Lab 1
CSE 7 Winter 2018
This lab is an introduction to the Matlab program environment.

LEARNING OBJECTIVES:
1) Acclimate yourself to the MATLAB program environment and Command Window
2) Interact with the interpreter using simple function calls with arguments, and variable assignments
3) Create different types of variables
4) Learn all the vocabulary words italicized above
5) Experience and recover from various error conditions

REMINDER: You must complete this Lab#1 assignment without the assistance of any person except course TA and Tutors and you must complete this assignment without looking at other student’s code or copying solutions from any source.

Homework and Lab assignments will be individual programming.

HELP: If at any point you get lost or stuck, go back to the beginning of the section and try again, then please ask the course TA/Tutor in your lab section. If you need help on the homework, please ask the TA/Tutors during your enrolled lab hours, homework help hours, office hours, piazza forums, or cs7w@acsmail.ucsd.edu email.

Learning the systematic process expert programmers follow to un-stuck themselves is one of the most essential parts of what you should learn in this course! Learning to program isn’t just about learning function names and syntax—it is learning how to debug, how to tutor yourself on the fly, and how to make gradual progress in the face of things that seem hopelessly and mysteriously broken.

Description:
This assignment consists of following a series of instructions and reporting on outcomes. It is different from future assignments in that you WILL NOT turn in your code, only reporting the outcomes of it.

Lab Instructions:

PART ONE: LOGIN to WINDOWS 7
1) Find your course specific account name, cs7wXX:  https://sdacs.ucsd.edu/~icc/index.php
2) Remember your cs7wXX login name, e.g. “cs7waa”.
3) Use Global Password Change Tool to set cs7wXX password by 1/15/18 (takes 30 minutes)
   http://acms.ucsd.edu/students/gpasswd.html

Note: If login screen shows Linux Cent OS, select restart from login menu under “ShutDown”, then choose the option of “Windows” for these “dual-boot” ETS/ACMS machines.
1) Press <CTRL><ALT><DELETE> to display Windows7 login window.
2) In login window, type in your CSE 7 login account name. i.e. cs7wXX
   - substitute the “XX” for your unique login name
PART TWO: LOGIN to cs7wXX ACCOUNT

1) <CLICK> on Windows 7 icon (Start) in lower left corner
   a) <CLICK> on right topmost “cse 7 Student Account”
   b) <CLICK> on “MyDocuments” folder
   (This is your cs7wXX home directory)

2) Keep this window open to one side of your monitor screen workspace.

PART THREE: NOTEPAD++ for RECORDING RESPONSES

Nelson Honor A01234567
Sec Tu 9am
LAB #1

#1: The expression with single quotes will have an answer ...
#2: YES
#3: >> ‘1+2’
ans =
1+2
The program takes the expression in quotes as ...

e.g. Notepad++ file named “Lab1_Honor”

PART THREE: NOTEPAD++ for RECORDING RESPONSES (continued)

1) Click on the Start menu on the bottom left side of your screen, choose All Programs, and drag the Notepad++ icon from the pop up menu onto the Taskbar on the bottom of your screen.
2) Click on the Notepad++ icon pinned to your Taskbar to open a Notepad++ document to record your responses for this Lab.
3) You will report the outcomes of some experiments in this document. Then you will show the entire document to a course TA/Tutor in order to get checked off. To save your document as you proceed with the lab, choose in the Menu Bar “File | Save As”. In the last Text Field, labeled “Saved type” choose: “Normal Text File (txt)”
4) In the top Text Field, labeled “Save in”, navigate to your cs7wXX home directory on ieng6.ucsd.edu. (MyDocuments or choose “MyDocuments” in drop down list)
5) In the “File Name” Text Field at the bottom, name your lab report “Lab1_Lastname”. Inside the document, at the top, put your full name and PID. Please format your document clearly. Clearly label which question you are answering. e.g. “#1” You must complete this assignment by the end of lab in order to get lab credit.
6) Keep this 2nd window open to one side of your monitor screen workspace.

PART FOUR: STARTING MATLAB

1) Click on the Start menu on the bottom left side of your screen, choose All Programs, and drag the MATLAB R2017a icon from the pop up menu onto the Taskbar on the bottom of your screen.
2) Click on the MATLAB icon pinned to your Taskbar to launch the MATLAB programming environment.
3) You should now be looking at a MATLAB window like this:
PART FIVE: ACCLIMATING TO MATLAB
The "Command Window" (large space in the middle) is where you will type in commands (code) for this assignment.

MATLAB will immediately perform the tasks you command as soon as you hit return at the end of each command. This interactive environment for coding is called an interpreter (remember that vocabulary word).

The great thing about an interpreter is that it allows you to quickly try things out—experiment—and immediately see the results, just like doing lots of tiny science experiments. I like to think of interpreters as a bit like the popular Will it Blend? videos on YouTube (go ahead, watch one if you want, you have enough time).

There are three steps to a 'Will it Blend?' video:
1) Wondering in your mind what it would look like when a given item is thrown in the blender—and mentally making a prediction,
2) Actually throwing the item into the blender,
3) Immediately observing the outcome. The appeal of the 'Will it Blend?' videos is that you take a question that some people might just idly wonder about, like, "I wonder what would happen if you put an iPhone in a blender...hmm...hmm..." and they actually do it rather than just wondering!

The interpreter allows you to get instant gratification answers to your MATLAB questions in much the same way: you can quickly experiment by typing individual lines into the interpreter [blender]. Just like ‘Will it Blend?’ doesn't fret about breaking things, I want you to develop a fearlessness about trying things in MATLAB as you build your programs. The rest of the assignment consists of experiments to try.

PART SIX: OPERATORS and EXPRESSIONS
Click in the Command Window. You should see a cursor next to the ">>" at the top of the Command Window. That is where you type things, called commands or expressions. Type the expression "2+2" and hit <ENTER>.

- Maybe you aren't sure if that means I want you to type "4+4" with the quotes, or without the quotes. Good question! Try it both ways! (Seriously, try it both ways, even if you think you know which is the 'right' way.) Will it blend?
- What do you think will happen when you type it with single quotes, like this: ' 4+4 '?
- BEFORE you try it, make a prediction and enter your prediction as QUESTION #1 in your Notepad++ document.
- After you have entered your prediction, go ahead and type ' 4+4 ' and report what happens as QUESTION #2 (yes/no was your prediction correct?) and
QUESTION #3 (Explain what the outcome was when you typed '2+2' (with single quotes). You may copy and paste the relevant text from the MATLAB command window into this form, but be sure to also explain in your own words what happened. Comment briefly on WHY you think that happened, and why it did or didn't match your prediction.).

- What happens if you hit <ENTER> before you are done typing your expression? Try hitting <ENTER> too soon, in other words, like this: 2+<ENTER> and 2<ENTER>. Record why the result of those two experiments was the same or different.

- For your information: operators in MATLAB include: +, -, *, /, \ and ^ (for addition, subtraction, multiplication, division, divided into, and exponent, respectively). Know these! Try a few if you want—in particular, you might be curious how exponent works.

PART SEVEN: FUNCTIONS
Now let's try using some of the many useful functions that are built into MATLAB. Please calculate the following:

- `sqrt(25)`  
  --- MATLAB can do square root

- `sin(0.5 * pi)`  
  --- MATLAB can do sine and cosine, and has conveniently made the value pi available for you to use

- `factor(60)`  
  --- MATLAB can provide the prime factorization of a number

  - What do you think the output will be when you factor a number like 16, or 20, where there is a prime factor that appears several times in the prime factorization? Note that 16 has only one prime factor: 2.

  Do you think MATLAB will output just '2' or will it also try to say that there are four 2's when you factor 16? How would it format that information? Try it and see what happens.

  - What do you think the output will be when you try to factor a number that is not an integer? Report the outcome in QUESTION #4.

- `gcd(10,25)`  
  --- Some MATLAB functions take more than one input, or argument. This function finds the greatest common divisor of two integers.

  - What happens if you only provide one argument? Report the outcome in QUESTION #5.

  - You should be seeing a pattern in the error messages you reported in QUESTIONs #4 and #5. Errors that happen inside functions generate error messages with a format that is slightly different from the other types of errors we saw earlier. Analyze the pattern and report your observations in QUESTION #6.

- You can try an example of "accidentally" hitting <ENTER> too soon with functions as well: what happens when you type `factor(16<ENTER>` or `factor(<ENTER)`? Does this error message have...
the function style format you just analyzed in lab question #6, or does it look more like the previous errors? Why?

- **NOTE:** *Why am I having you generate so many errors??* I want you to know what errors look like, how they are formatted, what kinds of information is contained in them, etc. There are two reasons you should care about this: (1) It will be on the midterm. (2) Encountering all sorts of different error messages now, when the stakes are very low and you are actually doing it on purpose, will help you when you are debugging more complicated programs. Continuing on:

**PART EIGHT: DOCUMENTATION**

Important things to note for future reference in using MATLAB's provided functions:

- To see a list of all functions, click on the "fx" next to the ">>" where you type your commands in the Command Window:

![MATLAB Command Window](image1)

- When you click on the fx, you will see a menu of all of MATLAB's provided functions. For each function, there is helpful documentation explaining how to use the function. Find the gcd and factor functions we have used by navigating to Mathematics->Elementary Math->Discrete Math, like this:

![MATLAB Documentation Menu](image2)
• Take a moment to read the description for one of the functions just to get a feel for the writing style. Sometimes the way function documentation is written is not terribly easy for new programmers to understand, but over time you will come to value this documentation and find yourself referring to it quite often.

• Another way to access the documentation, without clicking on fx, is to just begin typing a function. Type factor( and then pause without hitting <ENTER>. You should see a quick hint pop up, and an option to access the complete function documentation for factor.

PART NINE: MORE EXPRESSIONS

Now we'll work on combining operators and functions into more complex expressions:

- factor(gcd(200,100))  <- you can put one function inside another
- factor(gcd(20*10,100))  <- you can combine operators and functions
- factor(gcd(20*10,100))/2  <- Here's an interesting variation of the expression you just typed above--note we just added a divide by 2 to the end. What happens when you take the list of numbers "2 2 5 5" (the result of the expression above), and divide that by 2? Usually we just divide one number by one other number--so what does it mean to divide a group of numbers by 2? We'll talk more about this later, when we talk about vector and matrix computation. For now, just observe what happens.
- 1+4 * 2  <- BEFORE you type this, make a prediction of the outcome in QUESTION #7. Then continue to QUESTION #8 (Did it match your prediction? What did it do?) and QUESTION #9 (Add parenthesis to the expression "1+4 * 2" in order to make it easier to read but still give the same answer. Then add parenthesis in a different way in order to make the answer be 10. Test both your parenthesis-added expressions in the Command Window, and copy and paste below (both your expressions and the results)).

PART TEN: MORE EXPRESSIONS: VARIABLES, TYPES, and ASSIGNMENT

Sometimes we want to save the result of a computation, so we can use it later. Next we'll talk about saving the result of a computation into a variable by using assignment.

- First, we'll go back to our old friends, 1+2 and '1+2'. Type 1+2 (no quotes) and then look in the Workspace window (top right window in MATLAB), and see that ans is there, and that it shows the current value of ans:
Now try '1+2' (with single quotes), and then look back at the Workspace window. What happened?

- Note that the value of `ans` changed. Also note that the little icon next to `ans` (where the arrows are pointing in the pictures above) changed from a square divided into four quadrants to a square containing the letters `abc`. This change in icon indicates that the type of data being stored in the variable `ans` has changed from a number to a string of characters. Characters are letters, but in
MATLAB they include not only A-Z letters, but punctuation, numbers, etc. So '1+2' is a string of length 3. Single quotes are what signals to MATLAB that what you are typing should be interpreted as a string.

- MATLAB has two main types of **variables** that we will focus on for now: real numbers (sometimes called "doubles" in MATLAB), and **strings**. Each of these can be contained in a **vector** or **matrix**, which are two more types in MATLAB. (Actually, MATLAB views a number (ex: 2) to be a matrix that just happens to be a 1x1 matrix. So all integers and doubles are really of type matrix. *We’ll talk about vector and matrix types later.*)

- Let’s make some of new variables of our own by using the **assignment** operator, which is the equals sign =. Type in all of the following, copy all of it (and MATLAB’s responses) and paste into **QUESTION #10** (the questions are for observation/discussion only, you don’t need to answer them for lab question #10):
  - myname = 'Prof. Marx' <-- use your own name; now look for myname in the Workspace window
  - mystring = 'Hello, San Diego!' <-- try several different strings, including letters, many different punctuation, numbers
  - mystring = '3' <-- did the icon for mystring in the Workspace window change?
  - x = 3 <-- compare the icon for x to the icon for mystring in the Workspace window
  - y = 3^x <-- we can use the variable x that we made earlier
  - x = (x+x)*x <-- you can have the same variable on both sides of the assignment operator; see what happens

- Now let’s have some fun and make some errors (note: not all of the following generate errors--you should be looking to figure out which ones and why):
  - Start typing an assignment, but "accidentally" hit <ENTER> too soon--after you type the variable name but before you type the assignment operator "=" and the value to be assigned. Do you get an error?
  - x = 3
  - 3 = x
  - What does the above 2 expressions tell us about left-hand-side and right-hand-side of the assignment operator?
  - greatvariable1 = 3
  - lgreatvariable = 3
  - After trying the above 2 expressions, copy and paste your typing and MATLAB’s responses into **QUESTION #11**. Explain what rule for choosing variable names is illustrated.
  - **QUESTION #12**: Try making two variables that have the same name, except for the capitalization. Does MATLAB think these are the same variable, or two different variables?
(and describe how you can tell) Answer these questions in your own words, and also copy and paste the relevant interactions from your Command Window as needed to illustrate your points.

- **QUESTION #13**: Try making a variable name with an underscore character in it (underscore is `_`)--is that allowed? Does it matter where the underscore is located in the name? What about a dash (-)? Experiment with other punctuation marks and report more rules for variable names that you discover. Answer these questions in your own words, and also copy and paste the relevant interactions from your Command Window as needed to illustrate your points.

- `x = '1+1'` (with quotes)
- `y = 3`
- `z = x + y`  
  \(<--- What happens? Not what you expected?? Let's keep experimenting to figure it out. The next few suggested commands are just suggestions, you can design your own experiments to solve this mystery.

- `x = 'a'`
- `z = x + 1`
- `x = 'b'`
- `z = x + 1`
- `x = 'abcd'`
- `z = x + 1`
- `z = x - 1`
- `z = 2 * x`  
  \(<--- Are you seeing any patterns that give you clues as to what is going on? Speculate on what is going on in QUESTION #14. Do further experiments to test your hypothesis about what's happening, and be sure to report any further experiments in question #14. If you can correctly predict what would happen if you typed `x = 'efgh'` and then `z = x + 1`, then you probably have this figured out. Remember, you can always ask the TA if you need hints or explanations.

**PART ELEVEN: TURTLE (function practice)**

- So far, you have been using functions that are built-in to MATLAB. Your TA has written some functions that you will now use. To use them, you first need to download them and put them in your MATLAB folder.
- Download MURTLE.zip from the course webpage. Right click and save to desktop. Drag the MURTLE folder icon into your MATLAB folder. Now you should see a MURTLE folder inside MATLAB:
If you double-click the MURTLE folder, you will go in that folder, then you should see the list of files for this part of the lab:

- Type "NewTurtle()" in the Command Window to run the function that makes a new turtle. You should now see a large white image with a turtle in the center. We will now move the turtle.
- Type "Forward(20)" in the Command Window. Look at the image again to see what happened. Your turtle moves forward a small amount, leaving a line behind it.
• Now type the following commands, looking at the turtle image in between each command:

```
>> Forward(40)
>> Forward(20)
>> Left(90)
>> Forward(60)
```

• You should see two sides of a square. You have probably figured out that the Forward function moves the turtle forward by the amount specified in the argument. The Left function turns the turtle to the left by the number of degrees specified in the argument (the Right function does the same thing, but to the right).

• Type more commands in the Command Window to complete the square. If you get mixed up, you can close the turtle image window ("X" in the top right corner, just like all programs in Windows), and then type NewTurtle() again to make a fresh turtle image. Copy and paste your square-making code into your lab report as QUESTION #15.

PART TWELVE: LAB #1 CHECKOFF CHECKLIST

To receive credit for this lab you need to:

• Write your name on the whiteboard, when you are finished.
• Show your TA/Tutor your Notepad++ document in your ieng6 cs7wXX home directory folder.
• Be prepared to show the TA/Tutor that you are able to use Matlab.
• Be prepared to show the TA/Tutor all 15 questions answered in this document.
• Be able to answer questions about the MATLAB environment, functions, documentation and more.

It is your responsibility to make sure you get credit for each lab!

Do not leave until you have received an email (Lab#1) from autograder.

HOMEWORK #1:

• The Homework#1 assignment is due next week in YOUR lab section at the BEGINNING of Lab.
• You will need to show the Tutor/TA that you are able to use the Matlab commands from the homework assignment. He/She will ask questions to determine how knowledgeable you are.
• You will work individually on the homework assignments.
• You should have one file named “Hw1_LastName”, saved in your cs7wXX home directory. (See Homework # 1 details)
• You are free to go, or you are welcome to stay and work on the assignment (posted on the website) portion in the remainder of your enrolled lab section.