Vectors & Matrices

- Vectors & Matrices store sets of values, all of which have the same type.
- *row vector*
- *column vector*
- *scalar*
- *matrix*
- *elements*

Creating row vectors

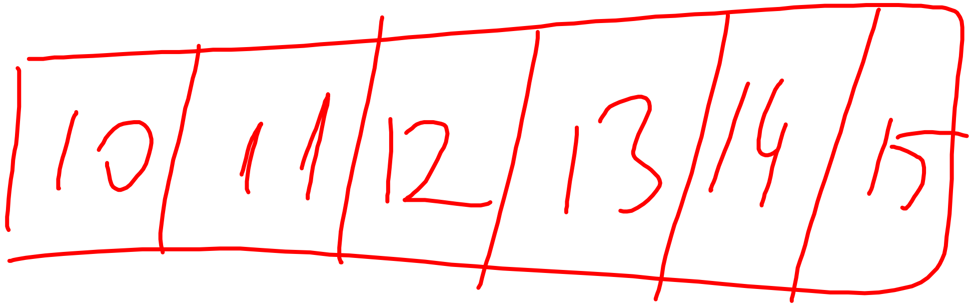
- v = [1,2,3,4]
- v = [1   2, 3   4]
- Colon operator (*iterator*): create equally spaced numbers
  - **from : step : to**
  - v = 2:1:6
  - v = 2:6
  - v = 1:2:9
  - 1:2:6
  - 6:3
  - 9:-2:1
- linspace(from , to, n)
  - linspace(2,6,5)
  - linspace(6,18,5)
  - linspace(18,6,5)

Concatenating vectors

- a = [1 2]
- b = [3 4 5]
- b = [[3 4] 5]
- c = [a b]
- c = [a b 1 2 3]

Indexing (Accessing) vectors

- **variable (index)**
- v=10:15
- v(3) *"v sub 3"*
- index itself can be a vector
  - v([1 2 3])
- the indexed entries can be modified:
  - v(2)=30
- Matlab automatically extends vector if indexed element does not exist.
  - v(10)=99
  - Avoid automatic extension when you care about speed.



Exercise

- What is the value of **a** after the following statements are executed?
  - a=2:2:8
  - a(4)=50
  - a(6)=11
  - a=a(3:6)
  - a=[a linspace(4,12,3)]

Creating column vectors

- c = [1; 2; 3; 4]
- Row vectors can be transposed using '
- r = 1:3
- c = r'
- Exercise: Does the following result in a row or column vector ?
  - 1:3'

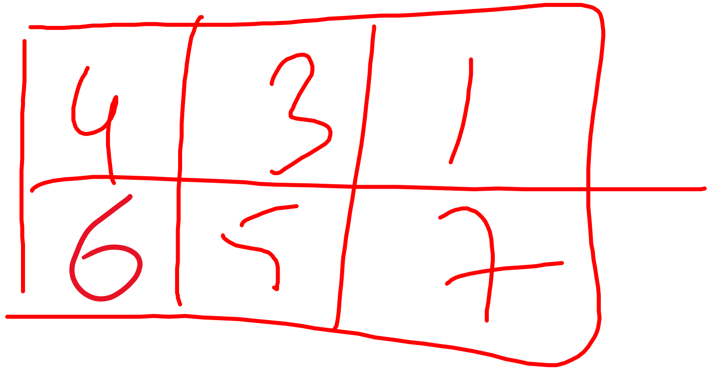
Creating matrix variables

- m=[4 , 3 1; 2, 5 6]
- m=[4 3 1

  2 5 6]
- There must always be the same number of elements in each row.
- m=[2:4; 3:6]

Linear Indexing

- Matlab stores and indexes matrices column-by-column.
- We can index a matrix as if it is a vector.
- m=[
  4 3 1
  6 5 7]
- m(1)
- m(2)
- m(end-1:end)



(Row,Column) Indexing Matrices

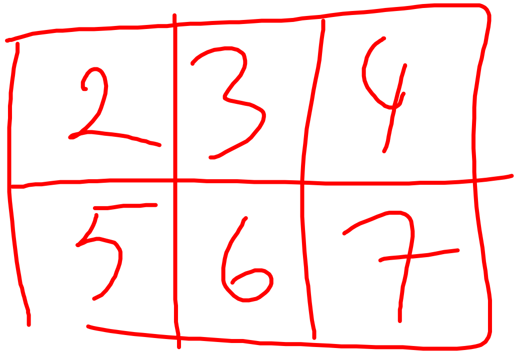
- m=[
  2 3 4
  5 6 7]
- m(2,3)
- m(1:2,2:3)
- m([2 2], [3 1 3])
- m(1,:)
- m(:,2)
- m(1,2:end)

- What about m([2, 3]) ?

2	3	4
5	6	7

Modifying matrix elements

- m=[
  2 3 4
  5 6 7]
- m(1,1)=9
- m(1,1:2)=13
- m(1,[2 1])=[8 11]
- m(1,:)=9
- m(1,:)=[9 9]
- m(5,:)=1:3



Generator Functions

- rand
- rand(R)
- rand(R, C)
- Others: zeros(), ones(), inf(), nan(), true(), false(), randi()

- randi(**Max**, R, C)

- magic(R)

Matrix Dimensions

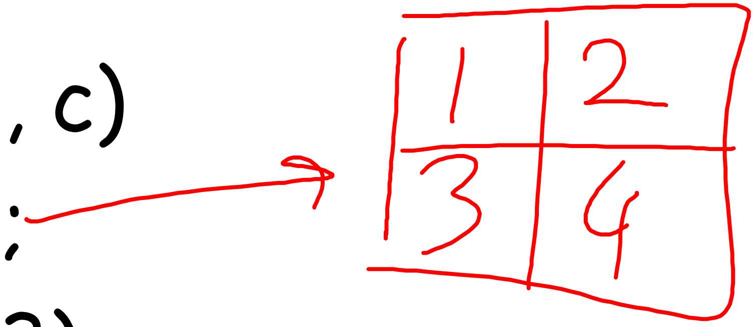
- size
- numel
- ~~length~~
- m=rand(2,3)
- size(m)
- [R, C] = size(m)
- numel(m)
- Exercise:
  - Write function that takes a matrix m as input, and returns a matrix of zeros with the same size as m.

Changing Dimensions

- reshape
  - m=randi(100,3,4)
  - reshape(m,2,6)
  - reshape(m,4,3)
  - reshape(m,4,[])

- fliplr(m)
- flipud(m)
- rot90(m) *rotates counterclockwise*
  - rot90(m,-1)
  - rot90(m,2)

Replicating matrix

- repmat(m, r, c)
- m=[1 2; 3 4];


- repmat(m,1,3)
- repmat(m,2,2)
- repmat(m,2,3)

Exercise

- 1.32: Find an "efficient" way to generate the following matrix:

  m =

7	8	9	10
12	10	8	6

- Increment the first row of the above matrix m with +1, and the second row of m with +2, in a single statement.
  - If matrix m had more than two rows, your code should add +3 to the third row, +4 to the fourth row, etc.

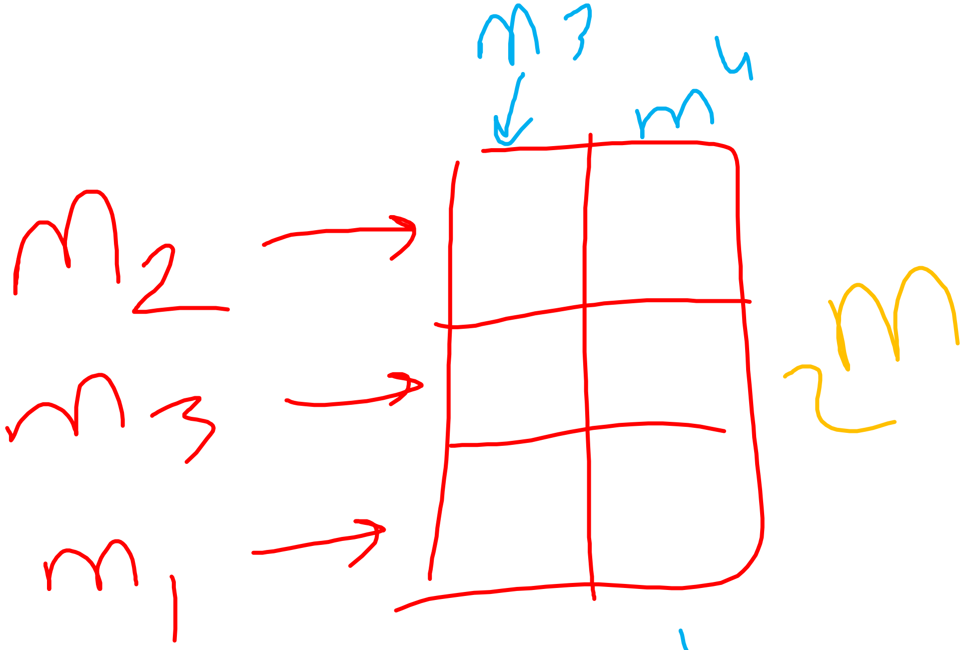
Empty vectors

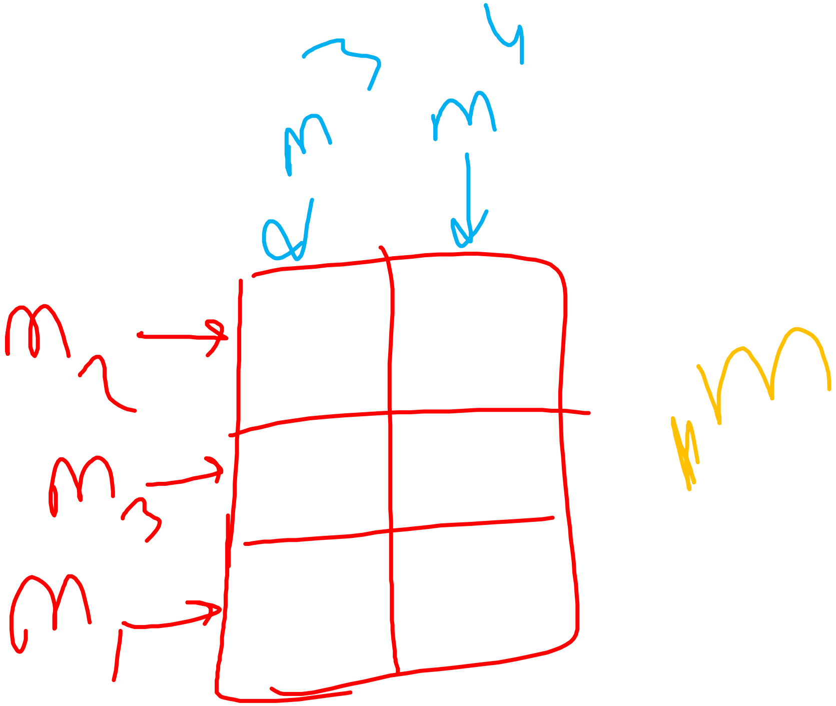
- e=[]
  - size(e)
  - numel(e)
- Empty vectors can be used to delete elements from vectors/matrices
  - m=randi(10,3,4)
  - m(:,4) =[]
  - m(1,:) =[]
  - m(2:4) =[]

Three dimensional matrices

- m=zeros(3,4,2)
- m(:,:,1)=randi(100,3,4)
- m(:,:,2)=randi(100,3,4)

```
m=randi(100,3,4,2)

m(:,:,1) =

    62    13    84    84
    35    74    40    33
    94    65    75    56


m(:,:,2) =

    98    62    42    98
    55    37    50    33
    34    76    70    84
```





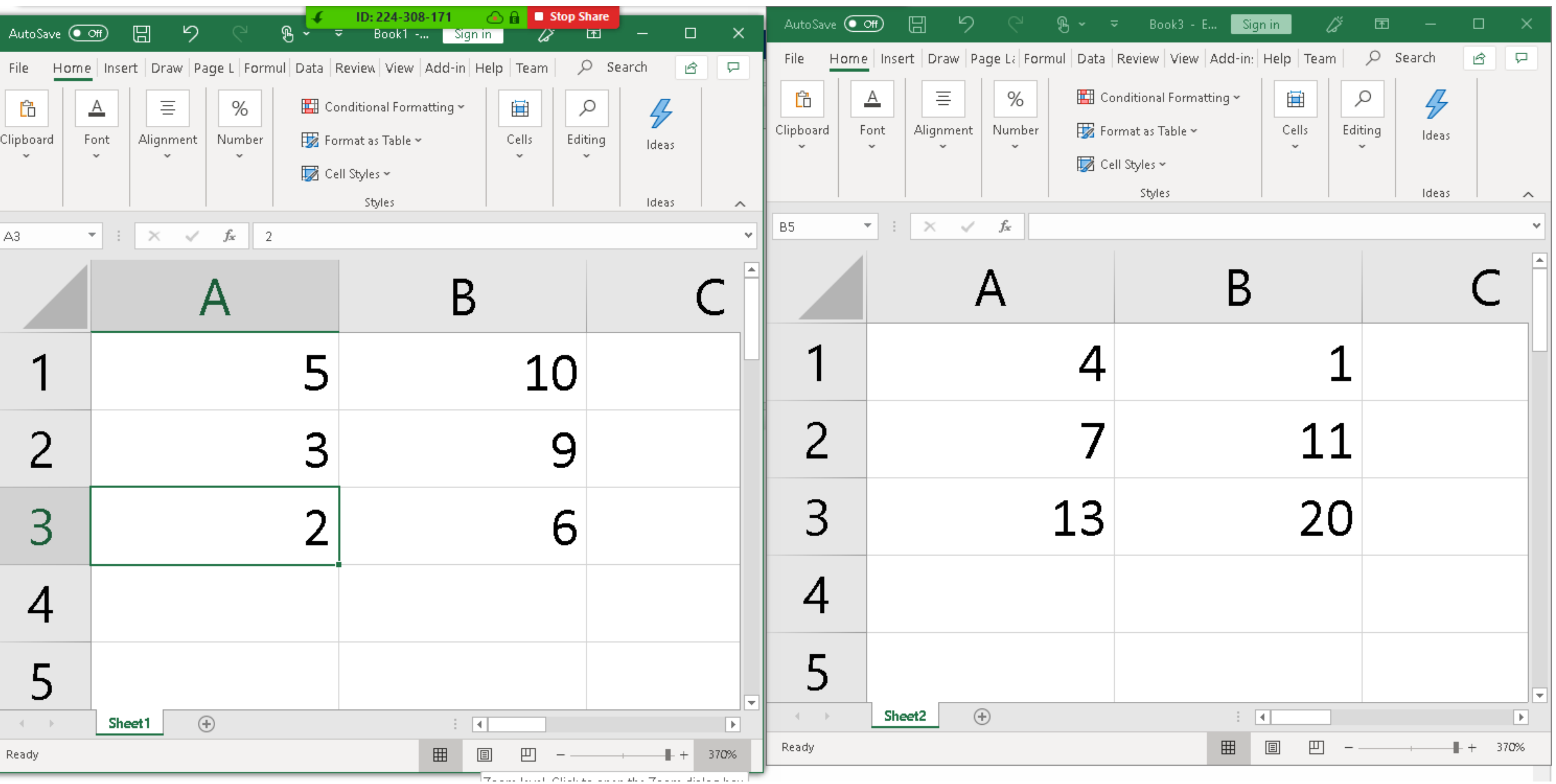
m([2 3 1],[3 4],[2 1])

Basic statistics functions

- **sum**(v), sum(A), sum(A,dim)
- **mean**(v), mean(A), mean(A,dim)
- **std**(v), std(A), std(A,0,dim)

- **diff**(v), diff(A), diff(A,[],dim), diff(A,k)
- **min**(v), min(A), min(A, [], dim), min(A,b), min(A,B)
- **max**(v), max(A), max(A, [], dim), max(A,b), max(A,B)

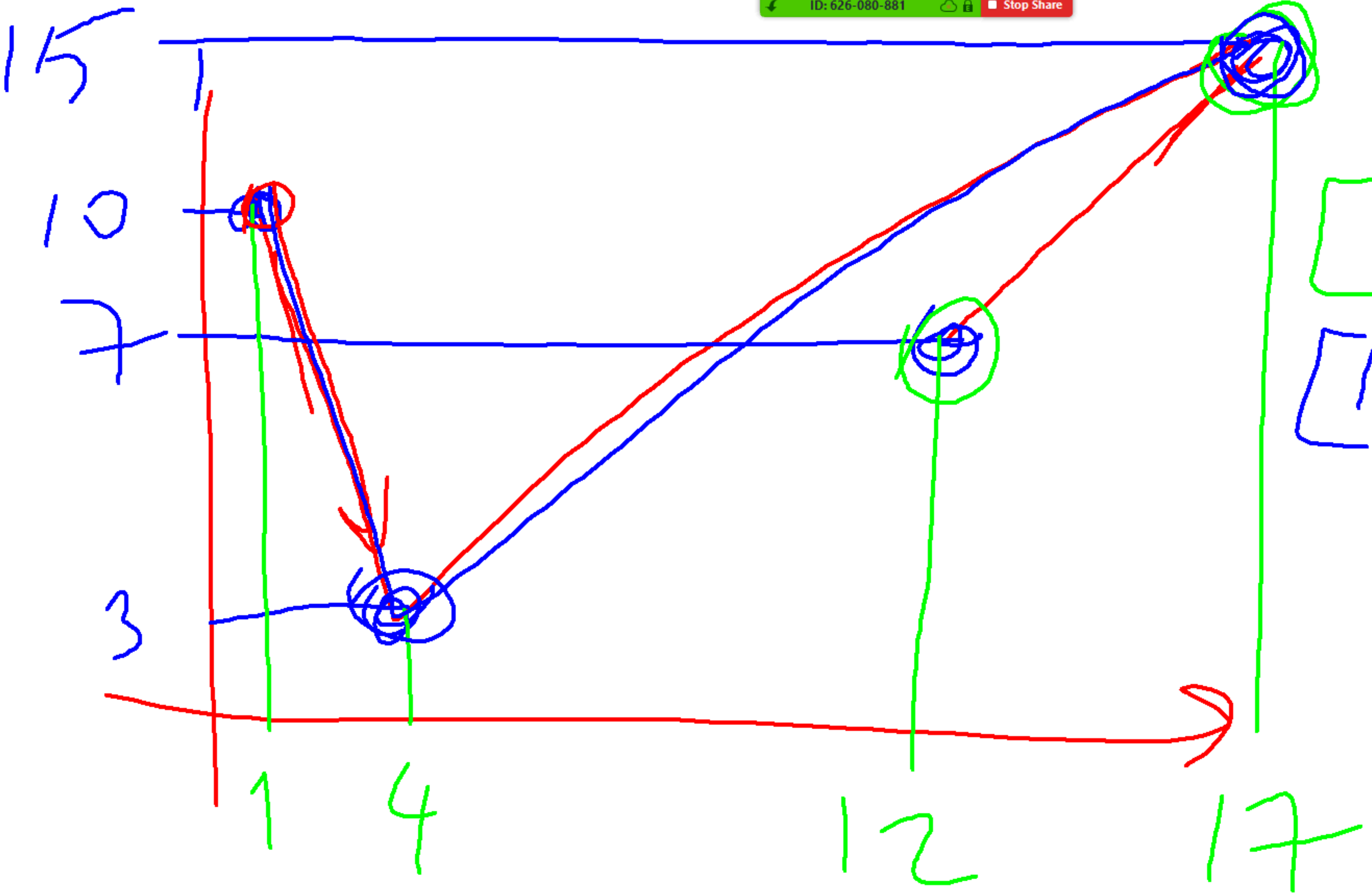
- [x,pos] = **min**(v)
- [x,pos] = **max**(v)


Book1
A3
2
Sheet1
A B C
1 5 10
2 3 9
3 2 6
Book3
B5
Sheet2
A B C
1 4 1
2 7 11
3 13 20

Plotting

- plot(X,Y)
- plot(X,Y,'r*')
- scatter(X,Y)

- axis([xlow, xhigh, ylow, yhigh])
- xlabel('time (sec)')
- ylabel('temperature (Fahrenheit)')
- title('Temperature vs. time')
- legend
- grid on
- subplot(R,C, i)
- hold on/off
- clf
- Saving plots as image files (File -> SaveAs).



[1 4 17 12]
[10 3 15 7]

plot([1 4 17 12],[10 3 15 7])

Exercise

- Draw four random triangles using the plot() function. Use a different color, marker, and line type for each triangle.

Bar-plot

- bar(Y)
- bar(X, Y)
- errorbar(X,Y, E)

Summary

- [...] creates vectors and matrices
  - comma or space adds entries on the same row
  - semicolon or linebreak introduces new row
- Linear Indexing: v(ind)
  - ind can be a scalar to access an individual element
  - ind can be a vector to access multiple elements
  - If v is a matrix, we pretend it is a vector by considering column-by-column ordering of its elements
  - "end" keyword within ind replaced with number of elements of v.
- Row-Column Indexing: m(r,c)
  - r/c can be scalars, to access an individual element
  - r/c can be vectors, to access multiple elements
  - The result will have the same number of rows as r, and the same number of columns as c.
  - Values in r determine which rows of m are used to fill in each result row. Values in c determine which columns of m are used to fill in each result column.
  - end keyword within r replaced with number of rows of m.
  - end keyword within c replaced with number of columns of m.

Summary

- v(ind) = x; m(r,c) = x;   When indexing is used as target of an assignment:
  - If multiple elements are indexed and there is a scalar x: x is copied into each indexed position.
  - If multiple elements are indexed and x is not a scalar: there needs to be the same number of elements in x and the number of positions being indexed.
- v(ind) = []   and m(r,c) = [ ]   are used to remove the indexed entries from vector/matrix.
  - When a matrix is linearly indexed, the removal of elements would force it to become a vector.

Summary

- rand, zeros, ones, inf, nan, true, false, randi
  - Create a scalar, when no dimension arguments are given: rand()
  - Create a square matrix, when a single dimension argument is given: rand(5)
  - Create a matrix with any number of rows and columns, when two dimension arguments are given: rand(3,4)
  - randi has a reserved first input that must be provided before any dimension arguments are given.
- sum, mean, std, min, max, diff
  - When a vector input given (regardless of row vector or column vector), operate on the vector.
  - When a matrix is given, operate on each column separately.
  - If need to perform on each row, supply a dimension argument 2.
  - Some of these functions have a reserved second input argument, which must be specified before any dimension arguments are given.

Summary

- reshape() creates a matrix with different number of rows and columns, while preserving the linear order of elements
- repmat(x, r, c) uses x as a "brick" to build a wall that is r high and c wide.