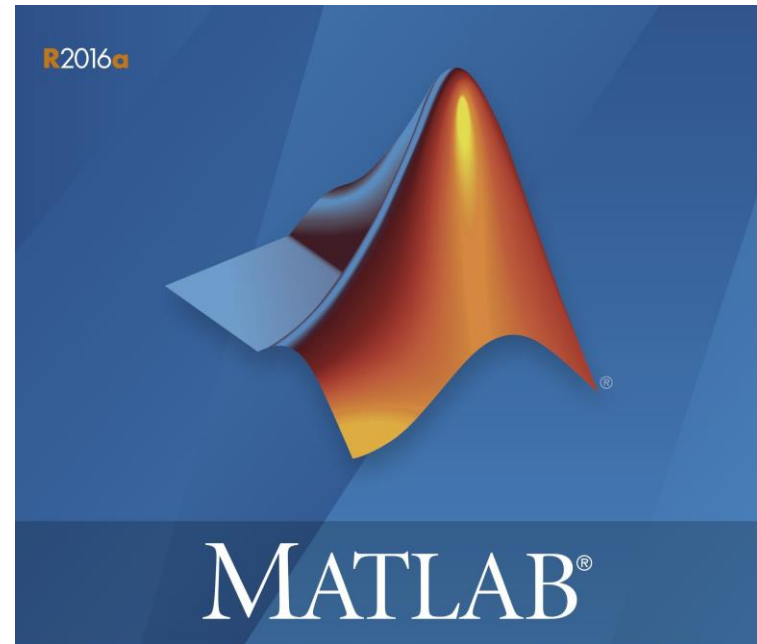


# MATLAB

---

a short primer



Ovidiu Ivanov  
'Gheorghe Asachi' Technical University,  
Electrical Engineering Faculty  
Iasi, Romania  
[www.tuiasi.ro](http://www.tuiasi.ro)



# What is MATLAB?

- A programming language for numerical computation
- A very powerful tool for any engineer, scientist or economist
- Its name stands for MATrix LABoratory

# Why use MATLAB?

- Very intuitive and easy to learn syntax, compared to other programming languages
- Many useful functions, just like Excel
  - built-in
  - provided into supplementary packages called toolboxes
- Easy debugging

# Why use MATLAB?

- **Built-in capabilities:**
  - Friendly user interface for beginners, packed with useful features for advanced users
  - Text editor for building algorithms, using its proprietary programming language
  - Easy to understand syntax
  - Easy data importing and exporting
  - Compatibility with Microsoft Excel, CSV and text files
  - Many useful functions for number, matrix and text processing
  - Easy graph plotting
  - GUI (Graphical User Interface) creation capabilities

# Content

## Part One: Get to know MATLAB

- The Main Window
- The Program Editor

## Part Two: Warm up

- How to define, save and load data
- Some useful functions and statements of general purpose

## Part Three: Writing programs in MATLAB

- Scripts
- Functions
- The **if**, **for**, **while**, **case** statements

# Content

## **Part Four:** Out-of-the-box numerical processing

- Matrix manipulation

## **Part Five:** Graphical Representations

- Line and scatter graphs

## **Part six:** Hands-on programming

- We write together a small program which uses functions, loops, if and graphical representations
- Demo: a slot machines game written in MATLAB

# Part One

---

## Meet MATLAB

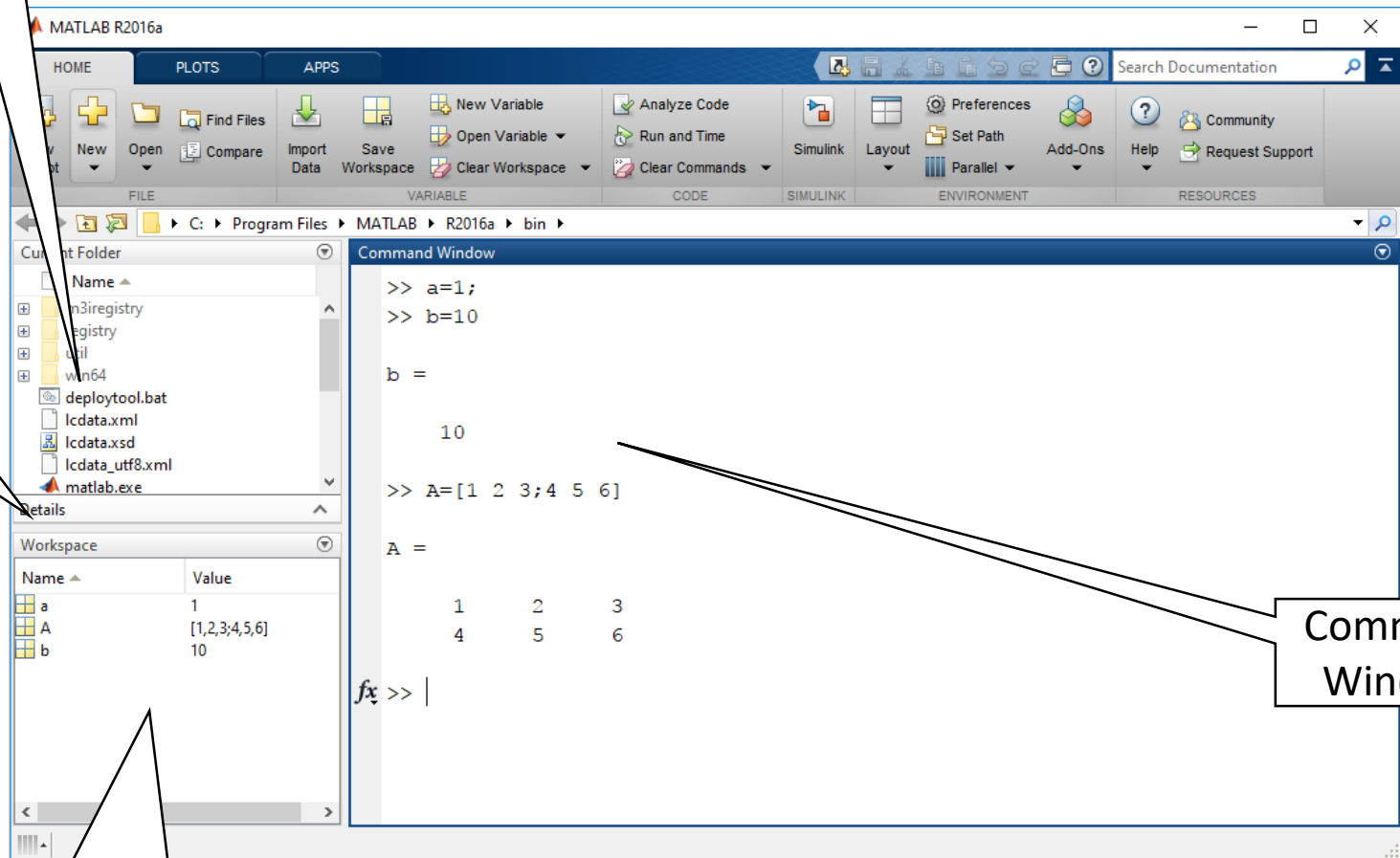
The Main Window  
The Program Editor

# The Main Window

Current Folder

Details  
(hidden)

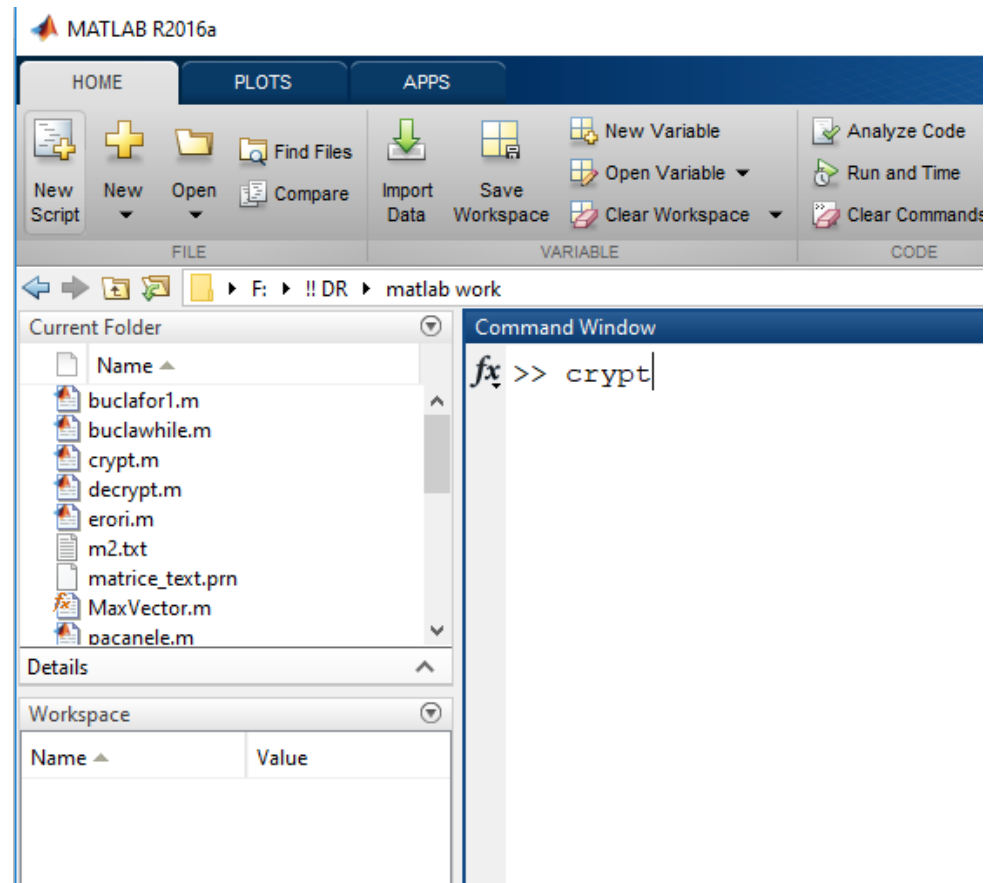
Workspace



Command Window

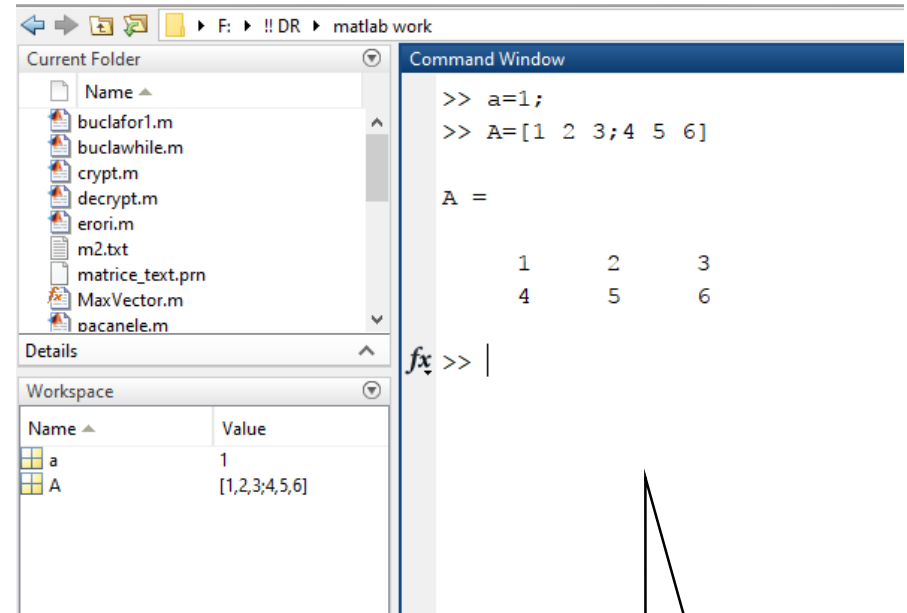
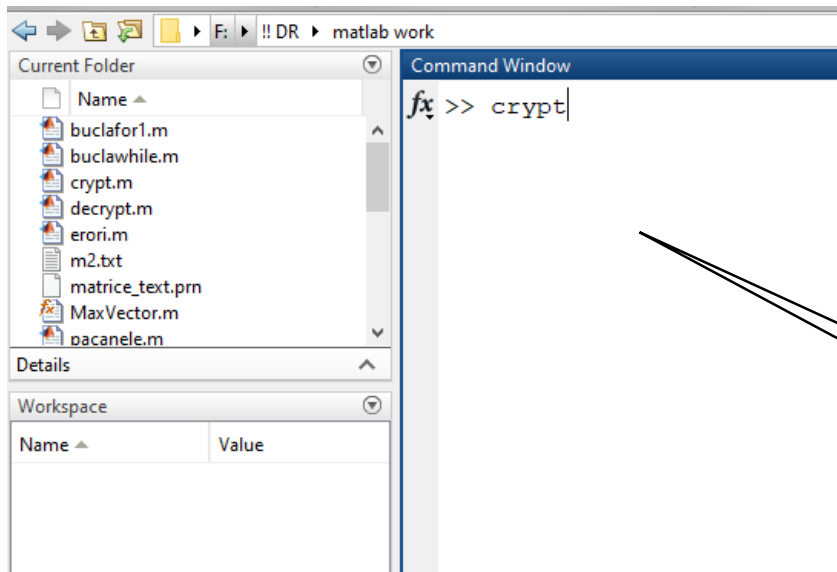
# Current Folder

- Your user-defined programs will run only if MATLAB is in their folder
  - Or if the folder is in MATLAB's search path
- Data files will be saved in the current folder



# The Command Window

- Can be used for
  - Creating variables
  - Writing code
  - Running programs
  - Debugging

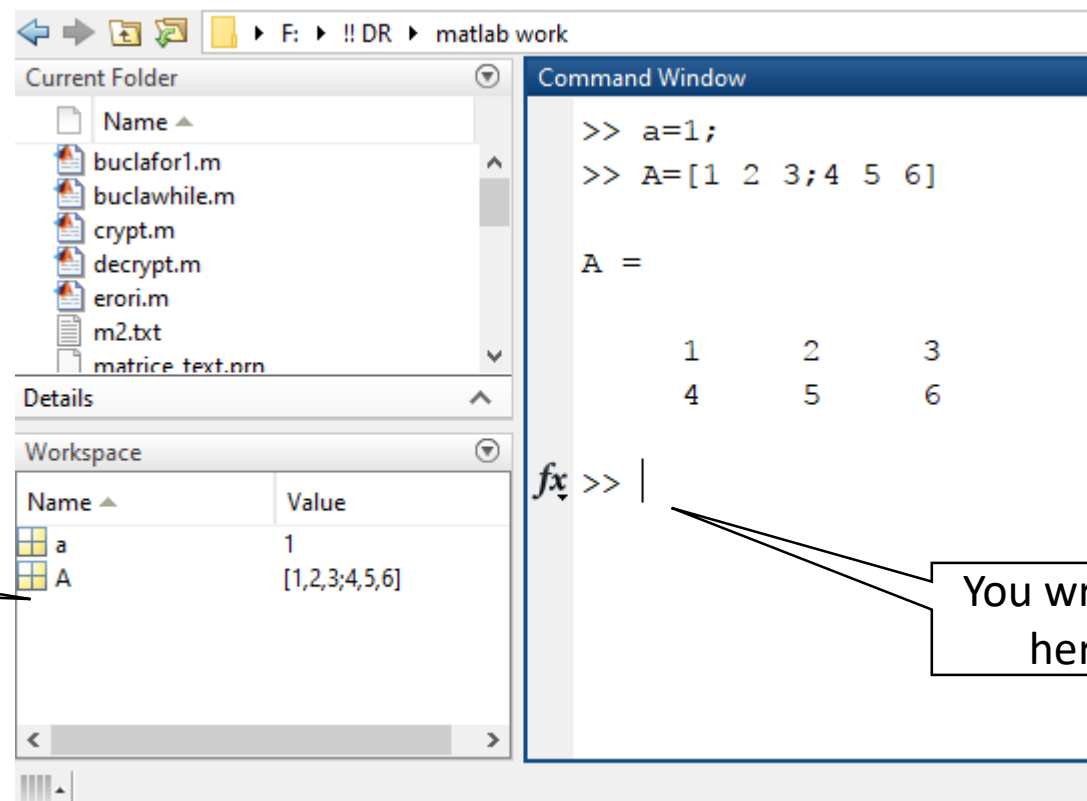


defining  
variables

running  
a script

# Workspace

- Any variable that you define in the Command Window or is created when running a program will show up in the Workspace
  - Different graphic symbols for different variable types



It shows up  
here

You write it  
here

# Workspace

- You can view and edit variables
  - Copy-Paste from and to Excel works from the Variables window !

The screenshot displays the MATLAB environment with the following components:

- Current Folder:** Lists files such as `buclafor1.m`, `buclawhile.m`, `crypt.m`, `decrypt.m`, `erori.m`, `m2.txt`, `matrice_text.prn`, `MaxVector.m`, `pacanele.m`, and `pacanele2.m`.
- Workspace:** A table showing variables in the workspace:

Name	Value
a	1
A	[1,2,3;4,5,6]

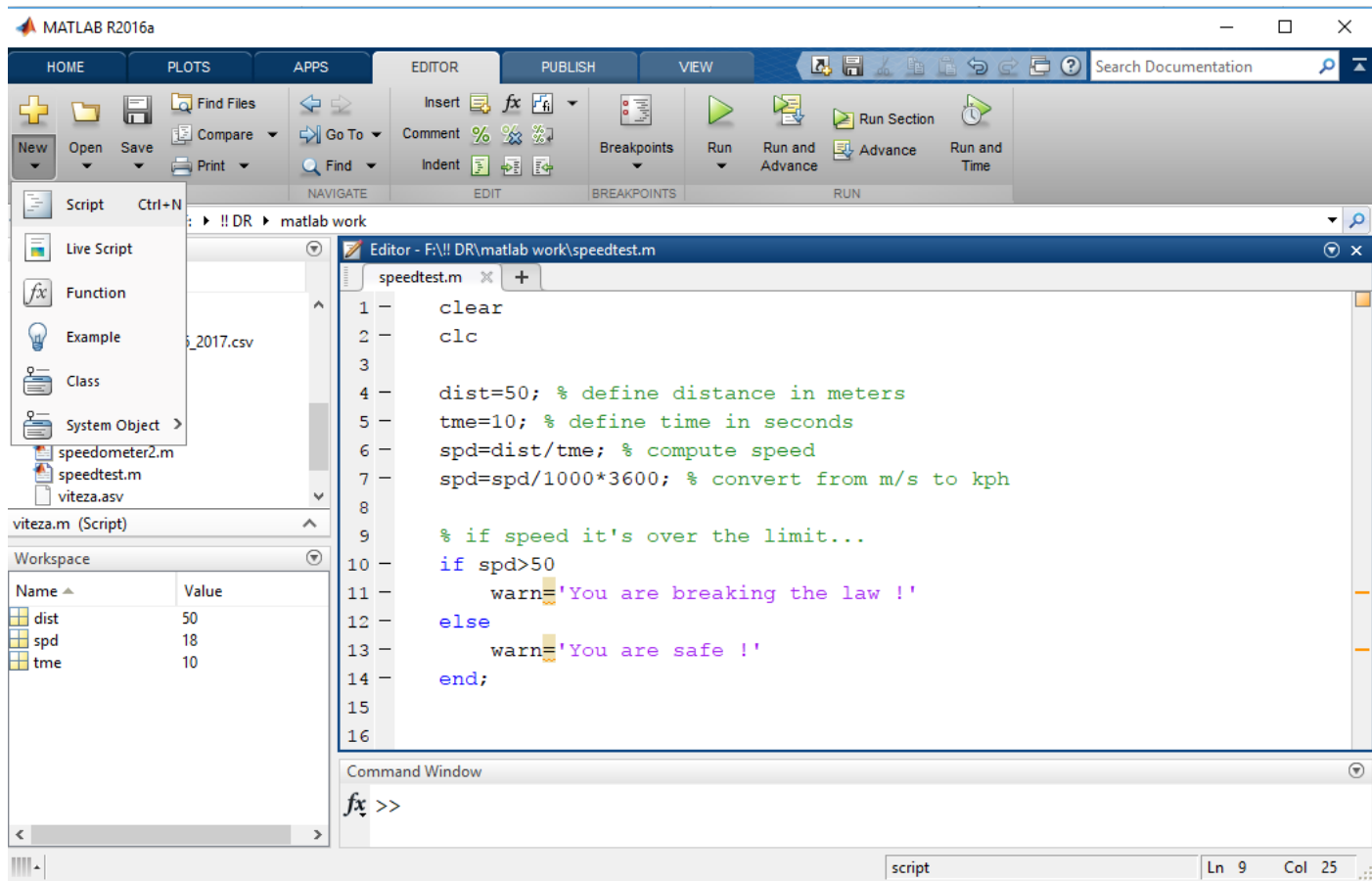
A callout box points to the variable `A` with the text: "You double click here".
- Variables - A:** A window showing the variable `A` as a 2x3 double matrix. A callout box points to the matrix content with the text: "It shows up here".

	1	2	3	4
1	1	2	3	
2	4	5	6	
3				
4				
5				
6				
7				
8				
9				
- Command Window:** Shows the command `A =` followed by the matrix values:

```
A =  
     1     2     3  
     4     5     6  
fx >>
```

# The Program Editor

- Writing code in the Command Window is cumbersome
- A better tool is the Program Editor



# The Program Editor

- syntax highlighting

Black is normal code

Green is a comment, defined with %

Blue is a reserved word

Pink is text, placed between prime symbols

```
clear
clc

dist=50; % define distance in meters
tme=10; % define time in seconds
spd=dist/tme; % compute speed
spd=spd/1000*3600; % convert from m/s to kph

% if speed it's over the limit...
if spd>50
    warn='You are breaking the law !'
else
    warn='You are safe !'
end;
```

# The Program Editor

Warning colors for errors:

- **Green:** code with no errors or warnings, all OK
- **Orange:** Warning, better performance can be achieved if the code is changed
  - The program can run properly
- **Red:** fatal error, wrong syntax in code
  - The program cannot run at all

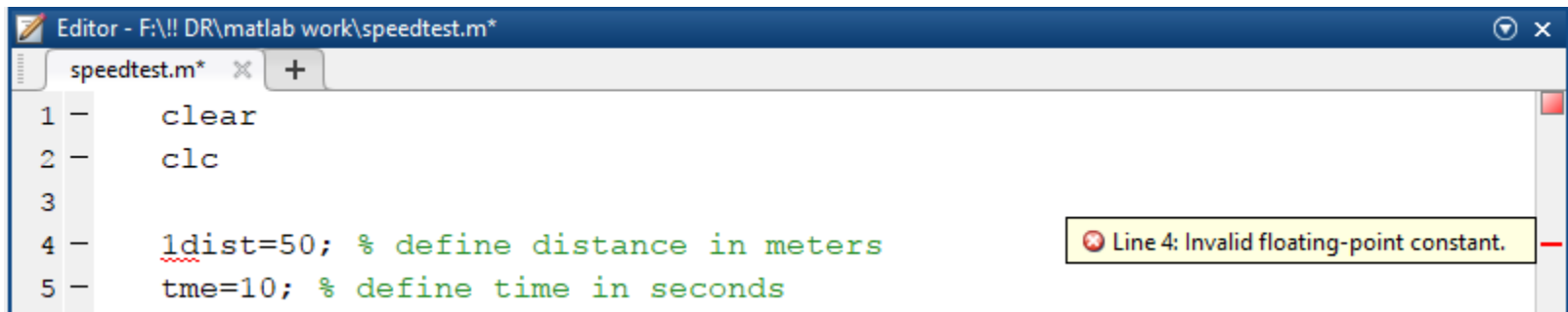
```
Editor - F:\!! DR\matlab work\speedtest.m
speedtest.m
1 clear
2 clc
3
4 dist=50; % define distance in meters
5 tme=10; % define time in seconds
6 spd=dist/tme; % compute speed
7 spd=spd/1000*3600; % convert from m/s to kph
8
9 % if speed it's over the limit...
10 if spd>50
11     warn='You are over the limit!'
12 else
13     warn='You are safe !'
14 end;
15
16
```

Line 11: Terminate statement with semicolon to suppress output (within a script). Details Fix



# The Program Editor

- A fatal error in the code:
  - Hover the mouse on the red line to see the error

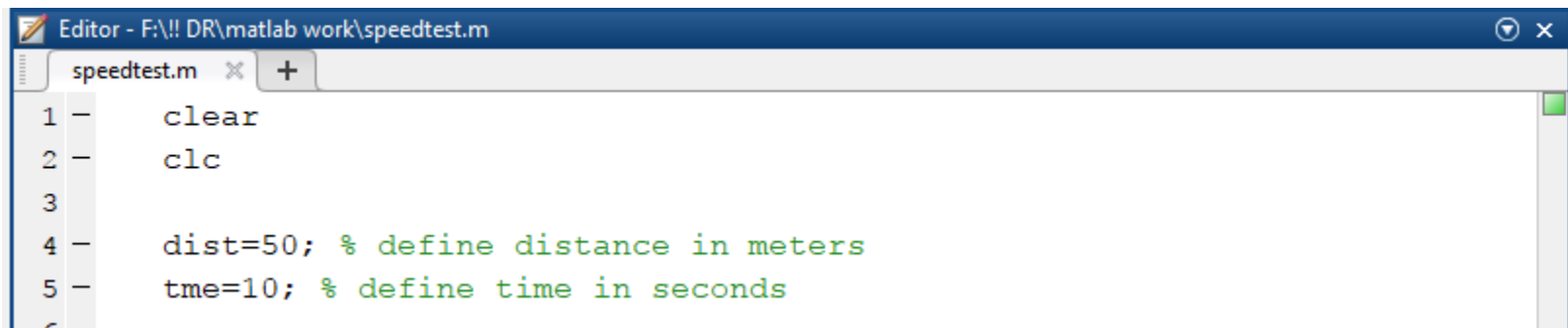


The screenshot shows the MATLAB Editor window for a file named 'speedtest.m'. The code is as follows:

```
1 - clear
2 - clc
3
4 - ldist=50; % define distance in meters
5 - tme=10; % define time in seconds
```

A red squiggly line is under the 'l' in 'l' on line 4. A yellow tooltip box with a red 'x' icon is displayed, containing the text: "Line 4: Invalid floating-point constant." The status bar on the right shows a red line, indicating a fatal error.

- A perfect sequence of code:
  - Syntax-wise, not program-wise



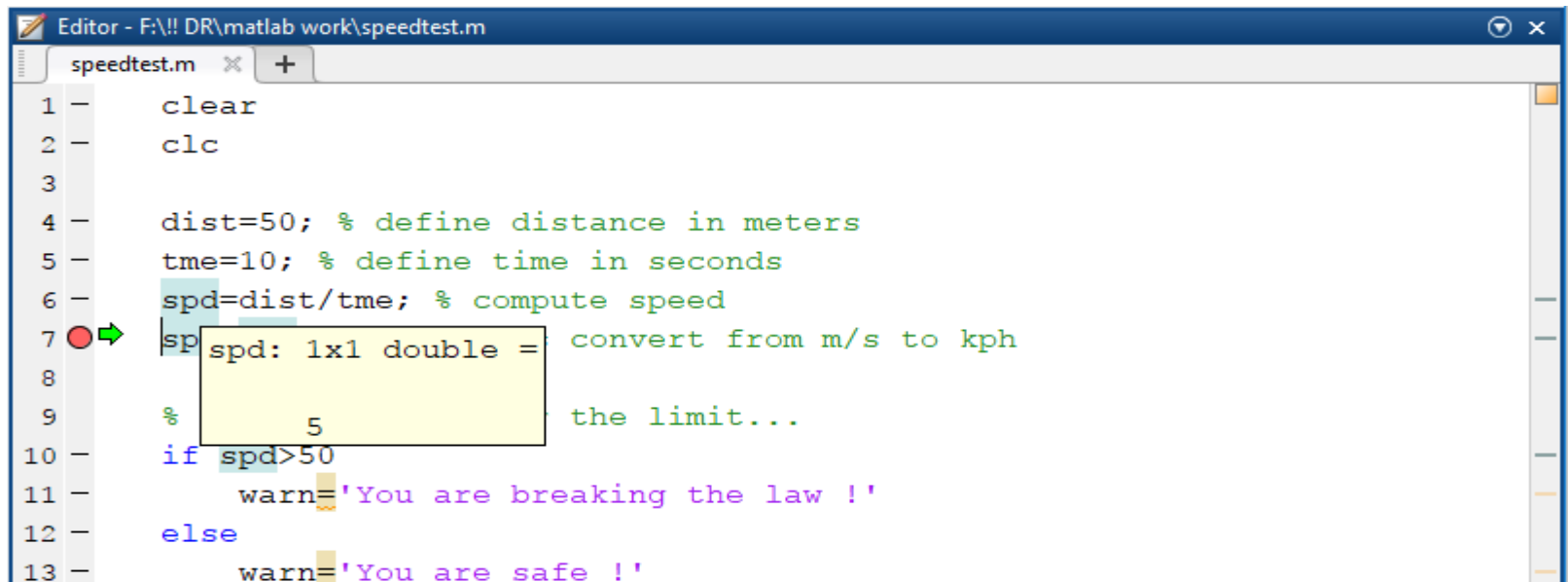
The screenshot shows the MATLAB Editor window for the same file 'speedtest.m', but with the error corrected. The code is now:

```
1 - clear
2 - clc
3
4 - dist=50; % define distance in meters
5 - tme=10; % define time in seconds
6
```

The 'l' has been removed from 'l' on line 4. The status bar on the right now shows a green line, indicating no errors.

# The Program Editor

- You can use the Program Editor to enable breakpoints and debug your code
  - Click on the horizontal black lines to enable red breakpoints
  - Use F5 to run the program until breakpoint line
  - Use F10 to run the program step by step
  - Hover on the variable names to see their values
  - Exit debugging with Shift+F5



The screenshot shows the MATLAB Program Editor window titled "Editor - F:\!! DR\matlab work\speedtest.m". The script "speedtest.m" is open, showing the following code:

```
1 - clear
2 - clc
3
4 - dist=50; % define distance in meters
5 - tme=10; % define time in seconds
6 - spd=dist/tme; % compute speed
7 - sp=spd; % convert from m/s to kph
8
9 - % 5 the limit...
10 - if spd>50
11 -     warn='You are breaking the law !'
12 - else
13 -     warn='You are safe !'
```

A red circle with a green arrow points to the line number 7, indicating a breakpoint. A yellow tooltip box is displayed over the variable "spd" on line 7, showing its value: "spd: 1x1 double = 5".

- **The Main Window**
  - Current Directory
  - The Command Window
  - The Workspace
- **The Program Editor**
  - Syntax highlighting
  - Debugging
- A small piece of code

***End of Part One - Review***

- **Feel free to contact me with questions !**

**Ovidiu Ivanov**

Lecturer, PhD

The 'Gheorghe Asachi' Technical University

The Electrical Engineering Faculty

[www.tuiasi.ro](http://www.tuiasi.ro)

[oviduiuivanov@tuiasi.ro](mailto:oviduiuivanov@tuiasi.ro)



**THANK YOU !**