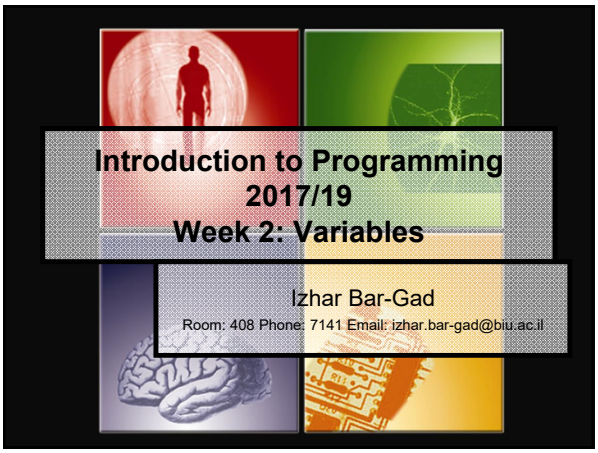
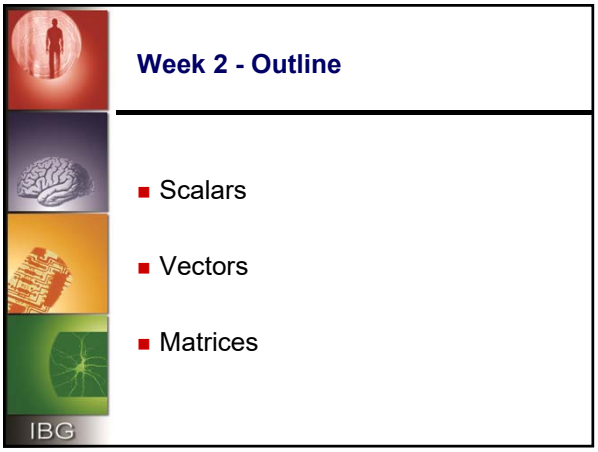
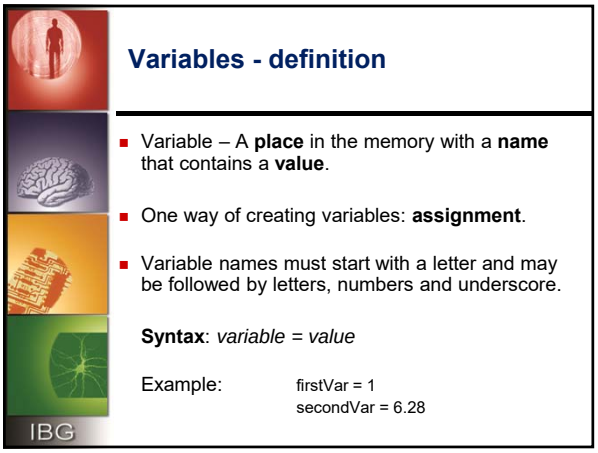

Introduction to Programming
2017/19
Week 2: Variables
Izhar Bar-Gad
Room: 408 Phone: 7141 Email: izhar.bar-gad@biu.ac.il


Week 2 - Outline
Scalars
Vectors
Matrices
IBG


Variables - definition
Variable – A place in the memory with a name that contains a value.
One way of creating variables: assignment.
Variable names must start with a letter and may be followed by letters, numbers and underscore.
Syntax: variable = value
Example:
firstVar = 1
secondVar = 6.28
IBG

Variables - naming

- **Meaningless**
  - rty11 = 13
  - w_w_w = 1.324556
- **Meaningful**
  - minNeuronActivity = 10
  - numPatients = 7

IBG

Variables - guidelines

- Assignment to an existing variable overrides it
  duplicateVar = 1
  duplicateVar = 2
- Variable names are case sensitive
  - 'Num' and 'num' are not the same.
- A variable may be assigned another variable
  firstVar = 1
  secondVar = firstVar
- One cannot use a variable without defining it first. Using an undefined variable causes **error**.
  firstVar = 1
  secondVar = firstvar → error

IBG

Variables - operations

- Numerical operations can be performed on numbers or variables:
  var1 = var2 * var3;
  varNew = 2 + varOld;
- Basic operators:

+ add	x = 2+3 → x = 5
- subtract	x = 2-3 → x = -1
* multiply	x = 2*3 → x = 6
/ divide	x = 2/3 → x = 0.666…
^ power	x = 2^3 → x = 8
function	x = sqrt(9) → x = 3

Syntax: *function_name(variable)*

IBG

One dimensional arrays - vector

(1D) Array – A sequence of ordered elements.
- Example: 3 1 7 9 4 8

- Creating an array via assignment. The elements are placed inside [ ], separated by spaces.
  firstVector = [3 1 7 9 4 8]
- An array of one element is also called a **scalar**.
  firstScalar = 3 or firstScalar = [3]
- An array may be empty → no elements
  tinyArray = [ ]

IBG

Arrays – Indices

- Every element in the array has a place → **index**.
  myVec = [3 1 7 9 1] → The index of 7 is 3.
- **Retrieval** – Getting an element from a specific index in the array: *arrayName(index)*
  myVec(3) → 7
- **Assignment** – an element can be replaced: *arrayName(index) = newValue*
  myVec(3) = 5 → myVec = [3 1 5 9 1]

myVec(1)	myVec(2)	myVec(3)	myVec(4)	myVec(5)
3	1	7	9	1

IBG

Arrays - Series

- When the elements in the array are a series, there is a shortcut: Use ':' for a series.
- Default increment is 1:
  seriesVec = [1 : 6] → seriesVec = [1 2 3 4 5 6]
- For other increments: The central number is the element difference.
  - Positive increments:
    posVec = [1 : 0.5 : 3] → posVec = [1 1.5 2 2.5 3]
  - Negative decements:
    negVec = [1 : -2 : -8] → negVec = [1 –1 –3 –5 –7]

IBG

Arrays – Length

- **Length** of an array = the number of elements in the array.
- Syntax: length(array_name).
- Examples:
  - Array: myVec = [3 1 5 9 4 8], length(myVec) → 6
  - Empty array: emptyVec = [ ], length(emptyVec) → 0
  - Scalar: myScalar = 3, length(myScalar) → 1

IBG

Operations: Array and scalar

- The operation is performed on the scalar and each of the array elements.

If: array = [a1 a2 a3 …] & s is a scalar
Then: array + s → [a1+s a2+s a3+s …]
The operations are: +, -, *, /

Examples:

A = [3 1 5 9]
B = A + 2 → B = [5 3 7 11]
C = 3 * A → C = [9 3 15 27]

IBG

Operations: Two arrays I

- Arrays must have the same length.
- Operation is element-wise.
  - Some operations are changed:
    * → .*
    / → ./
    ^ → .^
  - (+) and (–) remain the same

IBG

Operations: Two arrays II

- Numerical operations on 2 arrays:
  arrayA = [a1 a2 a3 …], arrayB = [b1 b2 b3 …]
  arrayA + arrayB → [a1+b1 a2+b2 a3+b3 …]
  arrayA .* arrayB → [a1*b1 a2*b2 a3*b3 …]
- Examples:
  A = [3 1 5 9 4 8] B = [8 2 7 4 1 6]
  C = A + B → C = [11 3 12 13 4 14]
  D = A .* B → D = [24 2 35 36 4 48]

IBG

Arrays – Functions Operations

A function can operate on an array.

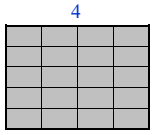
- Syntax: function_name(array_name).
  array = [a1 a2 a3 …]
  f(array) → f(a1) f(a2) f(a3) …
- Example:
  array = [9 49 16 4]
  sqrt(array) → [3 7 4 2]

IBG

Two dimensional Arrays - Matrices

- Matrix – 2D array (table).
- Array (reminder) – A sequence of ordered elements.
- 2D array – Elements are ordered in 2 dimensions: row and column.
- M x N matrix – M rows, N columns.

There are higher dimensional arrays but we will get to them only much later.



IBG

Matrices – Special cases

- Scalar – 1 x 1 matrix
- Vector – 1D array
  - Row vector – 1 x N matrix

$$\begin{pmatrix} 3 & 7 & 4 & 1 & 8 & 13 \end{pmatrix}$$

  - Column vector – M x 1 matrix

$$\begin{pmatrix} 2 \\ 1 \\ 7 \\ 9 \end{pmatrix}$$

IBG

Creating scalars and vectors

- Scalar.
  X = 3;
  X = [3];
- Row vector (separated by spaces of commas)
  vRow = [3 1 7 5]; → vRow is a 1x4 matrix.
  also [3, 1, 7, 5]

$$vRow = \begin{pmatrix} 3 & 1 & 7 & 5 \end{pmatrix}$$

- Column vector (separated by semicolons)
  vCol = [3; 1; 7; 5]; → vCol is a 4x1 matrix.

$$vCol = \begin{pmatrix} 3 \\ 1 \\ 7 \\ 5 \end{pmatrix}$$

IBG

Creating matrices

- Matrix – Values in the same row are separated with space/comma (like row vector), semicolon separates between rows (like column vector).
- All rows must have the same number of columns (elements).
- Example:
  validMat = [6 4 2 8 ; 1 9 5 4 ; 3 3 7 6];
  → A is a 3x4 matrix.

$$validMat = \begin{pmatrix} 6 & 4 & 2 & 8 \\ 1 & 9 & 5 & 4 \\ 3 & 3 & 7 & 6 \end{pmatrix}$$

- Invalid Example:
  invalidMat = [6 4 2 8 ; 1 9 5];
  → Number of columns in each row must be the same.

IBG

Vectors - Indices

- Vector - 1D Matrix – The elements are indexed from 1 to N (row vector) or from 1 to M (column vector), similar to 1D array.

$\begin{pmatrix} 3 & 7 & 4 & 1 & 8 & 13 \end{pmatrix}$ →The 3rd element is 4.

$\begin{pmatrix} 2 \\ 1 \\ 7 \\ 9 \end{pmatrix}$ →The 3rd element is 7.

IBG

Matrices – Indices

- 2D Matrix – Elements are numbered by 2 indices - the a_{ij} element is the element in the i-th row and the j-th column.

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}$$

- Example:

$$\begin{pmatrix} 5 & 8 & 12 & 4 \\ 7 & 1 & 9 & 3 \\ 11 & 5 & 2 & 13 \\ 3 & 6 & 10 & 8 \end{pmatrix}$$

a_{23}

IBG

Matrices - Retrieval of elements

matrix_name(i, j) get the a_{ij} element of the matrix.

Example: myMat = [6 4 2 8 ; 1 9 5 4 ; 3 3 7 6];

myMat(2,1) → 1
myMat(3,3) → 7

$$myMat = \begin{pmatrix} 6 & 4 & 2 & 8 \\ 1 & 9 & 5 & 4 \\ 3 & 3 & 7 & 6 \end{pmatrix}$$

IBG

Vectors - Retrieval of Elements

- A vector can be treated as one dimensional or two dimensional
- Examples:
  - Row vector
    V = [5 8 3 6];
    V(1, 2) → 8 or V(2) → 8
  - Column vector
    V = [5 ; 8 ; 3 ; 6];
    V(2, 1) → 8 or V(2) → 8

IBG

Matrices - Retrieval of vectors

- Matrices and vectors can retrieve vectors, and not only single elements.
- An array of indices is given:

M([1 2], 2) → $\begin{pmatrix} 4 \\ 9 \end{pmatrix}$

M(1, 1:3) → (6 4 2)

$$M = \begin{pmatrix} 6 & 4 & 2 & 8 \\ 1 & 9 & 5 & 4 \\ 3 & 3 & 7 & 6 \end{pmatrix}$$

All elements: row: M(m, :), column: M(:, m).

M(3, :) → (3 3 7 6)

M(3, 2:end) → (3 7 6)

IBG

Matrices - Retrieval of sub-matrix

- Matrices can retrieve sub-matrices.

$$M = \begin{pmatrix} 6 & 4 & 2 & 8 \\ 1 & 9 & 5 & 4 \\ 3 & 3 & 7 & 6 \end{pmatrix}$$

- Examples:

M([1 2], [3 4]) → $\begin{pmatrix} 2 & 8 \\ 5 & 4 \end{pmatrix}$

M(1:3, 2:end) → $\begin{pmatrix} 4 & 2 & 8 \\ 9 & 5 & 4 \\ 3 & 7 & 6 \end{pmatrix}$

IBG

Assignment of single elements

- Syntax: matrix_name(row, col) = new_value.
- Example: M(2,1) = 4;

$$M = \begin{pmatrix} 6 & 4 & 2 & 8 \\ 1 & 9 & 5 & 4 \\ 3 & 3 & 7 & 6 \end{pmatrix} \rightarrow \begin{pmatrix} 6 & 4 & 2 & 8 \\ 4 & 9 & 5 & 4 \\ 3 & 3 & 7 & 6 \end{pmatrix}$$

IBG

Assignment of vectors

- Vectors can also be assigned.
- The assigned vector must have the same size as the vector that is being replaced.
- Examples:

M([1 3], 3) = [8; 9];

$$\begin{pmatrix} 6 & 4 & 2 & 8 \\ 4 & 9 & 5 & 4 \\ 3 & 3 & 7 & 6 \end{pmatrix} \rightarrow \begin{pmatrix} 6 & 4 & 8 & 8 \\ 4 & 9 & 5 & 4 \\ 3 & 3 & 9 & 6 \end{pmatrix}$$

M(3, [2 4]) = [2 3];
M(2, :) = [8 2 1 5];

IBG

Assignment of matrices

- Matrices can also be assigned.
- The assigned matrix must have the same size as the matrix that is being replaced.
- Example

M([1 2], [3 4]) = [7 7 ; 7 7];

$$\begin{pmatrix} 6 & 4 & 2 & 8 \\ 4 & 9 & 5 & 4 \\ 3 & 3 & 7 & 6 \end{pmatrix} \rightarrow \begin{pmatrix} 6 & 4 & 7 & 7 \\ 4 & 9 & 7 & 7 \\ 3 & 3 & 7 & 6 \end{pmatrix}$$

IBG

Adding elements

- Elements may be added to a matrix directly by assignment.
  T = [-3 6 1];
  T(4) = 7 → T = [-3 6 1 7];
- If more elements are created while assigning - 0 is assigned automatically.

T(2, 3) = 4; → T = $\begin{pmatrix} -3 & 6 & 1 & 7 \\ 0 & 0 & 4 & 0 \end{pmatrix}$

- Vectors and Matrices can also be added to a matrix.

IBG

Deleting elements

- Elements can be deleted by assigning the empty array.

T = $\begin{pmatrix} -3 & 6 & 1 & 7 \\ 0 & 0 & 4 & 0 \end{pmatrix}$

T(:, 4) = []; → T = $\begin{pmatrix} -3 & 1 & 6 \\ 0 & 0 & 4 \end{pmatrix}$

- A whole row/column must be deleted, and not single elements.

IBG

Concatenation of matrices I

- Two (or more) matrices can be concatenated to one matrix: adding rows, adding columns.
- Adding rows - number of columns must be the same.
- Adding columns – number of rows must be the same.
- More than two matrices can be concatenated.

IBG

Concatenation of matrices II

- Scalars:
  rowVector = [var1 var2]
  columnVector = [var1; var2]
- Row vector:
  totalVector = [firstVector secondVector]
- Column vector:
  totalVector = [firstVector; secondVector]
- Matrix
  addColumnsMatrix = [firstMatrix secondMatrix]
  addRowsMatrix = [firstMatrix; secondMatrix]
  - Same number of rows/columns.
- 'cat' command.

IBG

Concatenation of matrices III

Examples:

$A = \begin{pmatrix} 8 & 5 & 2 \\ 4 & 1 & 9 \end{pmatrix}$ $B = \begin{pmatrix} 6 & 3 & 2 & 7 \\ 5 & 6 & 4 & 8 \end{pmatrix}$ $C = \begin{pmatrix} 7 & 4 & 6 & 2 \\ 2 & 1 & 8 & 5 \\ 9 & 3 & 7 & 9 \end{pmatrix}$

Adding columns: arrays separated with spaces

D = [A B]; → $\begin{pmatrix} 8 & 5 & 2 & 6 & 3 & 2 & 7 \\ 4 & 1 & 9 & 5 & 6 & 4 & 8 \end{pmatrix}$

Adding rows: arrays separated with ;

E = [B ; C]; → $\begin{pmatrix} 6 & 3 & 2 & 7 \\ 5 & 6 & 4 & 8 \\ 7 & 4 & 6 & 2 \\ 2 & 1 & 8 & 5 \\ 9 & 3 & 7 & 9 \end{pmatrix}$

IBG

Size of matrices

- Size of a matrix – vector of the length of the dimensions of the matrix (rows cols)
- Syntax: size(matrix_name)
  A is an MxN matrix → size(A) is [M N]
- Example:
  A = [6 3 ; 8 2 ; 4 9];
  size(A) → 3 2

IBG

Size of a specific dimension

- A specific dimension can be requested.
- Syntax: size(matrix_name, dim).
- Dimensions: 1 – rows, 2 – columns.

- Example:
  A = [6 3 ; 8 2 ; 4 9];
  Rows: size(A, 1) → 3
  Columns: size(A, 2) → 2

IBG

Numerical operations

- Numerical operations on matrices and vectors are the same as for arrays.
- Matrix and scalar: +, -, *, /
- Matrix and matrix : +, -, .*, ./, .^
  - Matrices must have the same size.
  - Operation is element-wise.
- Functions.
  - Example: sqrt(A).

IBG

MATLAB Programming Style Guidelines

- What do we need it for?
  - **We** will be able to understand our code.
  - **Others** will be able to understand our code.

- What is it?
  - Rules for naming variables and functions.
  - General guidelines for building our program.
  - General guidelines for organizing our code.

IBG

Style - Variables

- Variables names.
  - Meaningful names.
  - Start with lower-case letter, every word in the name starts with upper-case letter.
  - No underscores (_).
- Example:
  rowSum (and not: RowSum, row_sum, etc.)