# 11-Displaying Data

text: Chapter 4.3

ECEGR 101
Engineering Problem Solving with Matlab
Professor Henry Louie

## Overview

- disp
- fprintf with a single variable
- fprintf with multiple variables
- fprintf with an array

## Output

- To output a value
  - Type the variable and omit the semicolon
  - Use disp to output a value on the screen
  - Use fprintf to output a value on the screen or to a file



## disp Function

- Used to display the contents of a variable without displaying its name
- Can be used simply to display text

```
>> angle_rad
angle_rad =
    0.5236    0.2618
>>
>> disp(angle_rad)
    0.5236    0.2618
>>
>> disp(100.01)
   100.01
```

Only one variable can be displayed with the disp function

# Example

rad2deg.m

```
%%Example
close all
clear all
clc
angle_rad=input('Enter an angle in radians: ');
angle_deg=angle_rad*180/pi
disp('Angle in degrees: ')
disp(angle_deg)
```

```
>> rad2deg
Enter an angle in radians: pi/3
Angle in degrees:
        60
```

```
>> rad2deg
Enter an angle in radians: 0.1234
Angle in degrees:
    7.0703
```



# fprintf Function

- Used to display formatted text and numerical data on the screen or save it to a file.
- Displaying text:
  fprintf('text typed as string')
- Special characters:
  - \n start a new line (escape character)
  - \b backspace
  - \t horizontal tab

# fprintf: displaying text

```
>> fprintf('Enter a value between 1 and 10');
Enter a value between 1 and 10>>
>>
>> fprintf('Enter a value between 1 and 10\n');
Enter a value between 1 and 10
>>
```

Include \n to go to the next line after the text is printed.

```
>> fprintf('Enter a value between 1 and 10'); fprintf('Press any key to continue');
Enter a value between 1 and 10Press any key to continue>>
>>
>> fprintf('Enter a value between 1 and 10'); fprintf('\nPress any key to continue\n');
Enter a value between 1 and 10
Press any key to continue
>>
```

2nd fprintf does not automatically generate new line.



# fprintf: displaying text and data

Displaying text with numerical data:

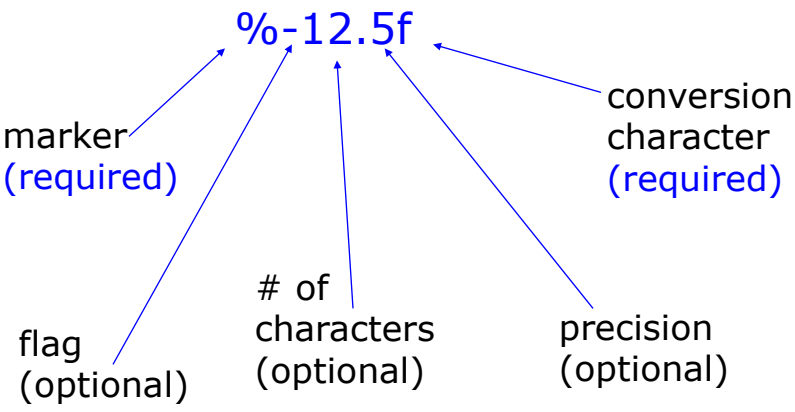
fprintf('text %-12.5f additional text', variable_name)

Formatting string: controls the appearance of the data. Specifies the alignment, significant digits, and field width.

name of the variable to be displayed

# fprintf

fprintf('text %-12.5f additional text', variable_name)






# fprintf: variable placement

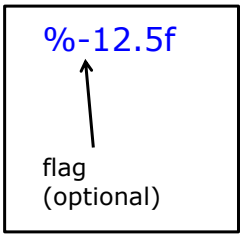
- % controls where the variable is placed
- Examples:

```
>> distance=8;
>> fprintf('I ran %2.1f miles \n',distance)
I ran 8.0 miles
```

```
>> distance=8;
>> fprintf('I ran miles %2.1f \n',distance)
I ran miles 8.0
```

# fprintf: optional flag

- Optional flag field
  - - Left justified within the field.
  - + Prints a sign character (-/+) in front of the number.
  - 0 Adds zeros if the number is shorter than the field.

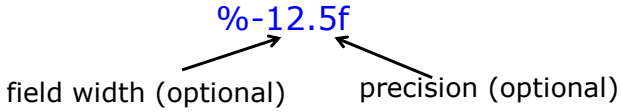



```
>> Q=25.6
Q =
   25.6000
>> fprintf('Reactive Power: %7.2f VAR \n',VAR)
Reactive Power:   25.60 VAR
>> fprintf('Reactive Power: %07.2f VAR \n',VAR)
Reactive Power: 0025.60 VAR
>> fprintf('Reactive Power: %+7.2f VAR \n',VAR)
Reactive Power:  +25.60 VAR
>> fprintf('Reactive Power: %-7.2f VAR \n',VAR)
Reactive Power: 25.60   VAR
```



# fprintf: field width and precision

- Number of characters allocated to the variable and precision can be controlled




- Examples

```
>> P=1245.3;fprintf('Power: %f Watts\n',P)
Power: 1245.300000 Watts
```

MATLAB decides field width and precision

```
>> P=1245.3;fprintf('Power: %1.0f Watts\n',P)
Power: 1245 Watts
```

```
>> P=1245.3;fprintf('Power: %10.0f Watts\n',P)
Power:       1245 Watts
```

```
>> P=1245.3;fprintf('Power: %10.2f Watts\n',P)
Power:     1245.30 Watts
```

## Fprintf: notation

- Notation can be controlled

%-12.5f

d or i - decimal
f - fixed point
e or E – exponential
s - string



## fprintf: conversion character

d or i - **decimal** (as opposed to binary, octal, hex, etc)
Example: 1.2568e+001

f - **fixed point** (no exponents)
Example: 12.5678

```
x = 12;
>> fprintf('the number is = %10.2d', x)
the number is =         12
>> fprintf('the number is = %10.2f', x)
the number is =      12.00
```

e or E – **exponential**
Example: 1.2568e+001 or 1.2568E+001

# Example

```
>> fprintf('The number %d is an example.\n', 123)
The number 123 is an example.
```

Use as many characters as required.

```
>> fprintf('The number %6d is an example.\n', 123)
The number    123 is an example.
```

Display the number in a 6-character field.

```
>> fprintf('The number %-6d is an example.\n', 123)
The number 123    is an example.
```

Display the number in a 6-character field and left justify it in the field.

```
>> fprintf('The number %06d is an example.\n', 123)
The number 000123 is an example.
```

Fill the rest of the field with zeros.



# Example

```
>> format long
>> x = pi
x =
   3.14159265358979
>>
>> fprintf('Result: %f. Press any key to continue.\n', x)
Result: 3.141593. Press any key to continue.
>>
```

Let MATLAB select the number of characters.

```
>> fprintf('Result: %8.2f. Press any key to continue.\n', x)
Result:     3.14. Press any key to continue.
```

Display the number in an 8-character field, with two places after the decimal point.

## Displaying Several Variables

Several variables can be printed out using one fprintf statement.

```
>> ave = 13.5698; med = 12;
>> fprintf('The average is %5.3f and the median is %d.\n', ave, med)
The average is 13.570 and the median is 12.
```

```
>> first_name = 'John';
>> last_name = 'Smith';
>> age = 25;
>> weight = 145.5;
>> id = 236743;
>>
>> fprintf('%s %s %d %6.1f %8d\n', first_name, last_name, age, weight, id)
John Smith 25  145.5   236743
```



## Exercise

Enter the following commands to note how the data is displayed:

```
>> x = 5;
>> fprintf('the answer is: %d',x);

>> fprintf('the answer is: %d\n',x);

>> fprintf('the answer is: %f\n',x);

>> fprintf('the answer is: %e\n',x);
```

# Exercise

```
>> format compact
>> x = 5;
>> fprintf('the answer is: %d\n',x);
the answer is: 5

>> fprintf('the answer is: %f\n',x);
the answer is: 5.000000

>> fprintf('the answer is: %e\n',x);
the answer is: 5.000000e+000
```

| | |
|---|---|
| d or i | - decimal |
| f | - fixed point |
| e or E | - exponential |
| g | - the shorter of e or f |



# Exercise

Enter the following commands to note how the data is displayed:

```
>> x = 5;

>> fprintf('the answer is: %10d\n',x);

>> fprintf('the answer is: %10.5f\n',x);
```

## fprintf with arrays

fprintf can be used to display an array (default reading order is columns first, then rows).

```
>> x = rand(2)
x =
   0.4451   0.4660
   0.9318   0.4186
>>
>> fprintf('Random number = %8.3f\n', x)
Random number =    0.445
Random number =    0.932
Random number =    0.466
Random number =    0.419
>>
```

Text is repeated for each entry



## fprintf with arrays

To display multiple columns (x, y and z) you first generate a matrix w as follows:

```
>> x = 1:5;
>> y = 10:10:50;
>> z = linspace(2,40,5);
>> w = [x; y; z]
```

```
x =
   1   2   3   4   5

y =
  10  20  30  40  50

z =
   2  11.5  21  30.5  40
```

```
w =
    1.0000    2.0000    3.0000    4.0000    5.0000
   10.0000   20.0000   30.0000   40.0000   50.0000
    2.0000   11.5000   21.0000   30.5000   40.0000
```

```
w =
    1.0000    2.0000    3.0000    4.0000    5.0000
   10.0000   20.0000   30.0000   40.0000   50.0000
    2.0000   11.5000   21.0000   30.5000   40.0000
```

x
y
z

Then use the fprintf command as follows:

```
>> fprintf('\n%5d %5d    %-5.2e', w)
```

which prints out the matrix as follows:

```
    1    10    2.00e+000
    2    20    1.15e+001
    3    30    2.10e+001
    4    40    3.05e+001
    5    50    4.00e+001
```

x y z

Matlab prints column by column, so
a column will end up as a row
or in other words:
*a row will become a column*



## Exercise

Generate a script that converts from the English unit of feet to the metric unit of meters. The display of the result to the command window should look like:

```
Enter the value of length in feet: 11.4
11.40 ft = 3.47 meters
```

where the value 11.4 was entered by the user.

## Exercise

```
% Exercise
% Convert feet to meters

% Ask the user for input
clc
feet = input('Enter the value of length in feet:');

% Convert feet to meters
meters = feet*0.3048;

% Print the result on the screen
fprintf('%4.2f ft = %4.2f meters\n\n', feet, meters);
```

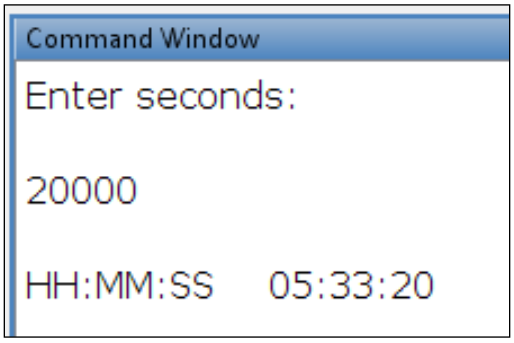


## Exercise

Write a MATLAB script that asks the user to input a time in seconds since the start of the day (this value will be somewhere between 0 and 86,400), and prints time in the form HH:MM:SS using the 24-hour clock convention.

Use the proper format converter to ensure that leading zeros are preserved in the MM and SS fields.

# Exercise






# Exercise

```
% Exercise
% Convert seconds to HH:MM:SS

% Ask the user for input
clc
secondsINPUT = input('Enter seconds:\n\n');

% Convert seconds to hours
hoursOUTPUT = floor(secondsINPUT/(60*60));
secondsINPUT = secondsINPUT - hoursOUTPUT*60*60;

% Convert the reminder to minutes
minutesOUTPUT = floor(secondsINPUT/60);
secondsOUTPUT = secondsINPUT - minutesOUTPUT*60;

% Print the result on the screen
fprintf('\nHH:MM:SS    %02d:%02d:%02d\n\n', hoursOUTPUT,
    minutesOUTPUT, secondsOUTPUT);
```