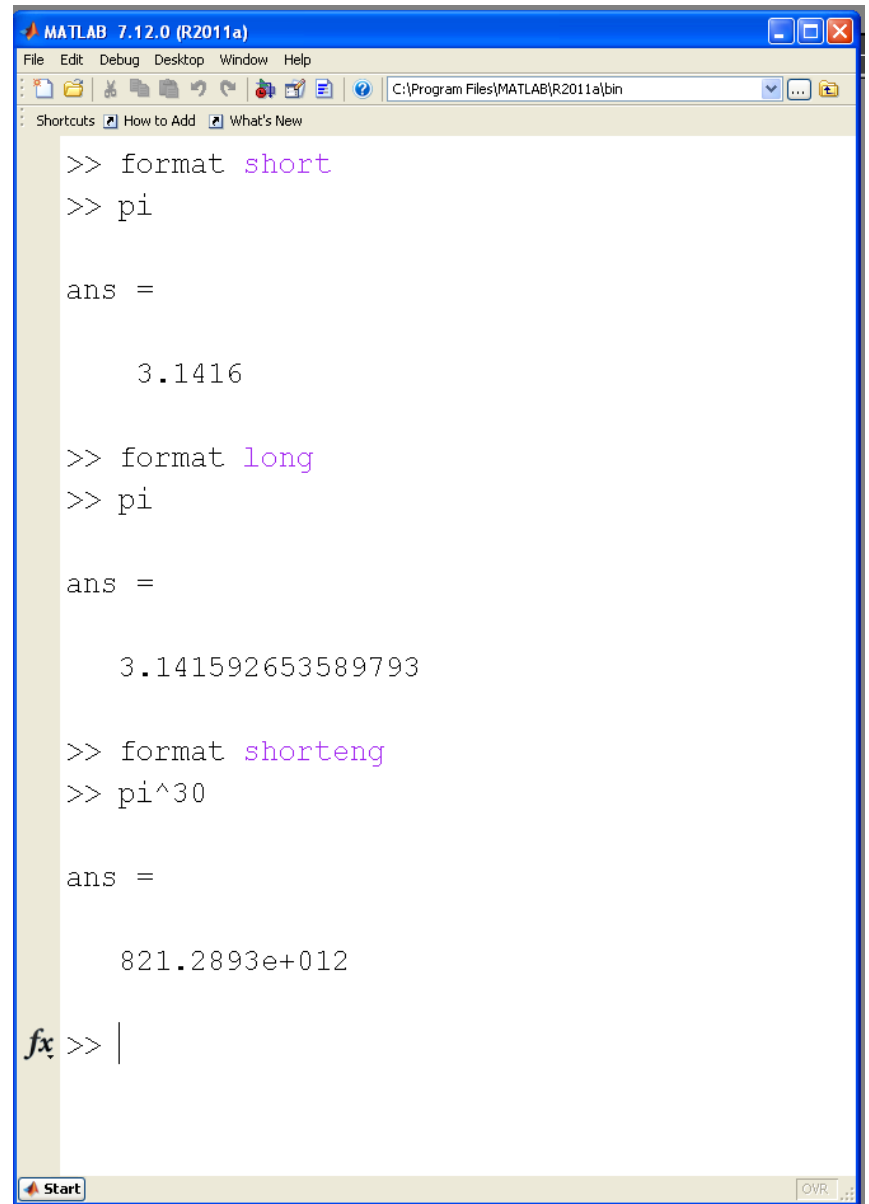
3.1416

```
format long  
pi
```

3.141592653589793

```
format shorteng  
pi^30
```

821.2893e+012

A screenshot of the MATLAB 7.12.0 (R2011a) command window. The window has a blue title bar and a menu bar with File, Edit, Debug, Desktop, Window, and Help. Below the menu bar is a toolbar with various icons. The command window shows the following commands and their outputs:

```
>> format short  
>> pi  
  
ans =  
  
    3.1416  
  
>> format long  
>> pi  
  
ans =  
  
    3.141592653589793  
  
>> format shorteng  
>> pi^30  
  
ans =  
  
    821.2893e+012  
  
fx >> |
```

The window also shows a status bar at the bottom with a Start button and an OVR indicator.

for loops:

```
x = zeros(1,5);  
for i=1:5  
    x(i) = i*i;  
end
```

```
x =      1      4      9     16     25
```

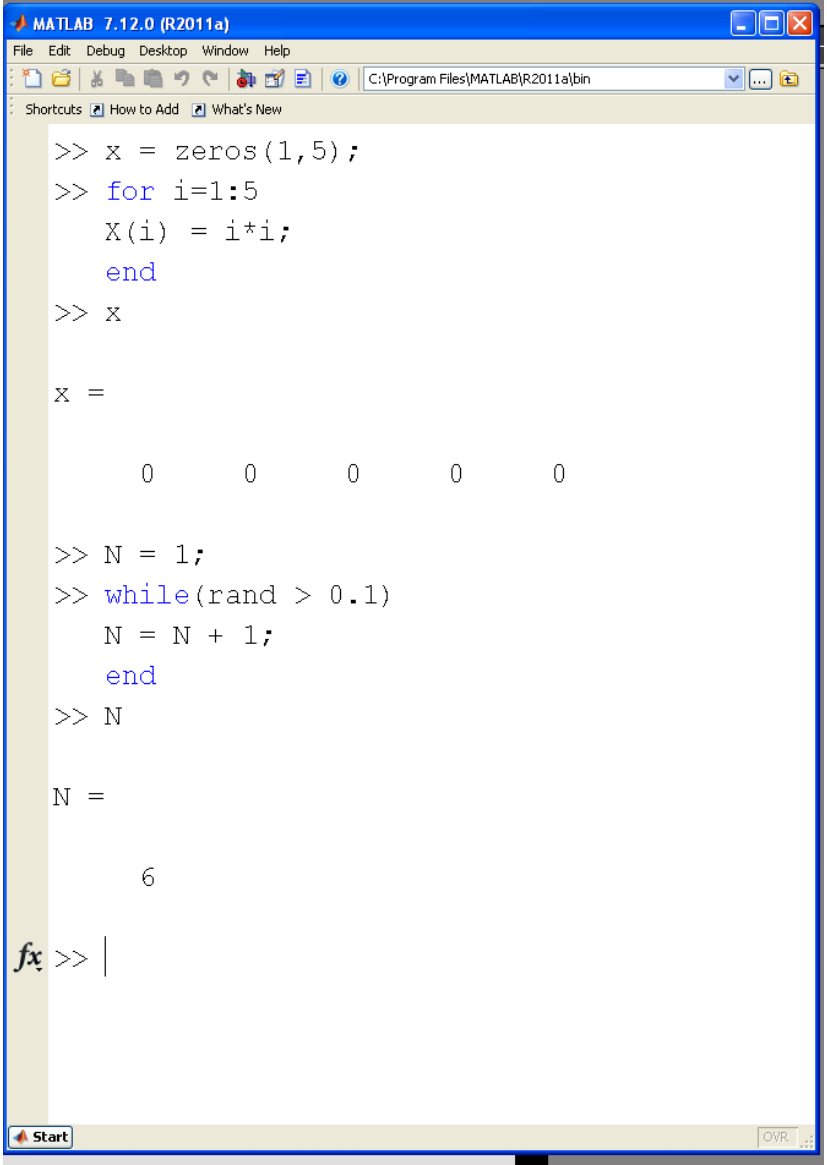
while loop

```
N = 1;  
while(rand > 0.1)  
    N = N + 1;  
end
```

```
N = 6
```

if - if else end

```
if(rand < 0.1)  
    die = 6;  
else  
    die = ceil( 6*rand );  
end
```



The image shows a screenshot of the MATLAB 7.12.0 (R2011a) interface. The window title is "MATLAB 7.12.0 (R2011a)". The menu bar includes File, Edit, Debug, Desktop, Window, and Help. The toolbar shows various icons for file operations and debugging. The command window displays the following code and its output:

```
>> x = zeros(1,5);  
>> for i=1:5  
    X(i) = i*i;  
end  
>> x  
  
x =  
  
      0      0      0      0      0  
  
>> N = 1;  
>> while(rand > 0.1)  
    N = N + 1;  
end  
>> N  
  
N =  
  
      6  
  
fx >> |
```

Scripts and Functions in Matlab

Matlab is also a programming language. This allows you to run the same code over and over without having to constantly retyping in the code over and over again.

- Scripts are like code you type in the command window. When you run a script, it is like you just typed in the code in the script in the command window
- Functions are subroutines. They allow you to create new commands in Matlab which can be called by other new functions and so on.

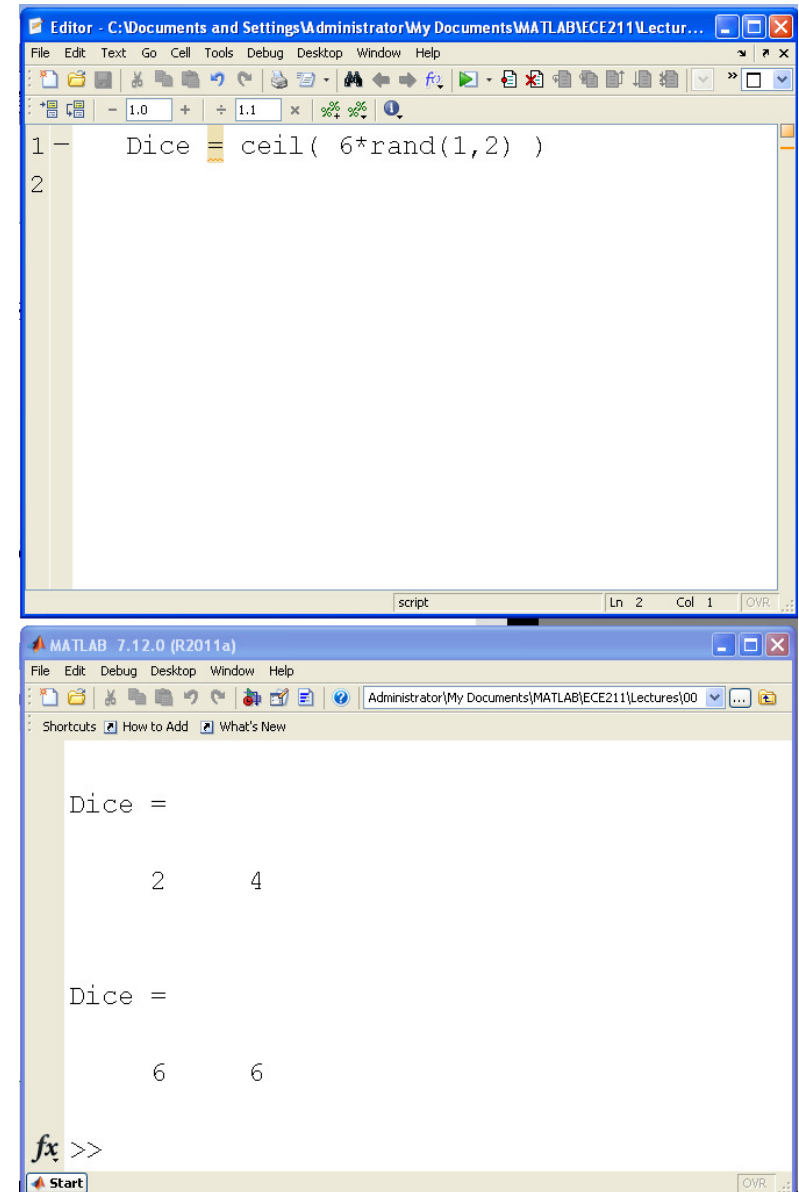
This is why Matlab is used by industry so much: you can create a library of functions and scripts which allow you to design your product.

Scripts:

Suppose you want to see the distribution of rolling two six-sided dice (2d6) 100 times. This is where scripts shine.

- Start with a New Script.
- Get the code to produce 2 random numbers
- Click on the green arrow to run the script (F5)

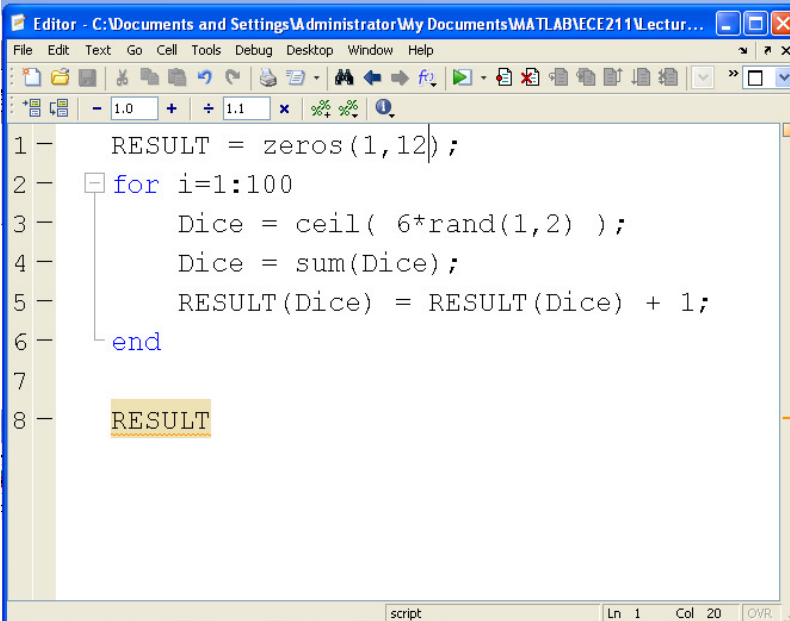
note: Save the script in the Matlab directory. If you save it somewhere else on your computer, Matlab won't be able to find it.



Once you are convinced the script works,

- Get it to sum the two dice
- Then repeat 100 times

This determines the frequency of rolling each number when rolling the dice 100 times



The top window is the MATLAB Editor, showing a script named 'script'. The script initializes a 1x12 array 'RESULT' to zeros. It then enters a 'for' loop from i=1 to 100. Inside the loop, it generates two random numbers between 1 and 2 using 'ceil(6*rand(1,2))', sums them, and increments the corresponding index in 'RESULT'. The script ends with 'end' and 'RESULT'.

```
1 RESULT = zeros(1,12);
2 for i=1:100
3     Dice = ceil( 6*rand(1,2) );
4     Dice = sum(Dice);
5     RESULT(Dice) = RESULT(Dice) + 1;
6 end
7
8 RESULT
```

The bottom window is the MATLAB Command Window, showing the output of the script. It displays the 'RESULT' array as a 1x12 vector. The values are: 0, 2, 7, 9, 19, 11, 16, 12, 7, 8, 6, 3.

```
MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
Administrator\My Documents\MATLAB\ECE211\Lectures\00
Shortcuts How to Add What's New

RESULT =

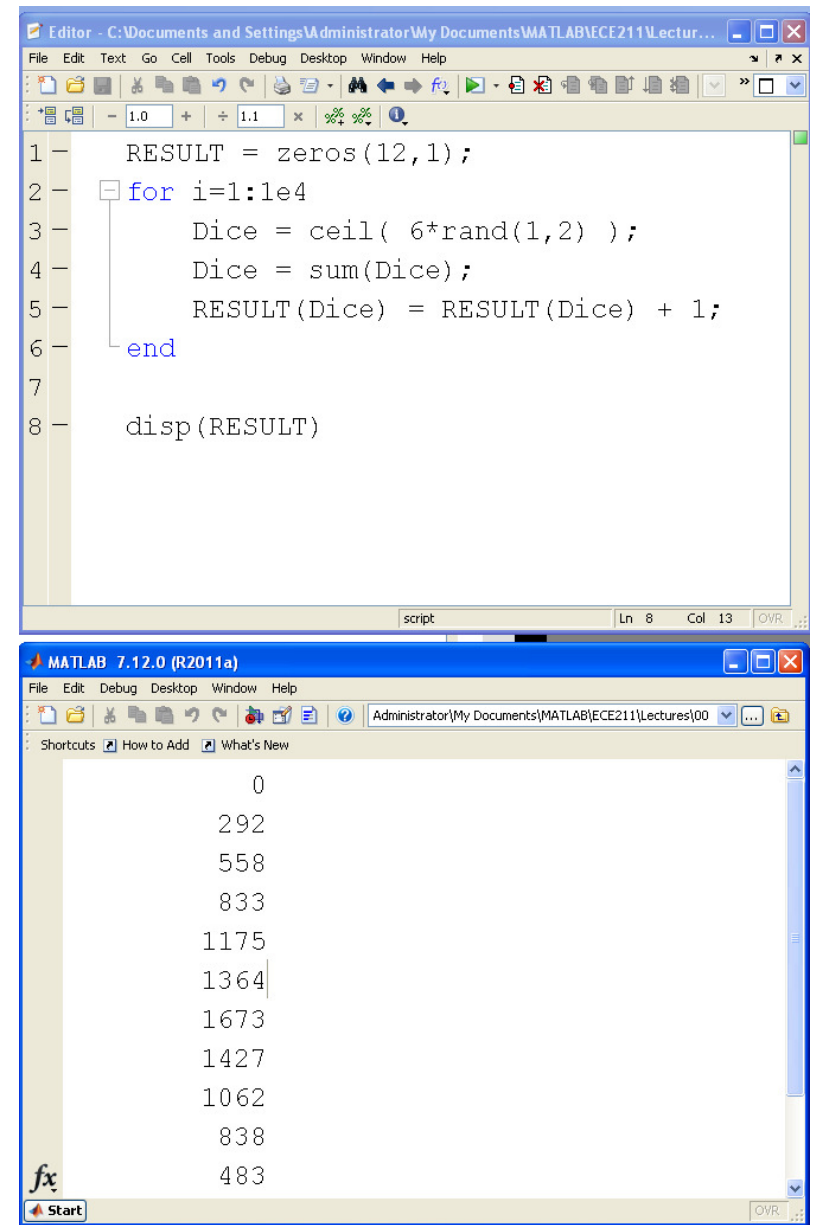
Columns 1 through 7
0      2      7      9     19     11     16

Columns 8 through 12
12      7      8      6      3

fx >>
Start
```

The beauty of scripts is that

- If you want to see what happens if you run it a second time, you can just by clicking the green arrow again.
- If you want to roll the dice 1000 times instead, change one line of code (`i = 1:100` becomes `1:1000`)



The image shows two windows from the MATLAB 7.12.0 (R2011a) environment. The top window is the MATLAB Editor, displaying a script named 'script.m'. The script initializes a 12x1 zero vector 'RESULT', then enters a 'for' loop from `i=1` to `1e4` (10,000). Inside the loop, it generates a random number between 1 and 2, multiplies by 6, and uses `ceil` to get a value between 1 and 12. This value is added to the corresponding index in 'RESULT'. After the loop, the final 'RESULT' vector is displayed. The bottom window is the MATLAB Command Window, showing the output of the script: a 12x1 column vector of counts for each face of the die. The values are: 0, 292, 558, 833, 1175, 1364, 1673, 1427, 1062, 838, and 483. The 11th element is missing, indicating a count of 0 for that face.

```
1 RESULT = zeros(12,1);
2 for i=1:1e4
3     Dice = ceil( 6*rand(1,2) );
4     Dice = sum(Dice);
5     RESULT(Dice) = RESULT(Dice) + 1;
6 end
7
8 disp(RESULT)
```

```

0
292
558
833
1175
1364
1673
1427
1062
838
483
```

Functions

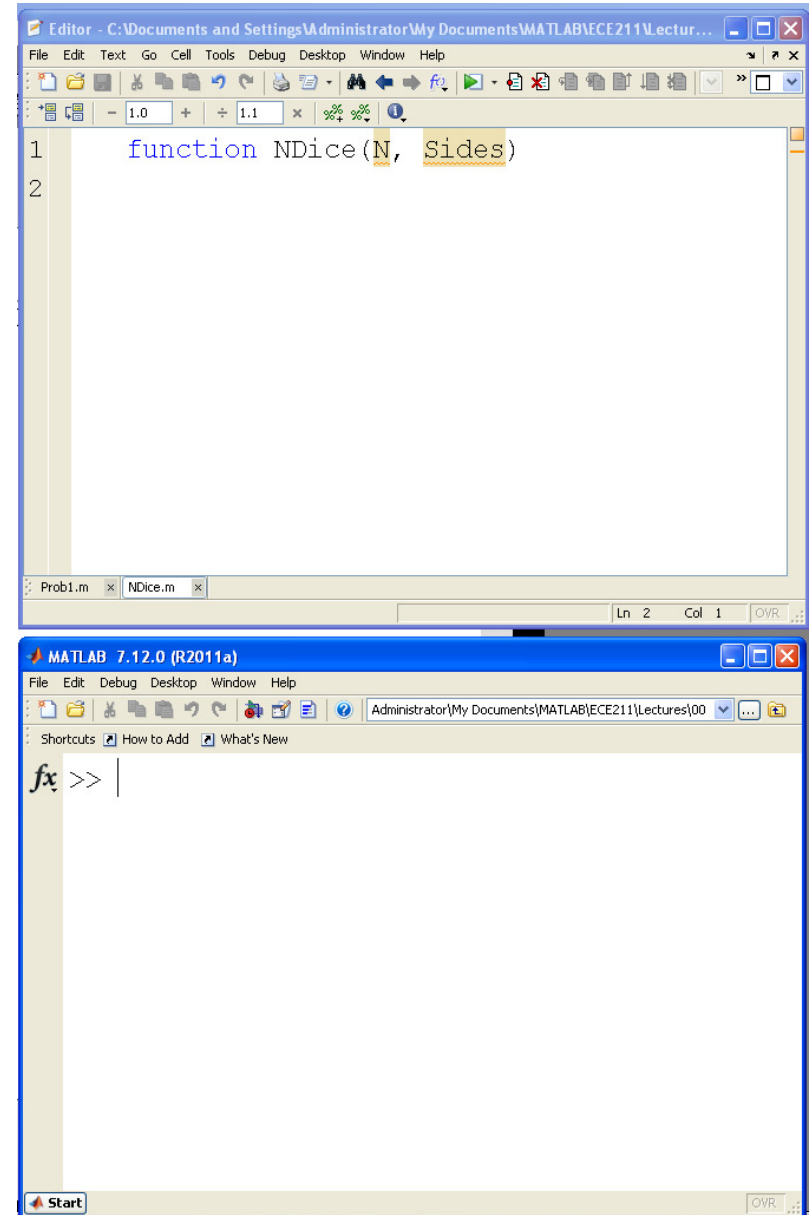
A function is a subroutine. It allows you to create new functions in Matlab that can be used over and over again.

For example, let's create a function called NDice. It will be passed two parameters

- N: The number of dice
- Sides: The range of numbers (1..6, 1..12, etc.)

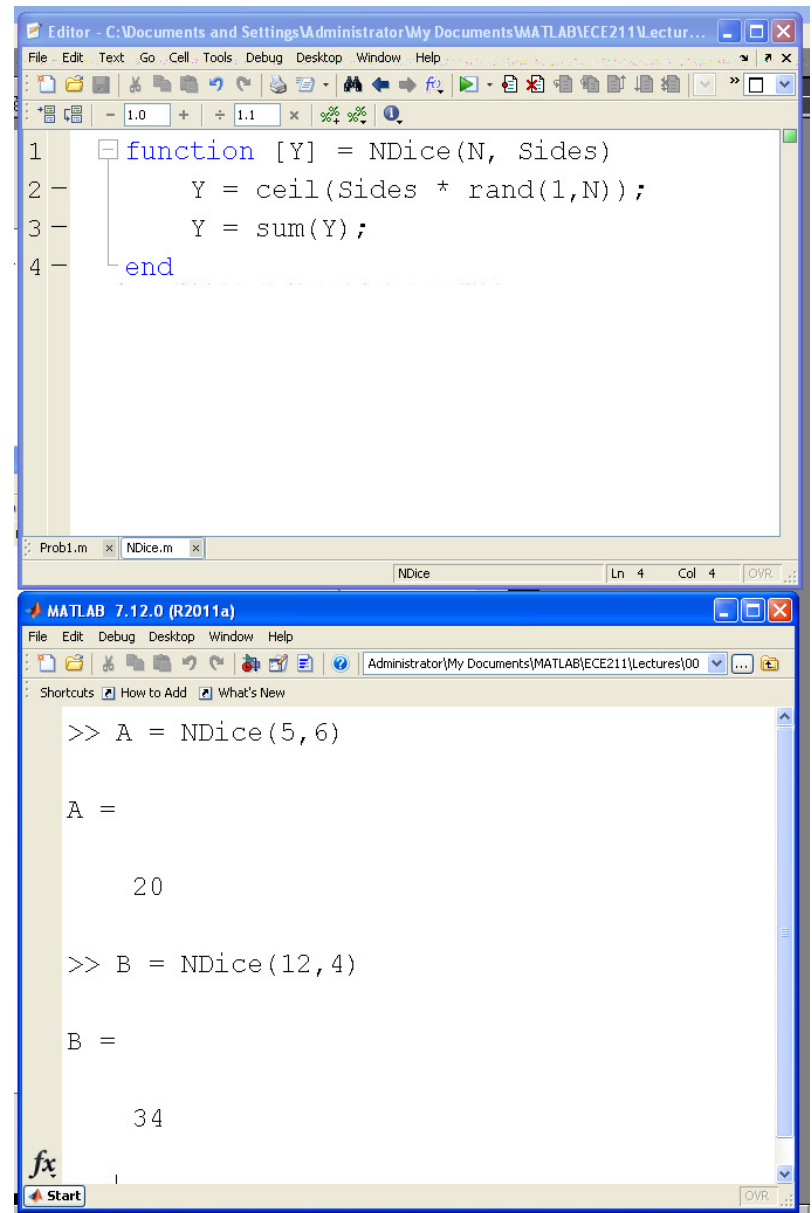
In Matlab,

- click on New Function
- Create the subroutine
- Save with the same name as the function



With that you just created a new function in Matlab called NDice. From the command window, you can

- Find the sum of rolling 5d6 (total is 18)
- Find the sum of rolling 12d4 (total is 26)



```
Editor - C:\Documents and Settings\Administrator\My Documents\MATLAB\ECE211\Lectur...
File Edit Text Go Cell Tools Debug Desktop Window Help
1 function [Y] = NDice(N, Sides)
2     Y = ceil(Sides * rand(1,N));
3     Y = sum(Y);
4 end

MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
Administrator\My Documents\MATLAB\ECE211\Lectures\00
Shortcuts How to Add What's New
>> A = NDice(5,6)

A =

    20

>> B = NDice(12,4)

B =

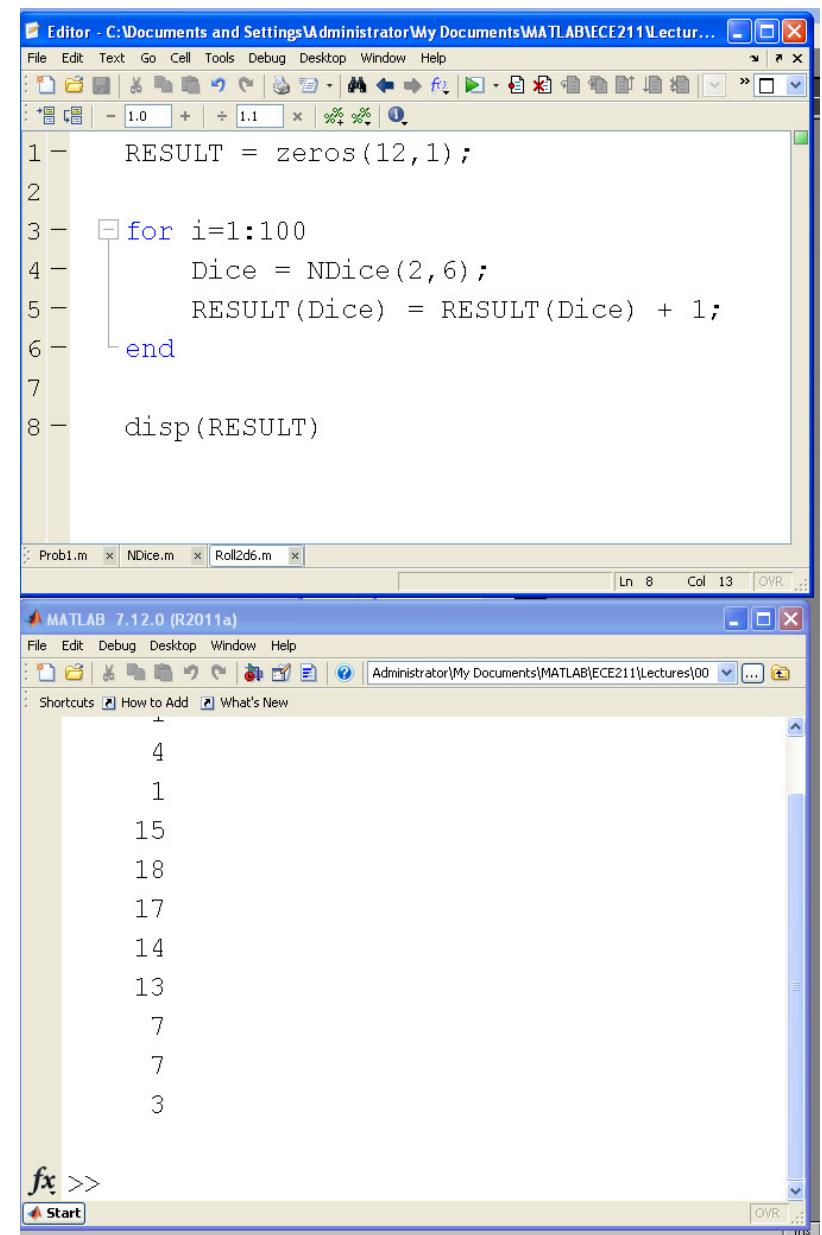
    34
```

Nested Subroutines

- Subroutines can use other subroutines

Now that you have this function, you can clean up the script

- Note that since this script uses a function we just create, other people won't be able to run this script unless they too create the function NDice.



The image shows two windows from the MATLAB 7.12.0 (R2011a) environment. The top window is the Editor, showing a script with the following code:

```
1  RESULT = zeros(12,1);  
2  
3  for i=1:100  
4      Dice = NDice(2,6);  
5      RESULT(Dice) = RESULT(Dice) + 1;  
6  end  
7  
8  disp(RESULT)
```

The bottom window is the Command Window, showing the output of the script:

```
4  
1  
15  
18  
17  
14  
13  
7  
7  
3
```

The Command Window also shows the MATLAB prompt `fx >>` and a `Start` button at the bottom.

Plotting Functions in Matlab:

Matlab has some pretty good graphics capabilities.

Matlab Plot Command	x axis	y axis	type of function
plot(x,y)	linear	linear	$y = ax + b$
semilogx(x,y)	log()	linear	$y = a \cdot e^{bx}$
semilogy(x,y)	linear	log()	$y = a + b \ln(x)$
loglog(x,y)	log()	log()	$y = a \cdot b^x$
subplot(abc)	Create 'a' rows, 'b' columns of graphs. Starting at #c		

For example, plot the function

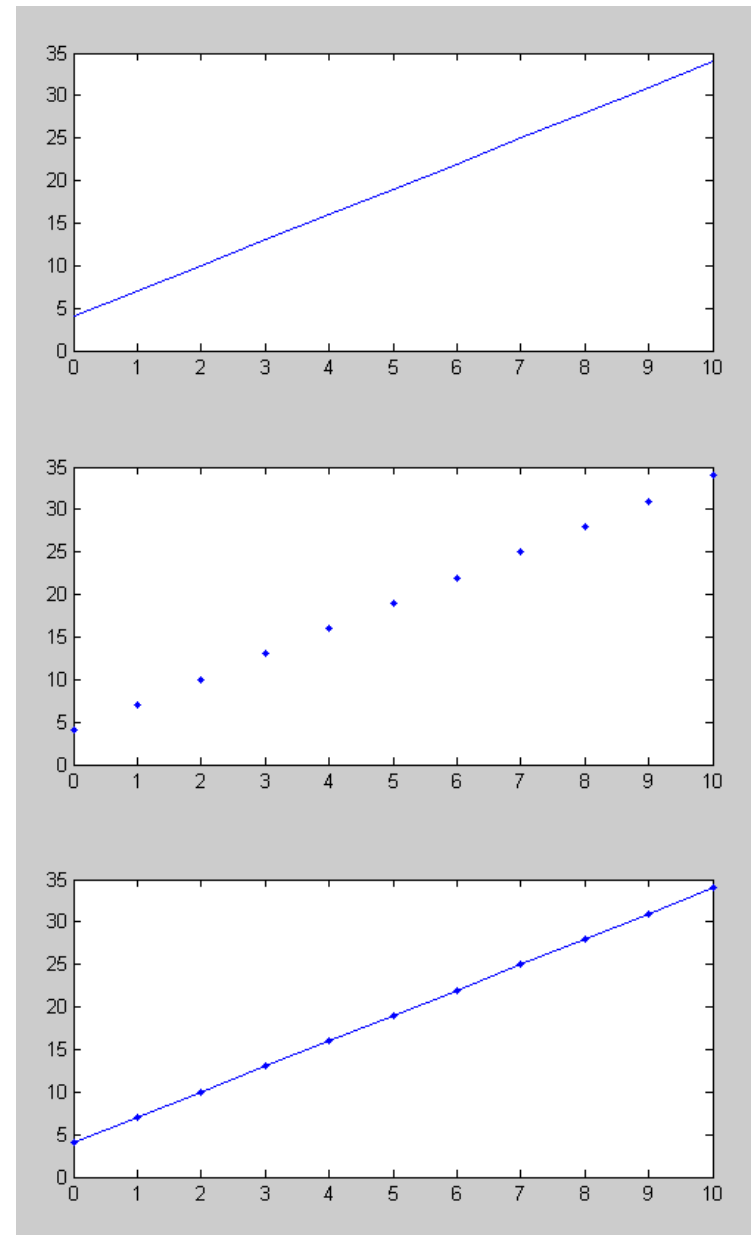
$$y = 3x + 4$$

```
x = [0:1:10]';  
y = 3*x + 4;
```

```
% connect the points with a line  
subplot(311)  
plot(x,y);
```

```
% plot a dot at each point  
subplot(312)  
plot(x,y, '.');
```

```
% connect the points and add a dot  
subplot(313);  
plot(x,y, '-.');
```



Snipping Tool & Homework Sets

A good way to do the homework in this class is to use MS Word

- Copy and paste your Matlab code in to MS Word
- Delete your mistakes (makes it look like the homework was easy)

For figures, use Snipping Tool

- From Windows, search for Snipping
- Run the snipping tool
- Select the figure you want to use
- Copy and paste it into your MS Word document

When done, submit your Word document on BlackBoard

- .doc file
 - pdf file also works
-

Summary:

CircuitLab is a very friendly circuit simulator

- Drag and Drop
- Easy way to check your analysis in this and other classes
- Free for all NDSU students

Matlab is a really useful tool

- You'll be using it in many courses at NDSU
- You'll probably use it when you go to industry
- Think of it as a calculator on steroids
- Free for all NDSU students

We'll be using both a lot in Circuits I

- If your calculations match CircuitLab's results, you probably did the problem correctly
-