

# Today . . .

- Matlab Output
  - Disp statements
  - Fprintf statements
- Conditional statements
  - Relational operators
  - Logical operators
- Associated Reading: Palm, pages 43-48, 194 – 210, 672-674

# To quit Matlab ...

You MUST do a File => Exit Matlab operation or type exit in the command window in order to close your session and return the Matlab License to the server.

# - Review - Command Window VS Editor/Debugger

Any command that can be used in the command window can be used in a program & vice versa

- Command Window: test code and run programs
- Editor/Debugger: write programs and run functions or subprograms

(Subprograms are programs within a larger program)

# Managing Your Work Session

- ***clc*** - clears command window
- ***clear*** - removes variables from memory
- ***clf*** - clears the current figure
- ***who*** - lists variables
- ***whos*** - lists current variables and sizes

It is a good idea to begin all programs with the `clc` and `clear` commands!!! (`clf` too if a figure is output)

# Get help by...

Typing in the command window:

- ***help function name*** - lists function info
  - ***help cos***
- ***help elfun*** - lists elementary math functions

Using Help from the menu bar

# Matlab Output

- Two basic commands
  - `disp`
  - `fprintf`
- To output just text, either is ok
  - `fprintf('text you want to output')`
  - `disp('text you want to output')`
  - `fprintf('text you want to output \n')`

The `\n` is for a new line



# Matlab Output

- Two basic commands
  - `disp`
  - `fprintf`
- To output numbers, `fprintf` is better due to limited control of numeric format with `disp` command
- `fprintf('%3.2f', variable_name)`
  - `%` sign tells Matlab to interpret the text behind the `%` sign as code
  - The first number after the `%` sign (3) sets the minimum field width to output the number
  - The number after the decimal point (2) sets the number of decimal places used in the numerical output

# Combined Numeric/Text Output

- *f-print* (see *Palm App. C*, p. 672)  
*fprintf('your text to display %#.#f more text \n',  
variable\_name)*
  - %f - displays fixed-point or decimal notation
  - %s - displays string variables (any length)
  - %e - displays exponential notation
  - \n - newline
  - \t - tab
  - \f - form feed (new line same position)

## ***Example:***

- ***%3.1f = three character “field width” with one decimal place***

# Type these commands in your command window . . .

```
>> disp('This is a display example.')
```

```
>> park = 1452.834956;
```

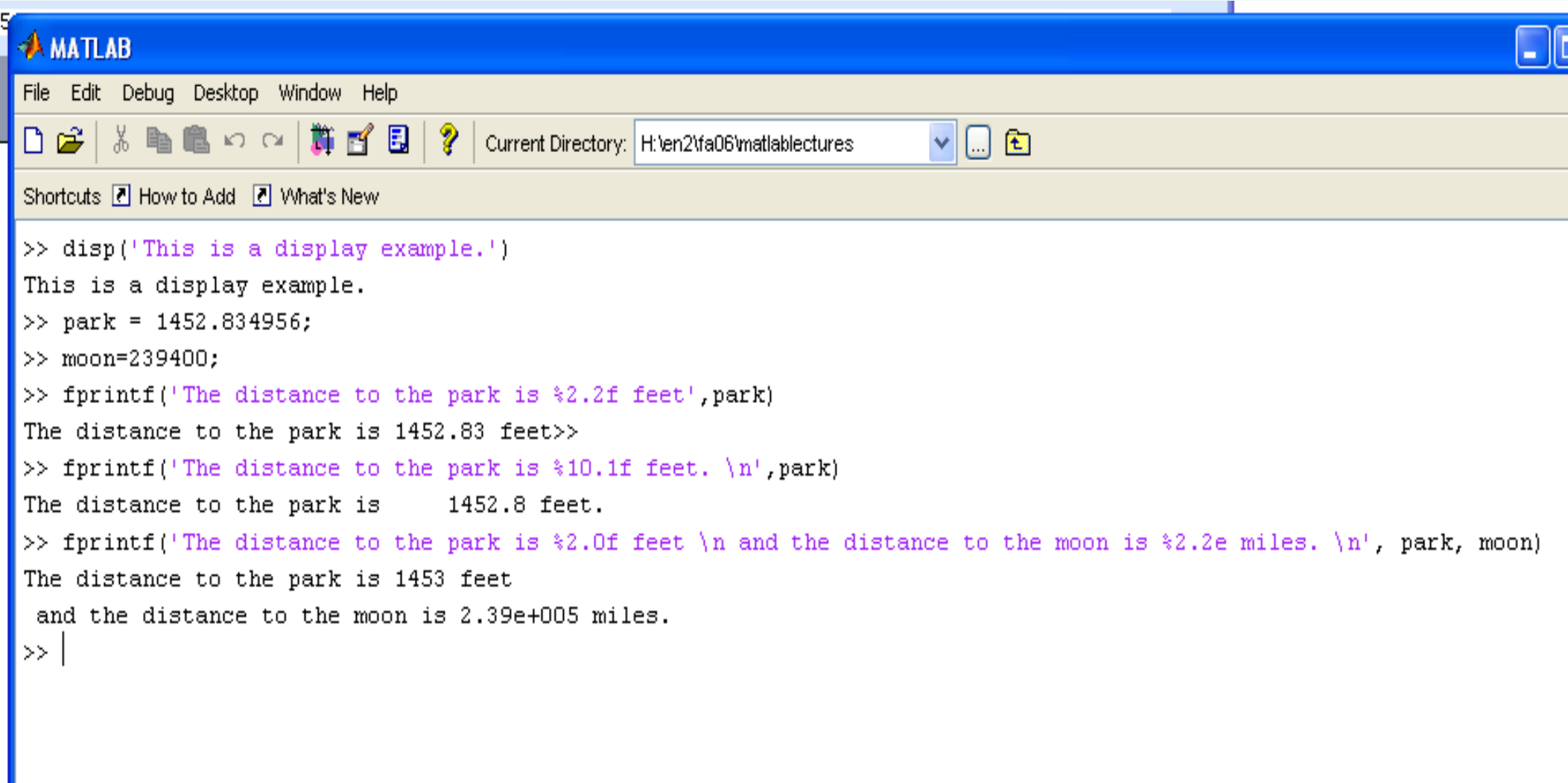
```
>> moon = 239400;
```

```
>> fprintf('The distance to the park is %2.2f feet.', park)
```

```
>> fprintf('The distance to the park is %10.1f feet. \n', park)
```

```
>> fprintf('The distance to the park is %2.0f feet \n and the  
distance to the moon is %2.2e miles. \n', park, moon)
```

# Type these commands in your command window . . .

A screenshot of the MATLAB command window interface. The window title is 'MATLAB'. The menu bar includes 'File', 'Edit', 'Debug', 'Desktop', 'Window', and 'Help'. The toolbar contains icons for file operations and a search icon. The 'Current Directory' is set to 'H:\en2\fa06\matlablectures'. Below the toolbar, there are links for 'Shortcuts', 'How to Add', and 'What's New'. The command window shows a series of commands and their outputs:

```
>> disp('This is a display example.')
This is a display example.
>> park = 1452.834956;
>> moon=239400;
>> fprintf('The distance to the park is %2.2f feet',park)
The distance to the park is 1452.83 feet>>
>> fprintf('The distance to the park is %10.1f feet. \n',park)
The distance to the park is      1452.8 feet.
>> fprintf('The distance to the park is %2.0f feet \n and the distance to the moon is %2.2e miles. \n', park, moon)
The distance to the park is 1453 feet
and the distance to the moon is 2.39e+005 miles.
>> |
```

# Combined Numeric/Text Output

- *f-print* (see *Palm App. C*, p. 672)

*fprintf('your text to display %#.#f more text  
%#.#f \n', variable1\_name, variable2\_name)*

***Write this statement and/or  
reference down!!!!***

# fprintf tips

- use `\\` to produce a backslash character
- use `%%` to produce the percent character
- when specifying number formats (ie `%2.1f` vs. `%10.1f`) use a small number for the field width when outputting numbers with text (`%2.1f`) and large numbers when outputting numbers in tabular (table) form (`%10.1f`)

# Flow Control

- Used to determine the sequence of program execution
- Flow Control in MATLAB
  - **Conditional statements**
    - if ... else ... end
  - **Loops**
    - for ... end
    - while ... end

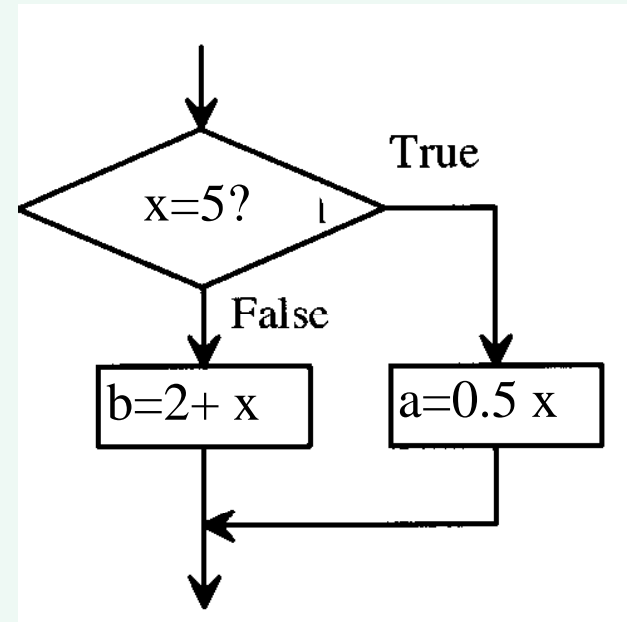
# Conditional Statement

General Format:

```
if expression  
    block of statements(1)  
else  
    block of statements (2)  
end
```

Example:

```
if  $x==5$   
     $a=0.5*x$   
else  
     $b=2+x$   
end
```



- *block statements(1)* only executed if the *expression* is **true**
- *block statements(2)* only executed if the *expression* is **false**
- Relational ( $>$ ,  $<$ ) and Logical ( $\&$ ,  $|$ ,  $\sim$ ) operations are usually combined with an if statement

# Operators

# Operators

1) arithmetic operators      ( $\wedge$ , \*, /, +, -)

2) relational operators      (<, >, ==, ~=)

3) logical operators      (~, &, |, xor)

(2 & 3 needed for program flow control)

# Relational Operators

- Comparison between arguments
- Result is either true (1) or false (0)
- Relational Operators (Palm 7 pg 44)

< less than

<= less than or equal to

> greater than

>= greater than or equal to

== equal to

~= not equal to

# Relational Operator Example

```
>> a=3+73; b=7*2;
```

```
>> a>b
```

```
ans = 1
```

```
>>c=a>b
```

```
c=1
```

# Element by Element Analysis

Can compare arrays on an element-by-element basis

```
>>x = [6, 3, 9]; y = [14, 2, 9]
```

```
>>z = (x < y)
```

```
z =
```

```
    1    0    0
```


```
 true false false
```

```
>>w = x<8
```

```
w =
```

```
    1    1    0
```

```
 true true false
```



Vectors must be  
same size!!!

# Logical Operators

(Palm, page 194)

**&    and**

**|    or**

**~    not**

**xor    exclusive or**

# Logical Operators

## & (and)

- Returns 1 (true) if both expressions are TRUE
- Returns 0 (false) when either expression is FALSE

## | (or)

- Returns 1 (true) if either expression is TRUE
- Returns 0 (false) when both expressions are FALSE

## ~ (not)

- Precedes any logical expression
- Changes the value of the expression to the opposite

## xor (**exclusive or** (*either or, but not both*))

- Returns 1 (true) if one or the other is TRUE
- Returns 0 (false) if both are TRUE or both are FALSE
- **Note: xor is used differently than the others. It works as a function e.g. xor (A,B)**

# Truth Tables

C = A & B		
1	1	1
0	0	1
0	1	0
0	0	0

C = A   B		
1	1	1
1	0	1
1	1	0
0	0	0

C = ~ (A & B)		
0	1	1
1	0	1
1	1	0
1	0	0

C = A xor B		
0	1	1
1	0	1
1	1	0
0	0	0

# Order of precedence

- First:** Parentheses; evaluated starting with innermost pair
- Second:** Arithmetic operators and logical NOT ( $\sim$ ); evaluated from left to right
- Third:** Relational operators, evaluated from left to right
- Fourth:** Logical AND
- Fifth:** Logical OR

# As a Team...

- In your Command Window, enter:

```
>> xnum=44;
```

```
>> ynum=21;
```

```
>> bb=33;
```

- Evaluate:

```
>> ynum<xnum
```

```
>> acdc=bb>xnum
```

```
>> zztop=xnum>bb
```

```
>> acdc & zztop
```

```
>> ~(zztop | acdc)
```

# As a Team...

- In your Command Window, enter:
  - >>xnum=44;
  - >>ynum=21;
  - >>bb=33;
- Evaluate: (can you predict the result?)
  - >>ynum<xnum
  - >>acdc=bb>xnum
  - >>zztop=xnum>bb
  - >>acdc & zztop
  - >>~(zztop | acdc)