Day Two: MATLAB to its fullest

1 If statements

In a new script type the following code:

```
test = 2
if test>20
     disp('That number's pretty big!')
else
     disp('That number's just a little one.')
end
```

When we run this for different values of *test* it will check if *test* is larger than 20: if it is, the script run all the code between *if* and *else* and then skips to after end. If *test* is not larger than 20 it runs all the code between *else* and *end*, skipping the other code.

Exercise: There's nothing special about 20 or the code that runs. Change the code above so that if *test* is less than 3 the function prints "It's no bigger than 3!" and if not it plots a sine function.

Nick Namer

Lets make a program to give you guys all nick names if your names are too long:

```
name = 'Martina'
len = length(name)
if len > 5
     nick = name(1:5)
     disp(['Yeah, I'm just going to call you ', nick])
else
     disp(['Good to meet you, ', name])
     nick=name
end
```

What this code does is uses the function "length" to check the length of a name and then if its too long (gasp!) picks the first 5 letters using name(1:5).

Exercise: How do you make it so that it only picks three letters?

Coin Flip

Who needs a coin when we have MATLAB? We'll use the *rand* comand to give us a random number between 0 and 1 and then check if it's greater than .5. If it is, we'll call it heads!

```
coin = rand
if coin >.5
     disp('Heads!')
else
     disp('Tails!')
end
```

Is this a fair coin? We can use a for loop to run it 10000 times and check:

```
headsCounter = 0
tailsCounter = 0
for I = 1:10000
     coin =rand
     if coin>.5
          disp('Heads!')
          % Add 1 to the heads counter
     else
          disp('Tails!')
          % Add 1 to the Tails counter
     end
end
```

You need to finish the code here to make it add to the heads and tails counters each time heads or tails is flipped. After you've figured it out, what do you get? Do you get the same answer each time?

If you're done, change the code so that it plots the result of each coin flip with either a 1 or a 0

Random walk: If you've finished the above try the following: imagine a man is walking and at each step he flips a coin, if the coin is heads he takes a step to the right (x = x+1) and if it's tails he takes a step to the left (x = x-1). Where do we end up after 100 steps? Write some code to simulate this and run it sevral times. What do you usually get?

Functions

A function in programing is a piece of code that does a specific thing that we want done many times. Lets start by making our nick namer code into a fuction. To make a function we can use the "New Function" button or we can write the following code:

```
function [nick] = nickName(name)

len = length(name)

if len>5

     nick = name(1:5)

     disp(['Yeah, I'm just going to call you ', nick])

     name = nick

else

     disp(['Good to meet you, ', name])

end

end
```

Save this as *nickName.m*. In the command line you can type

```
>>nickName('Christopher')
```

and MATLAB will pop out Christopher's new nick name. Here's how it works, whatever name you put in when you call the function (type it in and run the code) that gets put into the variable *name*. The fuction then runs the code inside it and when it hits the last *end* it prints whatever's in the variable *nick* to the command line.

2 Mini Projects

You and your partner should pick one of the following mini project to work on for the rest of the class period.

2.1 Quadratic polynomials

A quadratic polynomial is a mathematical function of the form $f(x)=ax^2 + bx + c$. We want to build a function that takes variables a, b, c and tells us all about quadratic polynomials. We'll start simple and make it more interesting.

```
function roots = quad(a, b, c)

     % Put the polynomial code here

end
```

We want this function to do three things:

- First, create a vector x with all the numbers from -10 to 10, you can pick the 'step'.
- Second, make a vector y that store the values $ax^2 + bx + c$.
- Finally, plot x vs y to get a graph of $y = ax^2 + bx + c$.

Once you have a function that can plot any quadratic polynomial, lets expand it.

1. The maximum or minimum of a quadratic function is given by $-b/2a$. Add to your function code that computes that minimum and plots it on the graph with the parabola. Remember that *hold on* allows you to write to the same graph more than once.
2. The quadratic formula tells us where a quadratic polynomial is zero, these are called the roots of the polynomial.

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

   Add code to your function that checks if the roots are real (ie if $b^2 - 4ac \geq 0$) and if so finds the roots of $ax^2 + bx + c$. Have your function save both values in a list called roots and then plot both values on the same plot as the graph of the function.
3. How would you figure out if the maximum or minimum you found above is, well, a max or a min? Looking at the graph is easy but can you figure out a way that the computer could tell? Discuss this with your group. The command *text(x,y,'Your text')* places the message 'Your text' on the plot at the point (x, y). Update your function to label the maximum or minimum with max or min, respectively.

2.2 A bouquet of circles

We want to make a function that will draw a circle of radius r centered at the point (x_0, y_0). The plot function plots a sequence of (x, y) pairs in any shape we want, so we just need to tell it what the x and y coordinates of the circle are. But these are given by cosine and sine respectively! Try it out, write

```
     t = 0:2*3.14

     x = cos(t)

     y = sin (t)

     plot(x,y)
```

Did this make a circle? Why do we use that strange number 2*3.14? What do you need to change about the code above to make a circle?

Once you've fixed the code above and have a real circle lets put it into a function. Define a function

```
function circ(r, x0, y0)

% Plot a circle

end
```

The function takes three variables r, x0 and y0. The points on a circle of radius *r* centered at the point (x_0, y_0) are given by

$$x = x_0 + r\cos(t)$$

$$y = y_0 + r\sin(t)$$

Fill in the code above to finish the function. Once you're done, typing

```
>> circ(5, 2, 1)
```

Should draw a circle of a radius 5 centered at (2,1). Check the axes, this will look the same as any other circle when the window is zoomed in!

Now that you have a function that draws circles do the following problems

2.3 Maze

You can use *x = input* to request user input and save it in a variable x. This allows you to stope your code part way through and interact with a user. For example, the code

```
x = input('There are two doors in front of you, door 1 and
door 2, behind one lies a Lion, behind the other something
way less exciting. Which do you pick?')

if x == 1

     disp('You get  to chill with an awesome Lion')

else
```

```
     disp('Your life is safe, but you'd kind of like a
Lion')

end
```

and so on. See if you can use more if statements to make a simple story/maze.

Day 2 Cheat Sheet

Structures

```
if (expression)

     % Do something

else

     % Do something else

end
```

- Evaluates *expression* as either true or false. If it's true than run the code *Do something*. If it's false, run the code *Do something else*

```
while (expression)

     % Do something

end
```

- Check if expression is true. If it is, run the code Do something. If it is not, exit. Each time you're finished running the code Do something check if expression is true again, etc.

'Logical Symbols'

Name	Symbol
Greater Than	>
Less Than	<
Are Equal	==
And	&
Or	|
Not	□

Useful Functions

length(x) - Returns the number of elements in x. If x is a string 'like this' it returns the number of characters, including whitespaces.

disp['thing 1', 'thing 2', x, 'thing 3']) - Prints text to the command line. For each string or variable separated by a comma it will simply print them in order.

mod(a,b) - Returns the remainder after dividing *a* by *b*.

3 Homework

1. Write a function that takes 2 values and uses an if statement to return the largest.
2. Recall that x = a:b:c generates a vector x = [a, a+b, a+2b,..., c].

   (a) Plot $y = x^3 - x$ for $x = -5$ to $x = 5$, you can choose how smoothly to make the plot.

   (b) Use a for loop and hold on to plot $y = x^3 + nx$ for n = 0,1,2,...,10.
3. (Bonus) Write a function that takes a number *n* and draws *n* circles next to eachother.

4 Project Ideas

- Write a function that takes 3 numbers and returns a vector of the numbers sorted from least to greatest.
- Write a function that takes two numbers, m and n, and draws an m×n grid using for loops.
- Consider the quadratic polynomial $ax^2 + bx + c$. Create a function that takes three inputs, *a*, *b* and *c*, plots the polynomial and returns it's zeros if they exist.
- You might notice if you run the coin flip code that often it goes HHHTHTTTHT. Lets call a streak what happens if you get several heads in a row. Modify the code to keep track of the longest streak. For 100 flips, what's the longest streak? What about 1000?
- Make a mad-lib generator! Make a function that takes 5 words as inputs and then puts them into a sentence and prints out the sentence.

- Write a function that simulates flipping a coin using the random function. If the coin lands heads, have a function draw a heads up coin. If it lands tails have the function draw a tails up coin.
- Write a function that takes a number *n* and draws an *n* sided polygon.
- Write a function that draws a circle of circles.