



Al-Mustaqbal University

College of Engineering & Technology

Biomedical Engineering Department

Subject Name: Computer 2

2nd Class, Second Semester

Subject Code: [UOMU011046]

Academic Year: 2024-2025

Lecturer: Dr. Ahmed Hasan Al-Janabi

Email: Ahmed.janabi@uomus.edu.iq

Lecture No.: 6

Lecture Title: [Understanding for Loop]





Computer II (MATLAB)

الحاسوب 2
2025-2024

Lecture 6

by

Dr. Ahmed Hasan Al-Janabi
Ahmed.janabi@uomus.edu.iq





Learning Objectives

- Understand how to use **for** loops to repeat operations in MATLAB.
- Understand the purpose and structure of **while** loops in MATLAB.
- Common Built-in Functions





Understanding for Loops

- Definition: A for loop repeats a block of code a specified number of times.
- Usage: Ideal for iterating over arrays, performing calculations repeatedly, and automating repetitive tasks.
- Basic Structure:

```
for index = start:step:end  
    % Code to execute  
end
```





Basic Syntax of a for Loop

- Structure:

```
for i = 1:5
```

```
    disp(i); % Displays values from 1 to 5
```

```
end
```

- Explanation:

- `i = 1:5` sets the loop to run from 1 to 5, incrementing by 1 each time.
- Inside the loop, `disp(i)` displays the current value of `i`.





Using Custom Step Sizes

- Syntax: Define step sizes by specifying start:step:end.
- Example:

```
for j = 1:2:10
    disp(j); % Displays odd numbers from 1 to 9
end
```
- Explanation: The loop starts at 1, increments by 2 each time, and stops at 10.



Iterating Over Arrays

- Purpose: for loops are commonly used to access each element in an array.
- Example:

```
A = [3, 6, 9, 12];  
for k = 1:length(A)  
    disp(A(k)); % Displays each element in A  
end
```
- Explanation: The loop runs from 1 to `length(A)`, displaying each element in A sequentially.





Using Nested for Loops

- Definition: A for loop inside another for loop.
- Common Use: Useful for iterating over matrices and multidimensional arrays.

- Example:

```
for i = 1:3
    for j = 1:3
        disp([i, j]); % Displays all combinations of i and j
    end
end
```

- Explanation: The outer loop runs for each row, while the inner loop iterates through each column.





Example: Sum Array

- Problem: Write a for loop to calculate the sum of all elements in an array.
- Solution:

```
A = [1, 2, 3, 4];  
total = 0;  
for i = 1:length(A)  
    total = total + A(i);  
end  
disp(total); % Displays 10
```





Using break in a for Loop

- Purpose: break stops the loop when a condition is met.
- Example:

```
A = [3, 5, 8, 2];  
for i = 1:length(A)  
    if A(i) == 8  
        disp('Found 8');  
        break; % Exit loop once 8 is found  
    end  
end
```

- Explanation: The loop stops immediately when $A(i) == 8$.





Using continue to Skip Iterations

- Purpose: continue skips to the next iteration without executing the remaining code in the loop.

- Example:

```
for i = 1:5
    if mod(i, 2) == 0
        continue; % Skip even numbers
    end
    disp(i); % Displays only odd numbers
end
```

- Explanation: The loop displays only odd numbers, as it skips even iterations.





Example: For Loop

```
A = [1, 2, 3, 4];  
total = 0;  
for i = 1:length(A)  
    total = total + A(i);  
end  
disp(total);
```

Example: For Loop



$A = [3, 5, 2, 7];$

$total = 0;$

for $i = 1:length(A)$

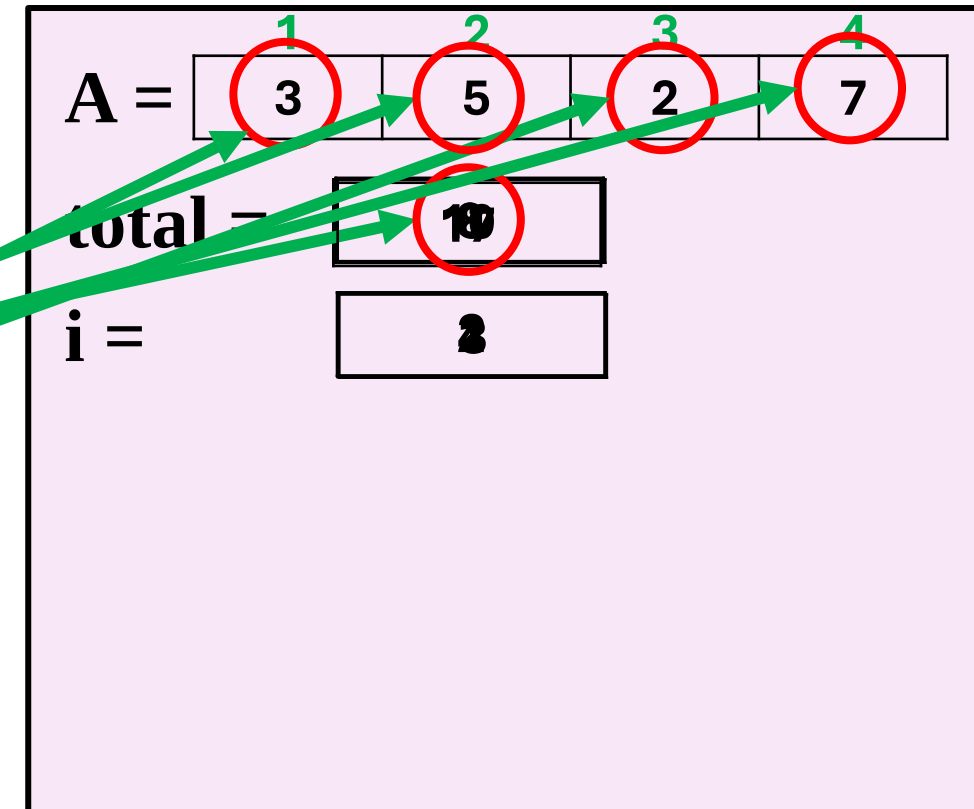
$total = total + A(i)$

end

$disp(total);$

$i = 1 \rightarrow 4$

Memory



What is the output of the following line?



Understanding while Loops

- Definition: A while loop repeats a block of code as long as a specified condition remains true.
- When to Use: Ideal when the number of iterations is not known in advance but depends on a condition.
- Basic Structure:

`while condition`

`% Code to execute repeatedly`

`end`





Basic while Loop Syntax

- Structure:

```
x = 0;  
while x < 5  
    disp(x);  
    x = x + 1;  
End
```

- Explanation:

- The loop will continue as long as $x < 5$.
- Each iteration increments x by 1 and displays its value.





Avoiding Infinite Loops

- Explanation: If the loop condition is always true, the loop will run indefinitely.
- Solution: Ensure that a variable inside the loop changes so the condition can eventually become false.
- Example of Infinite Loop:

```
x = 1;  
while x > 0  
    disp(x); % This will run indefinitely  
end
```

- Fix: Increment or modify x within the loop to avoid infinite execution.





Using while Loops with Arrays

- Example: Finding the first negative element in an array.
- Solution:

```
A = [3, 5, -2, 8, -7];
```

```
i = 1;
```

```
while i <= length(A) && A(i) >= 0
```

```
    i = i + 1;
```

```
end
```

```
if i <= length(A)
```

```
    disp(['First negative element is ', num2str(A(i))]);
```

```
else
```

```
    disp('No negative elements found');
```

```
end
```



Using Nested while Loops

- Definition: A while loop inside another while loop, useful for multi-level conditions.
- Example: Filling a 3x3 matrix with increasing numbers until a limit.

```
limit = 9;  
matrix = zeros(3);  
i = 1;  
j = 1;  
count = 1;  
while count <= limit
```

```
    while j <= 3  
        matrix(i, j) = count;  
        count = count + 1;  
        j = j + 1;  
    end  
    j = 1; % Reset column  
    i = i + 1; % Move to next row  
end  
disp(matrix);
```

```
1,1  1,2  1,3  
2,1  2,2  2,3  
3,1  3,2  3,3
```



Using break in a while Loop

- Purpose: break stops the loop immediately when a condition is met.

- Example:

```
A = [3, 5, 7, -2, 4];
```

```
i = 1;
```

```
while i <= length(A)
```

```
    if A(i) < 0
```

```
        disp(['Negative number found: ', num2str(A(i))]);
```

```
        break; % Exit loop when a negative number is found
```

```
    end
```

```
    i = i + 1;
```

```
end
```





Built-in Functions for Arrays

- MATLAB provides several built-in functions for performing operations on arrays.
- Benefits: Simplifies code and improves readability.
- Examples: **sum, max, min, mean, ...**

```
array = [1, 2, 3, 4];  
sum_array = sum(array);
```





Basic Matrix Function - sum

- Computes the sum of elements along a specified dimension.
- **Syntax:** `sum(A, dim)`
 - `dim = 1`: Sum along columns.
 - `dim = 2`: Sum along rows.
- **Examples:**

```
A = [1, 2, 3; 4, 5, 6];  
col_sum = sum(A, 1);  
row_sum = sum(A, 2);
```





Basic Matrix Function - max and min

- **max**: Returns the largest element in an array or matrix.
- **min**: Returns the smallest element.
- **Syntax**: `max(A, [], dim)` and `min(A, [], dim)`
- **Examples**:

```
A = [1, 3, 5; 2, 4, 6];  
max_val = max(A);  
min_val = min(A);
```





Basic Matrix Function - mean and median

- **mean**: Calculates the average.
- **median**: Finds the middle value.
- **Syntax**: `mean(A, dim)` and `median(A, dim)`
- **Examples**:

```
A = [1, 3, 5; 2, 4, 6];  
mean_val = mean(A);  
median_val = median(A);
```





Basic Matrix Function - length and size

- **length**: Finds the longest dimension of an array.
- **size**: Returns the dimensions of a matrix.
- **Examples:**

```
A = [1, 3, 5; 2, 4, 6];  
len = length(A);  
[rows, cols] = size(A);
```





Review of Key Concepts

- **Loop Structure:** Use for to repeat a block of code.
- **Step Sizes:** Customize increments with start:step:end.
- **Loop Structure:** Use while to repeat code while a condition is true.
- **Common Built-in Functions**





Let's try MATLAB

Launch MATLAB and work towards the exercises

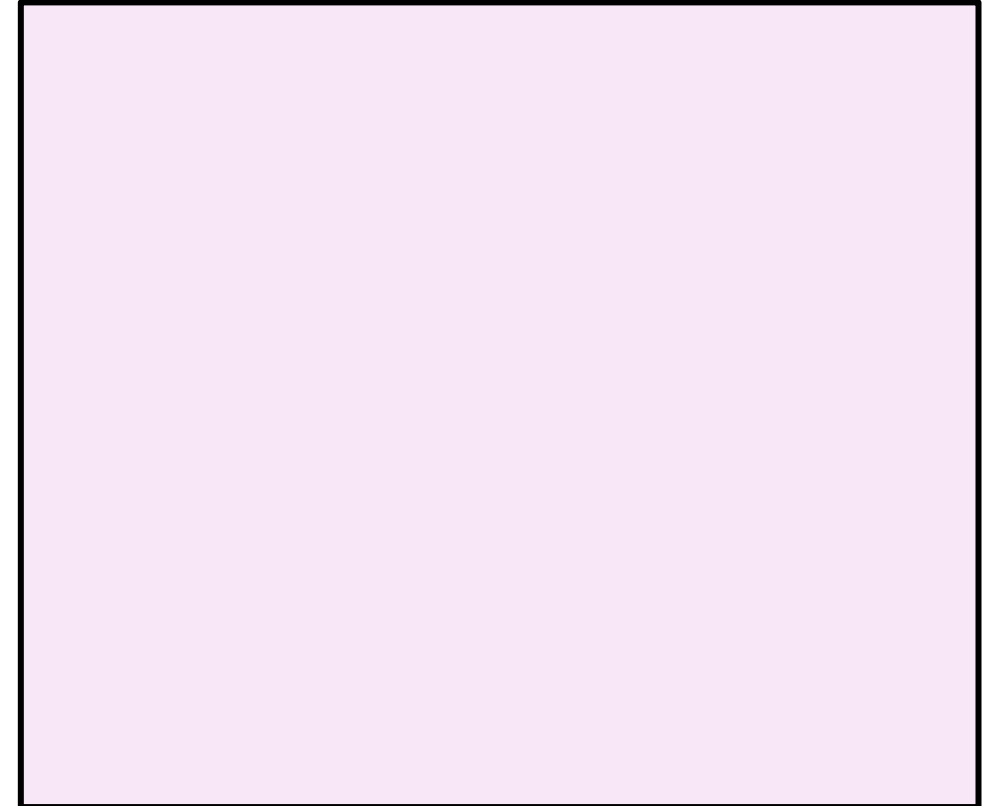




Example: For Loop

```
A = [3, 5, 2, 7];  
total = 0;  
for i = 1:length(A)  
    total = total + A(i);  
end  
disp(total);
```

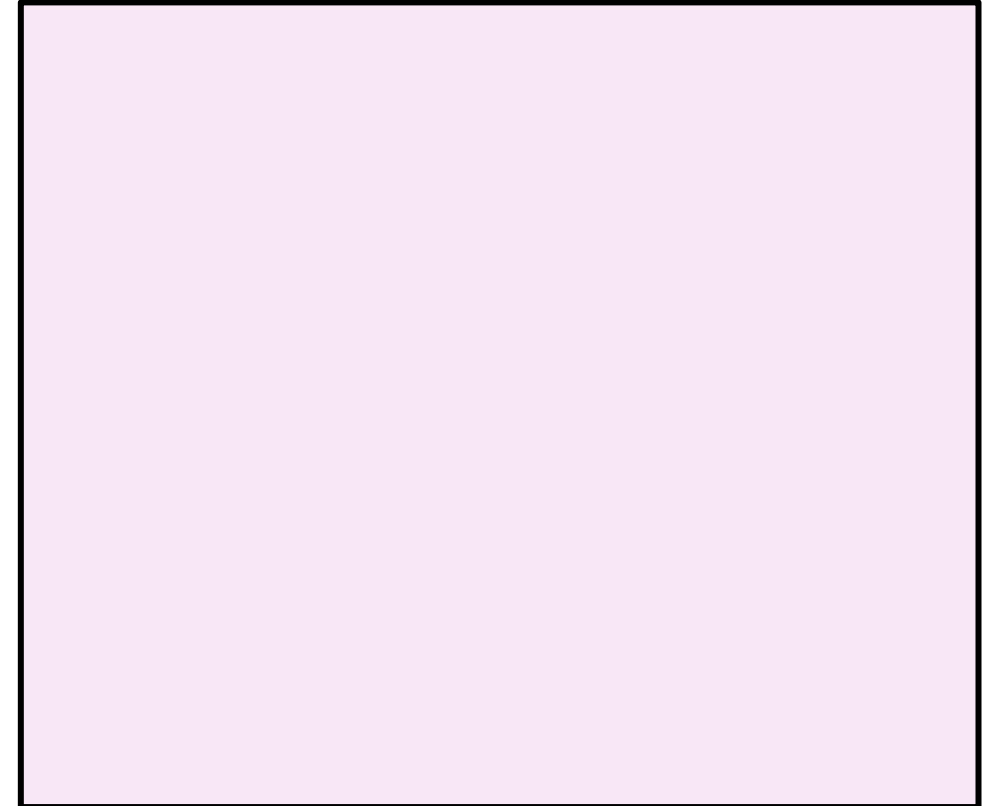
Memory





Example: For Loop

Memory



→ `A = [3, 5, 2, 7];`
`total = 0;`
`for i = 1:length(A)`
 `total = total + A(i);`
`end`
`disp(total);`

Example: For Loop

`A = [3, 5, 2, 7];` ✓

→ `total = 0;`

`for i = 1:length(A)`

`total = total + A(i);`

`end`

`disp(total);`

Memory

A =	3	5	2	7
------------	---	---	---	---

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

→ $for\ i = 1:length(A)$

$total = total + A(i);$

end

$disp(total);$

Memory

$A =$

3	5	2	7
---	---	---	---

$total =$

0

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

→ $for\ i = 1:length(A)$

$total = total + A(i);$

end

$disp(total);$

Memory

$A =$	3	5	2	7
$total =$	0			

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

$for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$\rightarrow total = total + A(i);$

end

$disp(total);$

Memory

	1	2	3	4
$A =$	3	5	2	7

$total =$ 0

$i =$ 1

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

$for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$\rightarrow total = total + A(i); \rightarrow 0 + 3$

end

$disp(total);$

Memory

	1	2	3	4
$A =$	3	5	2	7

$total =$ 0

$i =$ 1

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

$for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$total = total + A(i);$ ✓

→ end

$disp(total);$

Memory

	1	2	3	4
A =	3	5	2	7
total =	3			
i =	1			

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

→ $for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$total = total + A(i);$ ✓

end ✓

$disp(total);$

Memory

	1	2	3	4
A =	3	5	2	7

total = 3

i = 2

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

$for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$\rightarrow total = total + A(i); \rightarrow 3 + 5$

end

$disp(total);$

Memory

	1	2	3	4
A =	3	5	2	7

total = 3

i = 2

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

$for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$total = total + A(i);$ ✓

→ end

$disp(total);$

Memory

	1	2	3	4
$A =$	3	5	2	7

$total =$ 8

$i =$ 2

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

→ $for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$total = total + A(i);$

end

$disp(total);$

Memory

	1	2	3	4
A =	3	5	2	7
total =	8			
i =	3			

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

$for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$\rightarrow total = total + A(i); \rightarrow 8 + 2$

end

$disp(total);$

Memory

	1	2	3	4
$A =$	3	5	2	7

$total =$ 8

$i =$ 3

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

$for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$total = total + A(i);$ ✓

→ end

$disp(total);$

Memory

	1	2	3	4
A =	3	5	2	7

total = 10

i = 3

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

→ $for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$total = total + A(i);$

end

$disp(total);$

Memory

	1	2	3	4
$A =$	3	5	2	7

$total =$ 10

$i =$ 4

Example: For Loop

`A = [3, 5, 2, 7];` ✓

`total = 0;` ✓

`for i = 1:length(A) → i = 1 → 4`

→ `total = total + A(i); → 10 + 7`

`end`

`disp(total);`

Memory

	1	2	3	4
A =	3	5	2	7

total = 10

i = 4

Example: For Loop

$A = [3, 5, 2, 7];$ ✓

$total = 0;$ ✓

$for\ i = 1:length(A) \rightarrow i = 1 \rightarrow 4$

$total = total + A(i);$ ✓

$\rightarrow end$ ✓

$disp(total);$

Memory

	1	2	3	4
$A =$	3	5	2	7

$total =$ 17

$i =$ 4

Example: For Loop

`A = [3, 5, 2, 7];` ✓

`total = 0;` ✓

`for i = 1:length(A)` → `i = 1` → 4

`total = total + A(i);` ✓

`end` ✓

`disp(total);`

What is the output
of the following
line?

Memory

	1	2	3	4
A =	3	5	2	7

total = 17

i = 4



Quiz Group A

1. Which statement correctly finds all elements in an array A that are divisible by 11?

- A) `mod(A, 1) == 11`
- B) `A(A == 11)`
- C) `(mod(A, 11) == 0)`
- D) `mod(A, 11) == 11`

2. what does the following code do?

- A) 15
 - B) 10
 - C) 5
 - D) 0
- ```
sum = 0;
for i = 1:5
 sum = i+i;
end
disp(sum);
```



1. What will be displayed by the following code?

```
for i = 1:3:5
 if i == 3
 break;
 end
 disp(i);
end
```

2. What will the following code display?

```
for i = 10:-3:4
 disp(i);
end
```