Unit 4:
Control flow (IV)

Nesting loops

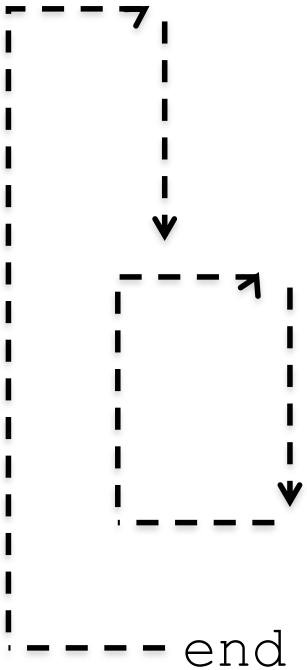
- It is possible to nest loops

```
for variable1=expression1
     statement1;
    statement2;
    …
    for variable2=expression2
       statementN;
       statementM;
           …
     end
    …
end
```

Nesting loops

- It is possible to nest loops

```
for variable1=expression1
    statement1;
    statement2;
    …
    for variable2=expression2
        statementN;
        statementM;
            …
    end
    …
end
```

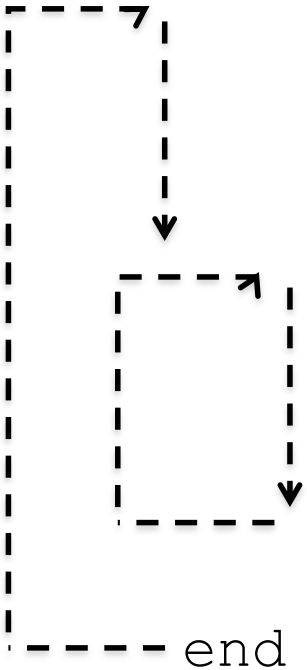


For each iteration of the outmost loop, MATLAB will repeat all the iterations of the innermost loop

Nesting loops

- It is possible to nest loops

```
for variable1=expression1
    statement1;
    statement2;
    …
    for variable2=expression2
        statementN;
        statementM;
            …
    end
    …
end
```



For each iteration of the outmost loop, MATLAB will repeat all the iterations of the innermost loop

If expression1 is a vector of size N and expression2 is a vector of size M:

- The statement1 will be executed N times
- The statementN will be executed N*M times

Nesting loops

- It is possible to nest loops

```
while expression1
     statement1;
     statement2;
     …
     while expression2
          statementN;
          statementM;
               …
   end
     …
end
```

You can nest any number/combination of while/for loops

Nesting loops

- It is possible to nest loops

```
for variable1=expression1
     statement1;
    statement2;
    …
    while expression2
        statementN;
        statementM;
            …
  end
    …
end
```

You can nest any number/combination of while/for loops

Nesting loops

- It is possible to nest loops

```
while expression1
     statement1;
    statement2;
    …
    for variable2=expression2
        statementN;
        statementM;
            …
     end
    …
end
```

You can nest any number/combination of while/for loops

Example

Compute and print the multiplication table of the numbers from 1 to 10

```
1 multiply by 1 is 1
1 multiply by 2 is 2
1 multiply by 3 is 3
1 multiply by 4 is 4
1 multiply by 5 is 5
1 multiply by 6 is 6
1 multiply by 7 is 7
. . .
1 multiply by 10 is 10
2 multiply by 1 is 2
2 multiply by 2 is 4
2 multiply by 3 is 6
...
```

Example

Compute and print the multiplication table of the numbers from 1 to 10

Step 1.- Write a for loop to print the multiplication table of one number (i.e. the number 2)

```
for j=1:10
        num = 2 * j;
        fprintf('\n %d multiply by %d is %d', 2, j, num);
end
```

Example

Compute and print the multiplication table of the numbers from 1 to 10

Steps 2.- We need to print 10 multiplication tables… so place the *for* loop within another *for* loop that goes from 1 to 10 (ten iterations)

This will print the multiplication table of the number 2 ten times

```
for i=1:10
  for j=1:10
        num = 2 * j;
        fprintf('\n %d multiply by %d is %d', 2, j, num);
  end
end
```

Example

Compute and print the multiplication table of the numbers from 1 to 10

Steps 3.- The first time it should print the multiplication table of the number 1, the next time of the number 2, the next time of the number 3…. and so on. Therefore, the 2 should be replaced by the values 1 2 3 4 5 6 7 8 9 10 consecutively… we have a variable which takes this values: **i**

```
for i=1:10
  for j=1:10
        num = i * j;
        fprintf('\n %d multiply by %d is %d', 2, j, num);
  end
end
```

Example

What would be the output of this program?

```
for i=1:10
    for j=1:10
     num = i * j;
    end
    fprintf('\n %d multiply by %d is %d', i, j, num);
end
```

Exercise

Write a program which asks the user to introduce numbers and prints their multiplication table. The program stops when the user introduces a negative value

```
Introduce a number :3
 3 multiply by 1 is 3
 3 multiply by 2 is 6
...
 3 multiply by 10 is 30
Introduce a number :7
 7 multiply by 1 is 7
 7 multiply by 2 is 14
 . . .
 7 multiply by 10 is 70
Introduce a number :-1
```

Exercise

Exercise: Compute and print the factorial of the numbers from 1 to 10

```
The factorial of 1 is 1
The factorial of 2 is 2
The factorial of 3 is 6
The factorial of 4 is 24
The factorial of 5 is 120
...
```

Steps:
1.- write the program to print the factorial
2.- modify the program so that it prints the factorial of all the numbers from 1 to 10

Exercise

- Excercise: Write a program which prints the following output on the screen.

```
11111
22222
33333
44444
55555
```

Exercise

- Excercise: Write a program which prints the following output on the screen.

```
1
22
333
4444
55555
```

Exercise

- Excercise: Write a program which prints the following output on the screen.

```
1
12
123
1234
12345
```

Exercise

- Excercise: Write a program which prints the following output on the screen.

```
12345
1234
123
12
1
```

Break and Continue

THE USE OF 'BREAK' DURING THIS COURSE IS STRICTLY FORBIDDEN!!!

- break
  - Terminates the execution of a ´for loop´ or ´while loop´.
  - When a break statement is encountered, execution **continues with the next statement outside of the loop**.
  - In nested loops, break exits from the innermost loop only.
- continue
  - Passes the control **to next** iteration of the for or while loop, skipping any remaining statements

EXERCISES TO PRACTICE LOOPS

Exercise

Exercise: Write a program which asks the user to introduce numbers and stores them in a vector one after the other. After introducing a value the program asks the user if he/she wants to introduce more (Y/N), and when the user finishes the program prints the content of the vector.

```
Introduce a number: 3
Do you want to introduce more values (Y/N)? Y
Introduce a number: 7
Do you want to introduce more values (Y/N)? Y
Introduce a number: 2
Do you want to introduce more values (Y/N)? N
The numbers in the vector are:
 3  7  2
```

Exercise

```
vector = [];
index  = 0;
cContinue = 'Y';
while (cContinue == 'Y')
    number = input('Introduce a number: ');
    index = index + 1;
    vector(index) = number;
    cContinue = input('Do you want to introduce more values (Y/N)? ','s');
end;

disp ('The numbers in the vector are:');
for value = vector
    fprintf(' %d ', value);
end;
```

Exercise

What will happen if the user answers the question with something different from Y or N?

```
vector = [];
index  = 0;
cContinue = 'Y';
while (cContinue == 'Y')
    number = input('Introduce a number: ');
    index = index + 1;
    vector(index) = number;
    cContinue = input('Do you want to introduce more values (Y/N)? ','s');
end;

disp ('The numbers in the vector are:');
for value = vector
    fprintf(' %d ', value);
end;
```

Exercise

What will happen if the user answers the question with something different from Y or N?
The program will stop asking for numbers.

```
vector = [];
index  = 0;
cContinue = 'Y';
while (cContinue == 'Y')
    number = input('Introduce a number: ');
    index = index + 1;
    vector(index) = number;
    cContinue = input('Do you want to introduce more values (Y/N)? ','s');
end;

disp ('The numbers in the vector are:');
for value = vector
    fprintf(' %d ', value);
end;
```

Exercise

Exercise: Modify the previous program so that the if the user answers with something different from Y or N the program repeats the question.

```
Introduce a number: 3
Do you want to introduce more values (Y/N)? X
Sorry, I don't understand.
Do you want to introduce more values (Y/N)? T
Sorry, I don't understand.
Do you want to introduce more values (Y/N)? Y
Introduce a number: 7
Do you want to introduce more values (Y/N)? N
The numbers in the vector are:
 3  7
```

Exercise

Exercise: Let's consider the variable *vect* contains a vector of integer numbers the user previously introduced. Write a program which asks the user to introduce a number and it says if the number is in the vector or not.

Exercise

Exercise: Write a program that given two vectors of numbers vect1 and vect2 prints on screen the numbers of vect1 that appear in vect2

Let's assume vect1 has been filled previously with the numbers to search and vect2 with the numbers to compare

Summary

- Make sure you understand:
  - When does a loop ends?
  - What is the value of each variable before, during and after the loop for the following cases:
    - Simple ‘for’ loops
    - Nested ‘for’ loops
    - Simple ‘while’ loops
    - ‘While’ loops with compound conditions