

More MATLAB...

```
>> nums = [1 3 7 -8 4]
nums =
     1     3     7    -8     4
>> nums2D = [3 8 -4; -1 0 2]
nums2D =
     3     8    -4
    -1     0     2
>> size(nums)
ans =
     1     5
>> size(nums2D)
ans =
     2     3

>> nums = 2 * nums + 3
nums =
     5     9    17    23    11
```

```
>> nums(3)
ans =
     7
>> nums2D(2, 1)
ans =
    -1
>> nums(4) = 10
nums =
     1     3     7    10     4
>> nums2D(1, 3) = 6
nums2D =
     3     8     6
    -1     0     2

>> sqrt(3^2 + 4^2)
ans =
     5
>> nums2D.^2
ans =
     9    64    36
     1     0     4
```

Creating large images

```
>> image1 = zeros(200,200);
>> image2 = ones(200,200);
```

→ double numbers
→ 320,000 bytes!



```
>> image1 = uint8(ones(200,200));
```

→ 40,000 bytes!
→ integers 0-255



Exercise: Write one statement to create an initial 200x200 image of 100's*

* there are multiple ways to do this!

Colon notation

```
>> nums = 1:8
nums =
     1     2     3     4     5     6     7     8
>> vals = 2:3:21
vals =
     2     5     8    11    14    17    20
>> vals = 0.5:-1.5:-6.0
vals =
    0.5000   -1.0000   -2.5000   -4.0000   -5.5000
>> nums(2:4) = 0
nums =
     1     0     0     0     5     6     7     8
>> index = 4;
>> nums(index-2:index+2)
```

default step is 1

extends the sequence
as far as possible...

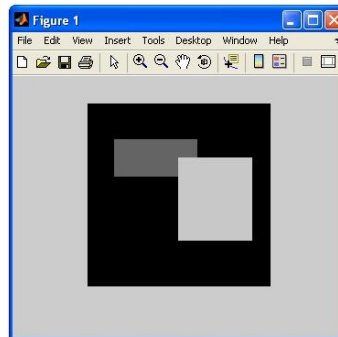
use variables in colon notation

```
ans =
```

Colon notation & images...

```
>> image = uint8(zeros(200,200));
>> image(40:80, 30:120) = 100;
>> image(60:150, 100:180) = 200;
>> imshow(image)
```

note:
pixel coordinates
printed by imshow
are reversed!



Defining new functions

```
function <outputs> = function-name (<inputs>)  
<statements comprising the body of the function>
```

getDistance.m:

```
function distance = getDistance(x1, y1, x2, y2)  
distance = sqrt((x2 - x1)^2 + (y2 - y1)^2);
```

```
>> dist = getDistance(1,1,8,4)  
dist =  
    7.6158
```

circleInfo.m:

```
function [area perimeter] = circleInfo(radius)  
area = pi * radius^2;  
perimeter = 2 * pi * radius;
```

```
>> [area1 perim1] = circleInfo(10.0)  
area1 =  
    314.1593  
perim1 =  
    62.8319
```

Conditionals: if statements

```
if (num < 0)  
    num = abs(num);  
end
```

```
nums = [3 7 2 9];  
if (val == 0)  
    result = sum(nums);  
elseif ((val == 1) | (val == 2))  
    result = prod(nums);  
elseif ((val > 2) & (val < 10))  
    result = min(nums);  
else  
    result = 0;  
end
```

```
if (val >= 10)  
    result = 10;  
else  
    result = 10 * val;  
    val = 0;  
end
```

```
if (num ~= 0)  
    result = 10/num;  
else  
    result = 0;  
end
```

Loops: for statements

```
for <variable-name> = <values>
    <commands to execute for each value of variable>
end
```

```
sum1 = 0;
for n = 1:10
    sum1 = sum1 + n^2;
end
```

```
prod1 = 1;
for val = 10.0:-1.5:-4.0
    prod1 = prod1 * val;
end
```

```
numbers = [7 1 5 9 2 3 6 4 8];
evens = 0;
for num = numbers
    if (rem(num,2) == 0)
        evens = evens + 1;
    end
end
```

```
image = uint8(50*rand(100,100));
count = 0;
for x = 1:100
    for y = 1:100
        if (image(x,y) > 25)
            count = count + 1;
        end
    end
end
```