

```

1 # Octave basics part 2
2
3 ## November 29, 2018
4
5 **No Binder links because Binder does not work with Octave. You are
  recommended to only use this as a reference while programming in the Octave or
  Matlab application.**
6
7 **You can vertically concatenate existing matrixes in a new matrix if the
  column lengths match**

```

```

In [1]: 1 array_1x3 = [1 2 3]
        2
        3 matrix_2x3 = [4 5 6; 7 8 9]
        4
        5 matrix_3x3 = [array_1x3; matrix_2x3]
        6

```

array_1x3 =

```

1  2  3

```

matrix_2x3 =

```

4  5  6
7  8  9

```

matrix_3x3 =

```

1  2  3
4  5  6
7  8  9

```

You can horizontally concatenate if the row lengths match

```

In [27]: 1 array_1x3 = [1 2 3]
        2
        3 matrix_1x2 = [4 5]
        4
        5 matrix_1x5 = [array_1x3, matrix_1x2]
        6

```

array_1x3 =

```

1  2  3

```

matrix_1x2 =

```

4  5

```

matrix_1x5 =

```

1  2  3  4  5

```

```
In [2]: 1 % Last expression automatically stored in variable 'ans'
        2
        3 [array_1x3; matrix_2x3]
        4
ans =
      1      2      3
      4      5      6
      7      8      9
```

```
In [3]: 1 ans
        2
ans =
      1      2      3
      4      5      6
      7      8      9
```

```
In [4]: 1 % Semicolon at the end of the line will suppress output
        2
        3 ans;
        4
```

Some basic MATLAB functions:

```
In [5]: 1 abs(-10)
        2
ans = 10
```

```
In [6]: 1 sqrt(9)
        2
ans = 3
```

```
In [7]: 1 disp('Hello CISC 106')
        2
Hello CISC 106
```

```
In [8]: 1 %ones(rows,cols) -- Matrix of all ones
        2
        3 ones(3, 3)
        4
ans =
```

```
      1      1      1
      1      1      1
      1      1      1
```

```
In [9]: 1 %zeros(rows,cols) -- Matrix of all zeros
        2
        3 zeros(3, 3)
        4
```

ans =

```
0  0  0
0  0  0
0  0  0
```

```
In [10]: 1 % rand(rows,cols) -- Matrix of uniformly distributed random elements
         2
         3 rand(3, 3)
         4
```

ans =

```
0.6611745    0.6893931    0.0510586
0.9652721    0.7360433    0.2345729
0.1414638    0.0636705    0.0094807
```

```
In [11]: 1 % randi([min_val,max_val],rows,cols) -- Matrix of random integers between min_
         2 randi([1, 6], 3, 3)
         3
```

ans =

```
2  2  1
1  3  4
1  2  5
```

Variable naming:

- Must start with a letter, and can have numbers, letters, underscores
- Case sensitive
- Don't need to declare
- You can make the variables any length -- but MATLAB only uses the first N characters to identify the variable – make first N characters unique
- N = number returned by - namelengthmax function
- Try: length_var_name = namelengthmax

```
In [12]: 1 namelengthmax()
         2
```

ans = 63

```
In [17]: 1 length_var_name = namelengthmax
         2
```

length_var_name = 63

```
In [20]: 1 length_var_name2 = namelengthmax
         2
```

length_var_name2 = 63

```
In [18]: 1 length_var_name
        2
        length_var_name = 63
```

```
In [23]: 1 % Every value assigned to a variable is an array
        2
        3 v = 15
        4 size(v)
        5
        6 v(1)
        7
        8 v = [15]
        9 size(v)
       10
       11 v = [15 16 17 18]
       12 size(v)
       13
```

```
v = 15
ans =
```

```
1 1
```

```
ans = 15
v = 15
ans =
```

```
1 1
```

```
v =
```

```
15 16 17 18
```

```
ans =
```

```
1 4
```

Let's solve [this Hackerrank problem \(https://www.hackerrank.com/challenges/diagonal-difference/problem\)](https://www.hackerrank.com/challenges/diagonal-difference/problem) using Octave: Complete the `diagonalDifference` function in the editor below. It must return an integer representing the absolute diagonal difference.

Here is an example function from Professor Wassil's slides (MATLAB Ch. 2 & 5) -- You can download from Canvas, I linked to them on the main page, and they are also listed under Files.

```
function [hzn_concat, vert_concat] = concatMat(A,B)
    % Use percent sign for comments.
    % Sorry, no block comments
    % Just use lots of percent signs
    hzn_concat = [A,B];
    vert_concat = [A;B];
end
```

```
In [31]: 1 function diag_diff = diagonalDifference(a_matrix)
2         regular_diag = sum(diag(a_matrix));
3         anti_diag = sum(diag(flipud(a_matrix)));
4
5         diag_diff = abs(regular_diag - anti_diag);
6
7     end
8
```

```
In [34]: 1 matrix_3x3 = [1 2 3; 7 8 9; 6 2 6]
2
3     d_diff = diagonalDifference(matrix_3x3)
4
```

matrix_3x3 =

```
1   2   3
7   8   9
6   2   6
```

d_diff = 2

```
In [35]: 1 % Function definition from Wassil's slides
2 function [hzn_concat, vert_concat] = concatMat(A,B)
3         hzn_concat = [A,B];
4         vert_concat = [A;B];
5     end
6
```

```
In [36]: 1 matrix_1x3_a = [1 5 9]
2 matrix_1x3_b = [10 15 19]
3
4 [h_cat, v_cat] = concatMat(matrix_1x3_a, matrix_1x3_b)
5
```

matrix_1x3_a =

```
1   5   9
```

matrix_1x3_b =

```
10  15  19
```

h_cat =

```
1   5   9  10  15  19
```

v_cat =

```
1   5   9
10  15  19
```

In [49]:

```
1 % From Wassil's PDF chap 2 & 5, slide 6
2
3 A = 10:-2:-10
4
5 A(1,5)
6
7 A(1,6)
8
9 if ( A(1,5) == A(1,6) )
10     disp("equal")
11 elseif ( A(1,5) > A(1,6) )
12     disp("greater")
13 else
14     disp("less than")
15 end
16
```

A =

```
10    8    6    4    2    0   -2   -4   -6   -8  -10
```

ans = 2

ans = 0

greater

In [50]:

```
1
2 i
```

ans = 0 + 1i

In [51]:

```
1 j
2
```

ans = 0 + 1i

In [52]:

```
1 i == j
2
```

ans = 1

In [61]:

```
1 % Boolean examples
2
3 % Our matrix
4 matrix_3x3 = [1 2 3; 4 5 6; 7 8 9]
5
```

matrix_3x3 =

```
1 2 3
4 5 6
7 8 9
```

```
In [62]: 1 % Not of every element (0 if not 0, 1 if 0)
          2
          3 ~matrix_3x3
          4
          5 ~[1 0 3]
          6
```

ans =

```
0 0 0
0 0 0
0 0 0
```

ans =

```
0 1 0
```

```
In [63]: 1 matrix_3x3 & 1
          2
```

ans =

```
1 1 1
1 1 1
1 1 1
```

```
In [64]: 1 matrix_3x3 | 1
          2
```

ans =

```
1 1 1
1 1 1
1 1 1
```

```
In [69]: 1 % How to find index with single number?
          2
          3 matrix_3x3(8)
          4
          5 % Start at (1, 1) and go down through the first column, then continue
          6 % counting at (1,2) and continue down the second column, etc., until you
          7 % find the 8th element
          8
```

ans = 6