

Certificate Course

On

Fundamentals of MATLAB Programming

13.09.2021 to 25.09.2021

Coordinator: T.Kishore Kumar



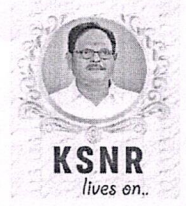
K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Lr./KSRMCE/ (Department of EEE)/2021-22/

Date: **03.09.2021**

To

The Principal,
K.S.R.M.College of Engineering
Kadapa.

Respected Sir,

//THROUGH PROPER CHANNEL//

Sub: \

KSRMCE - (Department of EEE) – Financial Assistance for Certification Course – Requested - Reg.

It is being brought to your notice that, Department of Electrical and Electronics Engineering of our organization is planning to conduct a certification course on **Fundamentals of MATLAB Programming** for V Sem students. Details of Syllabus, Students registered and Schedule are over leafed for your kind reference. In this regard we kindly request you to provide financial assistance to procure the needs and for smooth completion of the program, for which kind of act we would be grateful to you sir.

Thanking you

Yours Faithfully
K. Kalyan Kumar
Sri K. Kalyan akumar

(Course Coordinators Dept., of EEE)

*forwarded to
the principal/GV
Maddy*

*Permitted
H. S. S. Murthy*



/ksrmce.ac.in

Follow Us:



/ksrmceofficial



**K.S.R.M. COLLEGE OF ENGINEERING
(UGC-AUTONOMOUS)**

Kadapa, Andhra Pradesh, India – 516 005

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.
An ISO 14001:2004 & 9001: 2015 Certified Institution



Department of Electrical and Electronics Engineering

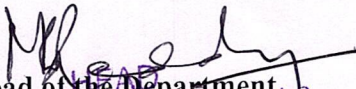
Cr./KSRMCE/(Department of E.E.E)/2021-22/

Date: 09.09.2021

Circular

It is hereby informed that the department of Electrical and Electronics Engineering is organizing a Certification Course on “**Fundamentals of MATLAB Programming** “ for B.Tech V Semester students, from 13.09.2021 to 25.09.2021., I request the students to register their names with the concerned coordinator on or before 12.09.2021. The details of the Workshop are as follows:

Name of the Event	Certification Course
Name of the Course	Fundamentals of MATLAB Programming
Date(s)	13.09.2021 to 25.09.2021
Resource persons	Mr.K.Kalyan Kumar, Assistant Professor in EEE,, KSRMCE.
Venue	SJ – 114 (Seminar Hall)
Faculty Coordinator	Mr.T. Kishore Kumar, Assistant Professor in EEE,, KSRMCE.


Head of the Department
Department of Electrical &
Electronics Engineering
K.S.R.M. College of Engineering
Kadapa -516003.



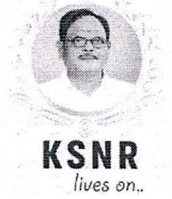
K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Date:13-09-2021

Name of the Event: Certification Course on Fundamentals of MATLAB Programming

Venue: SJ Block Computer Center

List of Participants

S.No	Roll Number	Name of the Student	Sem	Branch	Signature
1	199Y1A0204	B.P.EKSHITHA	V	EEE	B.P.Ekshitha
2	209Y5A0207	TIPPUGARI SHAHANSHA KHAN	V	EEE	T. shobansha Khan
3	199Y1A0236	N. INDU	V	EEE	N. Indu
4	199Y1A0251	T. JYOTHI BHARGAVI	V	EEE	T. Jyothi Bhargavi
5	209Y5A0203	KOTAGULA MOHAMMAD ALI	V	EEE	K. Mohammad Ali
6	209Y5A0205	SIDDI.MAHESH	V	EEE	S. Mahesh
7	199Y1A0253	VALLURU DIVYA TEJA	V	EEE	V. Divya Teja
8	199y1a0252	TIRUMALASETTY PRUDHULA	V	EEE	T. Prudhula.
9	199y1A0220	KALAVAPALLI PAVITHRA	V	EEE	K. Pavithra
10	199Y1A0211	DUDEKULA KARISHMA	V	EEE	D. Karishma
11	199Y1A0205	CHILUMURU MALLESWARI	V	EEE	C. Malleswari
12	209Y5A0202	C.RAKESH	V	EEE	C. Rakesh
13	209Y5A0201	BERI. SWATHI	V	EEE	B. Swathi
14	199Y1A0223	K. NIKHAT SULTANA	V	EEE	K. Nikhat Sultana
15	199Y1A0248	S. ASMA FARHEEN	V	EEE	S. Asma Farheen

1. K. Kalyan Kumar
Co-ordinator

2. A. Chakraborty

V. S. S. Murthy
Principal

K.S.R.M. COLLEGE OF ENGINEERING, KADAPA
(AUTONOMOUS)

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

Fundamentals of MATLAB Programming

(Certification course)

Course Objective: To acquire basic knowledge in MATLAB Programming to solve Electrical Engineering Problems.

Module-1: 10 hrs

Basic features: Introduction – Simple math – MATLAB Workspace – About variables – comments, punctuation and aborting execution – Script M-files.

Arrays and Array Operations: Simple arrays – Array addressing – Array construction – Scalar Array Mathematics - Array Array Mathematics – Array size.

Module-2: 10 hrs

Control Flow - Relational & Logical operators – For, While Loops, If-Else-End Construction.

Polynomials: Roots, multiplication, addition, division, derivatives and Integrals

Module-3: 10 hrs

Electrical Engineering Applications – Solving simple problems in Electrical Circuits, Electrical Machines, Control Systems and Power Systems.

Text books:

1. Mastering MATLAB by Hanselman, Littlefield – Pearson Publications, 1st Edition, 2012.
2. MATLAB Programming by David C. Kuncicky -Prentice Hall, 2004

Course Outcomes:

After successful completion of the course the students will be able to

CO1: understand the basic features of MATLAB Programming, Array construction methods, operations, Relational & Logical Operators.

CO2: Illustrate the Polynomial operations

CO3: Analyze the Control flow structures IF-ELSE, FOR and WHILE

CO4: Solve electrical engineering problems using MATLAB Programs

K.S.R.M. COLLEGE OF ENGINEERING, KADAPA
(AUTONOMOUS)

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

Fundamentals of MATLAB Programming

(Certification course)

Detailed Schedule for a total of 30 hours course

S.No	Date	Topic	No. of Hours
Module 1			
1	13/09/2021	Introduction –Simple math	02
2	14/09/2021	MATLAB Workspace – About variables – comments, punctuation and aborting execution	03
3	15/09/2021	Script M-files	03
4	16/09/2021	Simple arrays – Array addressing – Array construction –Scalar Array Mathematics	03
5	17/09/2021	Array Mathematics –Array size	03
Module 2			
6	18/09/2021	Relational & Logical operators	02
7	20/09/2021	For Loop	02
8	21/09/2021	While Loop	02
9	22/09/2021	If-Else-End Construction	02
10	23/09/2021	Polynomials	03
Module 3			
11	24/09/2021	Application to Electrical Engineering & problems	03
12	25/09/2021	Application to Electrical Engineering & problems	02
Total			30

Course coordinators and resource persons

Sri. K. Kalyan Kumar

K. Kalyan Kumar

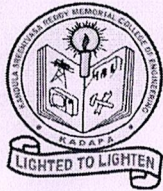
Sri. T. Kishore Kumar

T. Kishore Kumar

Agwal
HOD

HEAD

Department of Electrical &
Electronics Engineering
K.S.R.M. College of Engineering
Kadapa -516003.



K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



KSNR
lives on.

Name of the Event: Certification Course on Fundamentals of MATLAB Programming

Venue: SJ Block Computer Center

Attendance Sheet

S.No	Roll Number	Name of the Student	13/9	14/9	15/9	16/9	17/9	18/9	20/9	21/9	22/9	23/9	24/9	25/9
1	199Y1A0204	B.P.EKSHITHA	B.Ekshitha	B.Ekshitha	B.Ekshitha	B.Ekshitha	B.Ekshitha	B.Ekshitha	B.Ekshitha	B.Ekshitha	B.Ekshitha	B.Ekshitha	B.Ekshitha	B.Ekshitha
2	209Y5A0207	TIPPUGARI SHAHANSHA KHAN	T.Shah	T.Shah	T.Shah	T.Shah	T.Shah	T.Shah	T.Shah	T.Shah	T.Shah	T.Shah	T.Shah	T.Shah
3	199Y1A0236	N. INDU	N.Indu	N.Indu	N.Indu	N.Indu	N.Indu	N.Indu	N.Indu	N.Indu	N.Indu	N.Indu	N.Indu	N.Indu
4	199Y1A0251	T. JYOTHI BHARGAVI	T.Jyothi	T.Jyothi	T.Jyothi	T.Jyothi	T.Jyothi	T.Jyothi	T.Jyothi	T.Jyothi	T.Jyothi	T.Jyothi	T.Jyothi	T.Jyothi
5	209Y5A0203	KOTAGULA MOHAMMAD ALI	K.Ali	K.Ali	K.Ali	K.Ali	K.Ali	K.Ali	K.Ali	K.Ali	K.Ali	K.Ali	K.Ali	K.Ali
6	209Y5A0205	SIDDI.MAHESH	S.Mahesh	S.Mahesh	S.Mahesh	S.Mahesh	S.Mahesh	S.Mahesh	S.Mahesh	S.Mahesh	S.Mahesh	S.Mahesh	S.Mahesh	S.Mahesh
7	199Y1A0253	VALLURU DIVYA TEJA	V.Divya	V.Divya	V.Divya	V.Divya	V.Divya	V.Divya	V.Divya	V.Divya	V.Divya	V.Divya	V.Divya	V.Divya
8	199y1a0252	TIRUMALASETTY PRUDHULA	T.Prudhula	T.Prudhula	T.Prudhula	T.Prudhula	T.Prudhula	T.Prudhula	T.Prudhula	T.Prudhula	T.Prudhula	T.Prudhula	T.Prudhula	T.Prudhula
9	199y1A0220	KALAVAPALLI PAVITHRA	K.Pavithra	K.Pavithra	K.Pavithra	K.Pavithra	K.Pavithra	K.Pavithra	K.Pavithra	K.Pavithra	K.Pavithra	K.Pavithra	K.Pavithra	K.Pavithra
10	199Y1A0211	DUDEKULA KARISHMA	D.Karishma	D.Karishma	D.Karishma	D.Karishma	D.Karishma	D.Karishma	D.Karishma	D.Karishma	D.Karishma	D.Karishma	D.Karishma	D.Karishma
11	199Y1A0205	CHILUMURU MALLESWARI	C.Malleswari	C.Malleswari	C.Malleswari	C.Malleswari	C.Malleswari	C.Malleswari	C.Malleswari	C.Malleswari	C.Malleswari	C.Malleswari	C.Malleswari	C.Malleswari
12	209Y5A0202	C.RAKESH	C.Rakesh	C.Rakesh	C.Rakesh	C.Rakesh	C.Rakesh	C.Rakesh	C.Rakesh	C.Rakesh	C.Rakesh	C.Rakesh	C.Rakesh	C.Rakesh
13	209Y5A0201	BERI. SWATHI	B.Swathi	B.Swathi	B.Swathi	B.Swathi	B.Swathi	B.Swathi	B.Swathi	B.Swathi	B.Swathi	B.Swathi	B.Swathi	B.Swathi
14	199Y1A0223	K. NIKHAT SULTANA	K.Nikhath	K.Nikhath	K.Nikhath	K.Nikhath	K.Nikhath	K.Nikhath	K.Nikhath	K.Nikhath	K.Nikhath	K.Nikhath	K.Nikhath	K.Nikhath
15	199Y1A0248	S. ASMA FARHEEN	S.Asma	S.Asma	S.Asma	S.Asma	S.Asma	S.Asma	S.Asma	S.Asma	S.Asma	S.Asma	S.Asma	S.Asma

1. K. Kalpan Kumar
Co-Ordinator

2. [Signature]

[Signature]
H.O.D

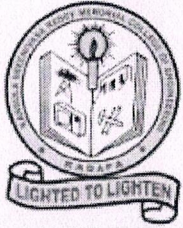
/krmce.ac.in

Follow Us:



/krmce

HEAD
Department of Electrical &
Electronics Engineering
K.S.R.M. College of Engineering
Kadapa -516003.



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

Department of Electrical and Electronics Engineering



KSNR
lives on..

Certification Course on Fundamentals of MATLAB programming

Date of Event:

13/09/2021 to 25/09/2021

Venue:

SJ Block Simulation Lab

Eligibility: EEE UG Students

Course Instructors:

Sri K. Kalyan Kumar, Assistant Professor

Sri T. Kishore Kumar, Assistant Professor



DR. K. AMARESH.
1100

DR. V.S.S. MURTHY
(Principal)

PROP. A. MOHAN
(Director)

SRI. K. CHANDRA OBUL REDDY
(Management Member)

SMT. K. RAJESWARI
(Correspondent Secretary, Treasurer)

SRI K. MADAN MOHAN REDDY
(Vice- chairman)

SRI. K. RAJA MOHAN REDDY
(Chairman)

f [ksrmceofficial](https://www.facebook.com/ksrmceofficial)

8143731980, 8575697569



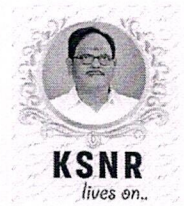
K.S.R.M. COLLEGE OF ENGINEERING

(AUTONOMOUS)

Kadapa, Andhra Pradesh, India – 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Department of Electrical & Electronics Engineering

Activity Report

Name of the Certification Course	:	Fundamentals of MATLAB Programming
Duration of the Course	:	30Hrs
Scheduled Date/Time	:	13.9.21 to 25.09.21 (3:00 to 5.00pm)
Target Audience	:	B. Tech., III, V and VII Semester Students
No. of Students participated	:	15
Resource Person	:	Mr. K. Kalyan Kumar, Assistant Professor Mr. T. Kishore Kumar, Assistant Professor
Venue	:	SJ 114 (Simulation Lab)

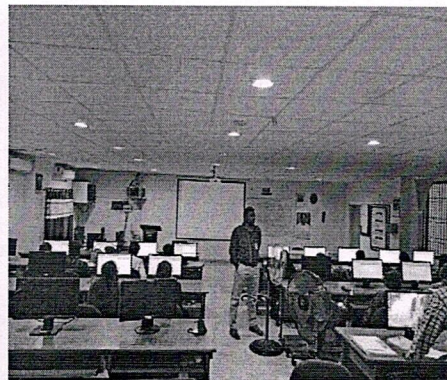
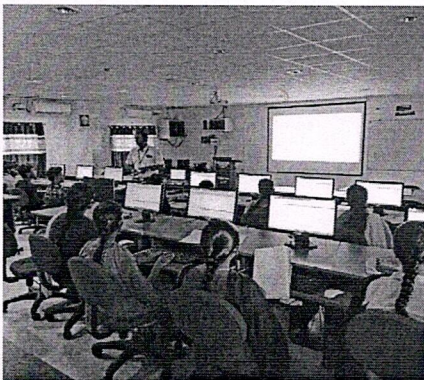
Activity Description:

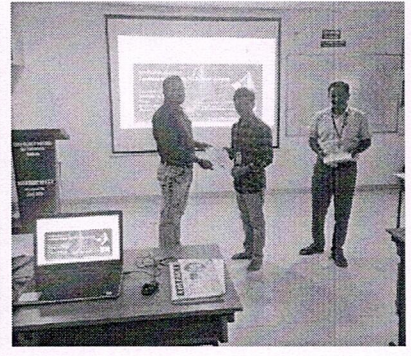
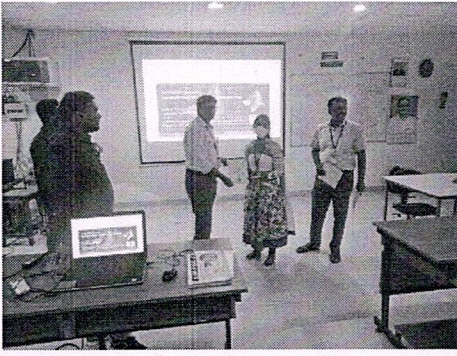
The availability of technical computing environment such as MATLAB is now reshaping the role and applications of computer laboratory projects to involve students in more intense problem-solving experience. This availability also provides an opportunity to easily conduct numerical experiments and to tackle realistic and more complicated problems.

“Fundamentals of MATLAB Programming” is an introductory course in MATLAB and technical computing. It focuses on the specific fundamental features of MATLAB that are useful for engineering classes. This hands on experience course allows students to become familiar with MATLAB to solve application problems.

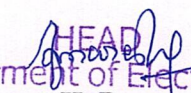
By the end of this Course, Students are able to use MATLAB for interactive computations, become familiar with memory and file management in MATLAB and able to program scripts using the MATLAB development environment.

Photos:






Co-ordinators


HEAD
Department of Electrical &
Electronics Engineering
HoD
S.S.R.M. College of Engineering
Kadapa -516003.



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India - 516 001

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.



KSNR
lives on.

Department of EEE

COURSE COMPLETION CERTIFICATE

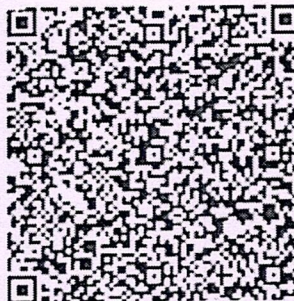
ON

Fundamentals of MATLAB Programming

This is to certify that **K. Nikhat Sultana** has participated in " **Certificate course on Fundamentals of MATLAB Programming** ", during 13/09/2021 to 25/09/2021 organized by the Department of Electrical and Electronics Engineering, K.S.R.M.College of Engineering (Autonomous), Kadapa.

Dr. K. Amaresh

Dr. K. Amaresh
Head of the Department



V. S. S. Murthy
Dr. V.S.S.Murthy
Principal

Feedback on Certification Course “Fundamentals of MATLAB Programming”

* Required

1. Roll Number *

2. Name of the Student *

3. Organization of Course and session planning by instructor. *

Mark only one oval.

Good

Very Good

Excellent

4. Clarity in content delivery. *

Mark only one oval.

Good

Very Good

Excellent

5. Content is relevant and useful *

Mark only one oval.

Good

Very Good

Excellent

6. Adequate opportunity to interact with trainer *

Mark only one oval.

Poor

Fair

Good

very Good

Excellent

7. Overall rating *

Mark only one oval.

Poor

Good

Very Good

Excellent

This content is neither created nor endorsed by Google.

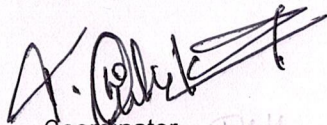
Google Forms

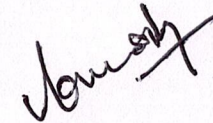
**K.S.R.M. COLLEGE OF ENGINEERING
(AUTONOMOUS)**

Department of Electrical & Electronics Engineering

Feedback of students on Certification Course on "Fundamentals of MATLAB Programming"

S.No	Roll Number	Name of the Student	Organization of Course and session planning by instructor.	Clarity in content delivery.	Content is relevant and useful	Adequate opportunity to interact with trainer	Overall rating
S.No	Roll Number	Name of the Student	Good	Good	Good	Very good	Good
1	199Y1A0204	B.P.EKSHITHA	Excellent	Excellent	Excellent	Excellent	Excellent
2	209Y5A0207	TIPPUGARI SHAHANSHA KHAN	Excellent	Excellent	Excellent	Excellent	Excellent
3	199Y1A0236	N. INDU	Excellent	Excellent	Excellent	Very good	Excellent
4	199Y1A0251	T. JYOTHI BHARGAVI	Very good	Very good	Very good	Fair	Very good
5	209Y5A0203	KOTAGULA MOHAMMAD ALI	Good	Good	Good	Good	Good
6	209Y5A0205	SIDDI.MAHESH	Excellent	Excellent	Excellent	Excellent	Excellent
7	199Y1A0253	VALLURU DIVYA TEJA	Excellent	Excellent	Excellent	Excellent	Excellent
8	199y1a0252	TIRUMALASETTY PRUDHULA	Excellent	Very good	Very good	Excellent	Excellent
9	199y1A0220	KALAVAPALLI PAVITHRA	Excellent	Very good	Excellent	Very good	Excellent
10	199Y1A0211	DUDEKULA KARISHMA	Very good	Very good	Very good	Very good	Very good
11	199Y1A0205	CHILUMURU MALLESWARI	Poor	Poor	Excellent	Poor	Poor
12	209Y5A0202	C.RAKESH	Excellent	Excellent	Excellent	Excellent	Excellent
13	209Y5A0201	BERI. SWATHI	Very good	Very good	Excellent	Very good	Excellent
14	209Y5A0223	K. NIKHAT SULTANA	Fair	Excellent	Excellent	Excellent	Excellent
15	199Y1A0248	S. ASMA FARHEEN	Excellent	Excellent	Excellent	Excellent	Excellent


Coordinator


HOD
HEAD

Department of Electrical &
Electronics Engineering
K.S.R.M. College of Engineering
Kadapa -516003.

BASICS OF MATLAB

1.1. Basic features:

Running MATLAB creates one or more windows on your computer monitor. One of these windows, entitled *MATLAB*, is commonly called the *MATLAB desktop*. This window is the primary graphical user interface for MATLAB. Within the *MATLAB* window, there is a window called the *Command* window, which is the primary place wherein you interact with MATLAB. The prompt `>>` is displayed in the *Command* window, and when the *Command* window is active, a blinking cursor appears to the right of the prompt. This cursor and prompt signify that MATLAB is waiting to perform a mathematical operation.

1.1.1. Simple MATH:

Just like a calculator, MATLAB can do basic math. Consider the following simple example: Mary goes to the office supply store and buys five pens at 30 cents each, seven notebooks at 60 cents each, and one pair of scissors for 70 cents. How many items did Mary buy, and how much did they cost?

To solve this problem with a calculator, you enter

$$5 + 7 + 1 = 13 \text{ items}$$

$$5 \times 30 + 7 \times 60 + 1 \times 70 = 640 \text{ cents}$$

In MATLAB, this problem can be solved in a number of different ways. First, the calculator approach can be taken:

```
>> 5+7+1
```

```
ans =
```

```
13
```

```
>> 5*30 + 7*60 + 1*70
```

```
ans =
```

```
640
```

Note that MATLAB doesn't care about spaces, for the most part, and that multiplication takes precedence over addition. Note also that MATLAB calls the result `ans`, which is short for *answer* for both computations.

As an alternative, the problem can be solved by storing information in *MATLAB variables*:

```
>> pens = 5
```

```
pens =
```

```
5
```

```
>> notebooks = 7
```

```
notebooks =
```

```
7
```

```
>> scissors = 1;
```

```
>> items = pens + notebooks + scissors
```

```
items =
```

```
13
```

```
>> cost = pens*30 + notebooks*60 + scissors*70
```

```
cost =
```

```
640
```


Here, we created three MATLAB variables—pens, notebooks, and scissors—to store the number of each item. After entering each statement, MATLAB displayed the results, except in the case of scissors. The semicolon at the end of the line tells MATLAB to evaluate the line, but not to display the answer. Finally, rather than calling the results ans, we told MATLAB to call the number of items purchased items and the total price paid cost. At each step, MATLAB remembered past information. Because MATLAB remembers things, let's ask what the average cost per item was:

```
>> average_cost = cost/items
average_cost =
49.2308
```

Since the term *average cost* is two words and MATLAB variable names must be one word, an underscore was used to create the single MATLAB variable `average_cost`.

In all, MATLAB offers the following basic arithmetic operations:

Operation	Symbol	Example
Addition	+	3 + 22
Subtraction	-	54.4 - 16.5
Multiplication	*	3.14 * 6
Division	/ or \	19.54/7, or 7\19.54
Exponentiation	^	2^8

1.1.2. THE MATLAB WORKSPACE

As you work in the *Command* window, MATLAB remembers the commands you enter, as well as the values of any variables you create. These commands and variables are said to reside in the *MATLAB workspace* or *base workspace*, and they can be recalled whenever you wish. For example, to check the value of scissors, all you have to do is ask MATLAB for it by entering its name at the prompt:

```
>> scissors
scissors =
1
```

If you can't remember the name of a variable, you can ask MATLAB for a list of the variables it knows by using the MATLAB command `who`:

```
>> who
Your variables are:
ans  items  scissors
average_cost  notebooks
cost  pens
```

Note that MATLAB doesn't tell you the value of all of the variables; it merely gives you their names. To find their values, you must enter their names individually at the MATLAB prompt.

1.1.3. ABOUT VARIABLES

Like any other computer language, MATLAB has rules about variable names. Earlier, it was noted that variable names must be a single word containing no spaces. More specifically, MATLAB variable-naming rules are listed as follows.

Variable-Naming Rules	Comments/Examples
Variable names are case sensitive.	Cost, cost, CoSt, and COST are all different MATLAB variables.
Variable names can contain up to <code>nameLengthmax</code> characters (63 as of version 7.12). Any characters beyond the 63rd are ignored.	Howaboutthisvariablename
Variable names must start with a letter, followed by any number of letters, digits, or underscores.	how_about_this_variable_nameX51483
Punctuation characters are not allowed, because many of them have special meanings in MATLAB.	a_b_c_d_e

There are some specific exceptions to these naming rules. MATLAB has several names that cannot be used for variables. These names are *keywords* and form a *reserved word list* for MATLAB:

Reserved Word List
for end if while function return elseif case otherwise classdef switch continue else try catch global persistent break parfor spmd

This list is returned as an output of the `iskeyword` function. MATLAB will report an error if you try to use a reserved word as a variable. However, you can use words similar to keywords by capitalizing one or more letters. The function `isvarname('teststring')` returns True (1) if the character-string argument 'teststring' is a valid variable name; otherwise, it returns False (0).

1.1.4. COMMENTS, PUNCTUATION, AND ABORTING EXECUTION

As we saw earlier, placing a semicolon at the end of a command suppresses printing of the computed results. This feature is especially useful for suppressing the results of intermediate calculations. For instance,

```
>> pens = 9
pens =
9
>> items = pens + notebooks + scissors;
>> cost = pens*30 + notebooks*60 + scissors*70;
>> average_cost = cost/items
average_cost =
44.7059
```


displays the average cost of the items that Mary bought when she purchased nine pens, rather than the original five. The intermediate results items and cost were not printed, because semicolons appear at the ends of the commands defining them.

In addition to semicolons, MATLAB uses other punctuation symbols. For example, all text after a percent sign (%) is taken as a comment statement:

```
>> scissors = 1 % number of pairs of scissors purchased
```

The variable scissors is given the value of 1, and MATLAB simply ignores the percent sign and all text following it.

If they are separated by commas or semicolons, multiple commands can be placed on one line:

```
>> pens = 6, notebooks = 6; scissors = 2
```

```
pens =
```

```
6
```

```
scissors =
```

```
2
```

Commas tell MATLAB to display results; semicolons suppress printing. Sometimes, expressions or commands are so long that it is convenient to continue them onto additional lines. In MATLAB, statement continuation is denoted by three periods in succession, as shown in the following code:

```
>> average_cost = cost/items % command as done earlier
```

```
average_cost =
```

```
44.7059
```

```
>> average_cost = cost/... % command with valid continuation
```

```
items
```

```
average_cost =
```

```
44.7059
```

```
>> average_cost = cost... % command with valid continuation
```

```
/items
```

```
average_cost =
```

```
44.7059
```

```
>> average_cost = cost... command with valid continuation (no % needed)
```

```
/items
```

```
average_cost =
```

```
44.7059
```

```
>> average_cost = cost/it... % command with Invalid continuation
```

```
ems
```

```
??? ems
```

```
|
```

Error: Missing MATLAB operator.

Note that statement continuation works if the three periods appear between variable names and mathematical operators, but not in the middle of a variable name. That is, variable names cannot be split between two lines. Furthermore, all text after the three periods is considered to be a comment, so no percent symbol is needed.

Finally, MATLAB processing can be interrupted at any time by pressing **Control-C** (i.e., pressing the **Ctrl** and **C** keys simultaneously).

1.1.5. COMPLEX NUMBERS

One of the most powerful features of MATLAB is that it does not require any special handling for complex numbers. Complex numbers are formed in MATLAB in several ways. Examples of complex numbers include the following:

```
>> c1 = 1-2i % the appended i signifies the imaginary part
c1 =
1.0000 - 2.0000i
>> c1 = 1-2j % j also works
c1 =
1.0000 - 2.0000i
>> c1 = complex(1,-2) % a function that creates complex numbers
c1 =
1.0000 - 2.0000i
>> c2 = 3*(2-sqrt(-1)*3)
c2 =
6.0000 - 9.0000i
>> c3 = sqrt(-2)
c3 =
0 + 1.4142i
>> c4 = 6+sin(.5)*1i
c4 =
6.0000 + 0.4794i
>> c5 = 6+sin(.5)*1j
c5 =
6.0000 + 0.4794i
```

In the last two examples, multiplication by $1i$ and $1j$ are used to form the imaginary part. Multiplication by $1i$ or $1j$ is required in these cases, since $\sin(.5)i$ and $\sin(.5)j$ have no meaning in MATLAB. Termination with the characters i and j , as shown in the first two examples above, works only with numbers, and not with expressions. In MATLAB, the conversion between polar and rectangular forms makes use of the functions `real`, `imag`, `abs`, and `angle`:

```
>> c1
c1 =
1.0000 - 2.0000i
>> mag_c1 = abs(c1) % magnitude
mag_c1 =
2.2361
>> angle_c1 = angle(c1) % angle in radians
angle_c1 =
```



```

-1.1071
>> deg_c1 = angle_c1*180/pi % angle in degrees
deg_c1 =
-63.4349
>> real_c1 = real(c1) % real part
real_c1 =
1
>> imag_c1 = imag(c1) % imaginary part
imag_c1 =
-2

```

The MATLAB function `abs` computes the magnitude of complex numbers or the absolute value of real numbers, depending on which one you assign it to compute. Likewise, the MATLAB function `angle` computes the angle of a complex number in radians. *MATLAB does not natively perform trigonometric operations with units of degrees; however, basic trigonometric functions supporting angles in degrees are provided in MATLAB.*

1.2.Script M-files:

For simple problems, entering your requests at the MATLAB prompt in the *Command* window is fast and efficient. However, as the number of commands increases, or when you wish to change the value of one or more variables and reevaluate a number of commands, typing at the MATLAB prompt quickly becomes tedious. MATLAB provides a logical solution to this problem. It allows you to place MATLAB commands in a simple text file and then tells MATLAB to open the file and evaluate the commands exactly as it would have if you had typed the commands at the MATLAB prompt. These files are called *script files* or *M-files*. The term *script* signifies that MATLAB simply reads from the script found in the file. The term *M-file* means that script filenames must end with the extension '.m', as in, for example, `example1.m`.

1.2.1. SCRIPT M-FILE USE

To create a script M-file, click on the blank page icon on the MATLAB desktop toolbar, or choose **New** from the **File** menu and select **M-file**. This procedure brings up a text editor window wherein you can enter MATLAB commands. The following script M-file shows the commands from an example considered earlier:

```

% script M-file example1.m
pens = 5; % number of each item
notebooks = 7;
scissors = 1;
items = pens + notebooks + scissors
cost = pens*30 + notebooks*60 + scissors*70
average_cost = cost/items

```

This file can be saved to disk and executed immediately by (1) choosing **Save and Run** from the **Debug** menu, (2) pressing the **Save and Run** button on the *Editor* toolbar, or (3) simply pressing the function key **F5**. Alternatively, you can save this file as the M-file `example1.m` on your disk by choosing **Save** from the **File** menu; then, at the MATLAB prompt, it's just a matter of typing the name of the script file without the .m extension:

```
>> example1
```



```
items =
13
cost =
640
average_cost =
49.2308
```

When MATLAB interprets the `example1` statement, it follows the search path described in Chapter 3. In brief, MATLAB prioritizes current MATLAB variables ahead of M-file names. If `example1` is not a current MATLAB variable or a built-in function name, MATLAB opens the file `example1.m` (if it can find it) and evaluates the commands found there just as if they had been entered directly at the *Command* window prompt. As a result, commands within the M-file have access to all of the variables in the MATLAB workspace, and all of the variables created by the M-file become part of the workspace. Normally, the M-file commands are not displayed as they are evaluated. The `echo` command tells MATLAB to display, or *echo*, commands to the *Command* window as they are read and evaluated. You can probably guess what the `echo off` command does. Similarly, the command `echo` by itself toggles the `echo` state.

Because of the utility of script files, MATLAB provides several functions that are particularly helpful when used in M-files:

Function	Description
<code>beep</code>	Makes computer sound a beep
<code>disp(variablename)</code>	Displays results without identifying variable names
<code>echo</code>	Controls <i>Command</i> window echoing of script file contents as they are executed
<code>input</code>	Prompts user for input
<code>keyboard</code>	Temporarily gives control to keyboard. (Type return to return control to the executing script M-file.)
<code>pause</code> or <code>pause(n)</code>	Pauses until user presses any keyboard key, or pauses for <code>n</code> seconds and then continues
<code>waitforbuttonpress</code>	Pauses until user presses any keyboard key or any mouse button over a figure window

When a MATLAB command is not terminated by a semicolon, the results are displayed in the *Command* window, with the variable name identified. For a prettier display, it is sometimes convenient to suppress the variable name. In MATLAB, this is accomplished with the command `disp`:

```
>> items
items =
13
>> disp(items)
13
```


Rather than repeatedly edit a script file for computations for a variety of cases, you can employ the input command to prompt for input as a script file is executed. For example, reconsider the example1.m script file, with the following modifications:

```
% script M-file example1.m
pens = 5; % Number of each item
notebooks = 7;
scissors = input('Enter the number of pairs of scissors purchased > ');
items = pens + notebooks + scissors
cost = pens*30 + notebooks*60 + scissors*70
average_cost = cost/items
```

Running this script M-file produces this result:

```
>> example1
Enter the number of pairs of scissors purchased > 1
items =
13
cost =
640
average_cost =
49.2308
```

In response to the prompt, the number 1 was entered and the **Return** or **Enter** key was pressed. The remaining commands were evaluated as before. The function input accepts any valid MATLAB expression for input.

1.3. BLOCK COMMENTS AND CODE CELLS

In prior versions of MATLAB, comments were line oriented. That is, comments began with an initial percent sign and continued to the end of the current line. To continue the comments on the next line required another initial percent sign.

Therefore, a block of comments would all start with initial percent signs, as shown here:

```
% This is an example of multiple line comments
% in an M-file. Each line requires an initial % sign, or MATLAB
% assumes the line contains code to be executed.
```

While the block of comments is visually simple, it can become cumbersome to manage later, when comment text is augmented or edited. In this case, the percent signs must remain at the beginning of the lines, and the comment text must flow after the initial percent signs on each line. In the past, to make it less cumbersome, the MATLAB editor included commands for adding or removing the initial percent signs on a highlighted block of lines. Now, MATLAB 7 and above supports *block comments* through the use of the syntax `%{` and `%}`. These symbols mark the beginning and end, respectively, of a block of text to be treated as comments by MATLAB. For example, using block comment syntax on the previous example produces

```
%{
This is an example of multiple line comments
in an M-file. Each line requires an initial % sign or MATLAB
```


assumes the line contains code to be executed.

(Now lines can be added and edited as desired without having to place percent signs at the beginning of each line.)

```
%}
```

In addition to their utility in composing multiline comments, block comments will allow you to rapidly turn on and off the interpretation and execution of any number of lines of code in an M-file. This feature is particularly helpful in creating and debugging large M-files. Simply adding `%{` before and `%}` after a block of MATLAB code turns the enclosed code into comments that are not processed by MATLAB. When this feature is used, different sections of a script file can be executed at different times during the editing and debugging processes.

In the past, the MATLAB editor offered commands for executing a block of highlighted code in an editor window. In MATLAB 7 and above, the editor now supports the selective execution of M-file code through the use of *code cells*. A code cell is simply a block of M-file code that starts with a comment line containing two percent signs followed by a space (i.e., `% %`). The code cell continues to the end of the M-file or to the beginning of another code cell. From within the MATLAB editor, cells can be created, individually executed, and sequentially executed, thereby enabling effective M-file debugging. The **Cell** menu in the *Editor* window facilitates these operations. It is important to note that the syntax for code cells is interpreted by the editor, not by the MATLAB command interpreter. As a result, when an M-file is executed after its name has been entered in the *Command* window, code-cell syntax is ignored, and all executable lines in the file are processed.

1.4. Arrays and Array Operations:

All of the computations considered to this point have involved single numbers called scalars. Operations involving scalars are the basis of mathematics. At the same time, when we wish to perform the same operation on more than one number at a time, performing repeated scalar operations is time consuming and cumbersome. To solve this problem, MATLAB defines operations on data arrays.

1.4.1. ARRAY ADDRESSING OR INDEXING

MATLAB handles arrays in a straightforward, intuitive way. Creating arrays is easy—just follow the preceding visual organization to create the following array:

```
>> x = [0 .1*pi .2*pi .3*pi .4*pi .5*pi .6*pi .7*pi .8*pi .9*pi pi]
```

```
x =
```

```
Columns 1 through 7
```

```
0      0.3142  0.6283  0.9425  1.2566  1.5708  1.8850
```

```
Columns 8 through 11
```

```
2.1991  2.5133  2.8274  3.1416
```

```
>> y = sin(x)
```

```
y =
```

```
Columns 1 through 7
```

```
0      0.3090  0.5878  0.8090  0.9511  1.0000  0.9511
```

```
Columns 8 through 11
```

```
0.8090  0.5878  0.3090  0.0000
```


In the previous example, since x has more than one element (it has 11 values separated into columns), MATLAB gives you the result with the columns identified. As shown, x is an array having 1 row and 11 columns; or in mathematical jargon, it is a row vector, a 1-by-11 array, or simply an array of length 11.

In MATLAB, individual array elements are accessed by using subscripts; for example, $x(1)$ is the first element in x , $x(2)$ is the second element in x , and so on.

The following code is illustrative:

```
>> x(3) % The third element of x
```

```
ans =
```

```
0.6283
```

```
>> y(5) % The fifth element of y
```

```
ans =
```

```
0.9511
```

To access a block of elements at one time, MATLAB provides *colon notation*:

```
>> x(1:5)
```

```
ans =
```

```
0      0.3142  0.6283  0.9425  1.2566
```

These are the first through fifth elements in x . The notation 1:5 says, start with 1 and count up to 5. The code

```
>> x(7:end)
```

```
ans =
```

```
1.8850  2.1991  2.5133  2.8274  3.1416
```

starts with the seventh element and continues to the last element. Here, the word end signifies the last element in the array x . In the code

```
>> y(3:-1:1)
```

```
ans =
```

```
0.5878  0.3090  0
```

the results contain the third, second, and first elements in reverse order. The notation 3:-1:1 says, start with 3, count down by 1, and stop at 1. Similarly, the results in the code

```
>> x(2:2:7)
```

```
ans =
```

```
0.3142  0.9425  1.5708
```

consists of the second, fourth, and sixth elements in x . The notation 2:2:7 says, start with 2, count up by 2, and stop when you get to 7. (In this case adding 2 to 6 gives 8, which is greater than 7, and so the eighth element is not included.)

The code

```
>> y([8 2 9 1])
```

```
ans =
```

```
0.8090  0.3090  0.5878  0
```