Unit 4: Control flow (II)

uc3m | Universidad Carlos III de Madrid

DEI Interactive Systems Group

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
fprintf('\n Number: %d',1);
fprintf('\n Number: %d',2);
fprintf('\n Number: %d',3);
fprintf('\n Number: %d',4);
fprintf('\n Number: %d',5);
fprintf('\n Number: %d',6);
fprintf('\n Number: %d',7);
fprintf('\n Number: %d',8);
fprintf('\n Number: %d',9);
fprintf('\n Number: %d',10);
```

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
fprintf('\n Number: %d',1);
fprintf('\n Number: %d',2);
fprintf('\n Number: %d',3);
fprintf('\n Number: %d',4);
fprintf('\n Number: %d',5);
fprintf('\n Number: %d',6);
fprintf('\n Number: %d',7);
fprintf('\n Number: %d',8);
fprintf('\n Number: %d',9);
fprintf('\n Number: %d',10);
```

We are repeating the same command 10 times…
In this case it's not too bad, but what if we have to print the numbers between 1 and 1000?
We don't want to write the same instruction 1000 times!!

Iterative Statements

- Allow us to repeat the execution of the same block of statements.
- There are 2 types of loops in MATLAB:
  - **for-loops**: repeat a sequence of instructions a **fixed number of times** by stepping an index through a set of values.
  - **while-loops**: repeat a sequence of instructions **until a boolean test is true.**
- The structure in both cases is similar:
  - Variables initialization.
  - A expression to test the end condition of the loop.
  - The statements to be iterated, including at least one variable referred to in the test condition.

For loop

-
  ```
  for variable = expression
           statement1;
            statement2;
            …
  end
  ```

  The for-loop in MATLAB works like this

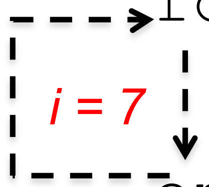
  - MATLAB evaluates the <*expression*> **obtaining a vector of values as a result**.
  - The <*variable*> is set to the first value in the vector and the sequence of statements is executed with this value of the <*variable*>
  - Then the <*variable*> is set to the second value in the vector and the sequence of statements is executed with this value of the <*variable*>
  - This process is repeated until <*variable*> has been set to all the values of the vector. Then the loop ends.

Example

```
for i=[7 1 3 4]
    fprintf('\n Number: %d',i);
end
```

Example

1st iteration



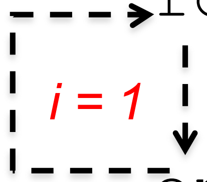
```
for i=[7 1 3 4]
   fprintf(‘\n Number: %d‘,i);
end
```

Program output

```
Number: 7
```

Example

2nd iteration



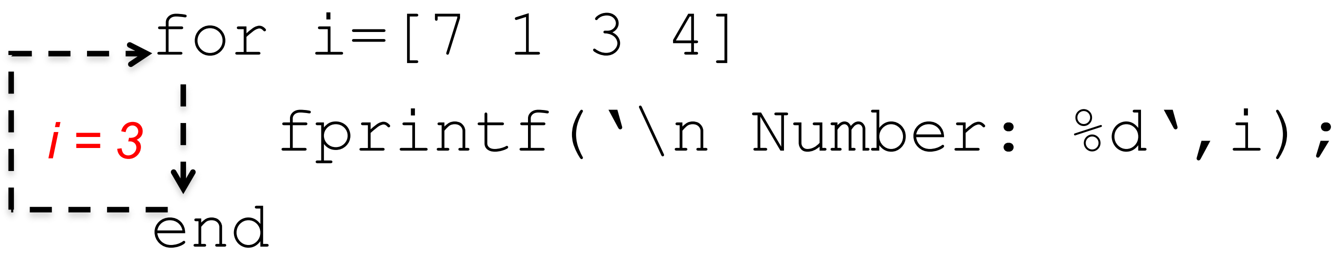
```
for i=[7 1 3 4]
   fprintf('\n Number: %d',i);
end
```

Program output

```
Number: 7
Number: 1
```

Example

3rd iteration



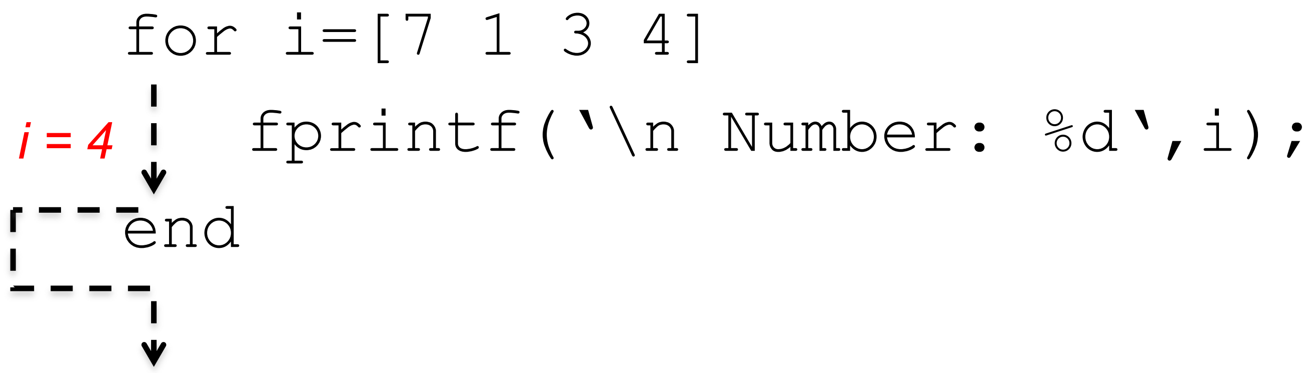
```
for i=[7 1 3 4]
    fprintf('\n Number: %d',i);
end
```

Program output

```
Number: 7
Number: 1
Number: 3
```

Example

4th iteration



```
for i=[7 1 3 4]
    fprintf('\n Number: %d',i);
end
```

Program output

```
Number: 7
Number: 1
Number: 3
Number: 4
```

Example

You can also use vectors of characters

```
for i=['C' 'a' 't']
    fprintf('\n%c',i);
end
```

Example

1st iteration

i = ‘C’

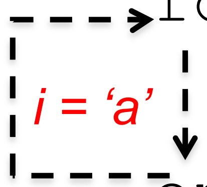
```
for i=['C' 'a' 't']
    fprintf('\n%c',i);
end
```

Program output

```
C
```

Example

2nd iteration



```
for i=[‘C’ ‘a’ ‘t’]
    fprintf(‘\n%c‘,i);
end
```

Program output

```
C
a
```

Example

3rd iteration

i = ‘t’

```
for i=['C' 'a' 't']
   fprintf('\n%c',i);
end
```

Program output

```
C
a
t
```

Example

Most of the times you'll write the for loops like this:

```
for i= init_number:last_number
    . . .
end
```

MATLAB will replace this by a vector containing the numbers between init_number and last_number. Then the variable *i* will be assigned to those values one by one at each iteration

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
   fprintf('\n Number: %d',i);
end
```

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
    fprintf('\n Number: %d',i);
end
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
    fprintf('\n Number: %d',i);
end
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Output

```
Number: 1
```

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
   fprintf('\n Number: %d',i);
end
```

Output

```
Number: 1
Number: 2
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
    fprintf('\n Number: %d',i);
end
```

Output

```
Number: 1
Number: 2
Number: 3
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
    fprintf('\n Number: %d',i);
end
```

Output

```
Number: 1
Number: 2
Number: 3
Number: 4
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
    fprintf('\n Number: %d',i);
end
```

Output

```
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
    fprintf('\n Number: %d',i);
end
```

Output

```
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
Number: 6
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
   fprintf('\n Number: %d',i);
end
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Output

```
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
Number: 6
Number: 7
```

uc3m | Universidad **Carlos III** de Madrid

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
    fprintf('\n Number: %d',i);
end
```

Output

```
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
Number: 6
Number: 7
Number: 8
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
    fprintf('\n Number: %d',i);
end
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Output

```
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
Number: 6
Number: 7
Number: 8
Number: 9
```

Example

Example: Display on the screen the numbers between 1 and 10 sorted in increasing order.

```
for i=1:10
   fprintf('\n Number: %d',i);
end
```

The statement `fprintf('\n Number: %d ',i)` will be executed ten times. Each time the variable `i` will have a different value ranging from 1 to 10.

Output

```
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
Number: 6
Number: 7
Number: 8
Number: 9
Number: 10
```

Example

It is also possible to write loops like this:

```
for i= init_number:step:last_number
    . . .
end
```

$\downarrow$

MATLAB will replace this expression by a vector which contains numbers between init_number and last_number obtained adding/substracting the step to the previous number in the vector

Example

It is also possible to write loops like this:

```
for i= init_number:step:last_number
    . . .
end
```

For example:

```
    for i= 1:2:10
              . . .
    end
```

Example

It is also possible to write loops like this:

```
for i= init_number:step:last_number
    . . .
end
```

This will be replaced by [1 3 5 7 9]

For example:

```
    for i= 1:2:10
           . . .
    end
```

Example

It is also possible to write loops like this:

```
for i= init_number:step:last_number
    . . .
end
```

For example:

```
    for i= 0:5:20
              . . .
    end
```

Example

It is also possible to write loops like this:

```
for i= init_number:step:last_number
    . . .
end
```

This will be replaced by [0 5 10 15 20]

For example:

```
    for i= 0:5:20
               . . .
    end
```

Example

It is also possible to write loops like this:

```
for i= init_number:step:last_number
    . . .
end
```

For example:

```
    for i= 100:-25:10
            . . .
    end
```

Example

It is also possible to write loops like this:

```
for i= init_number:step:last_number
   . . .
end
```

This will be replaced by [100 75 50 25]

For example:

```
    for i= 100:-25:10
           . . .
    end
```

Example

Example: Display on the screen the numbers between 1 and 10 sorted in decreasing order.

Example

Example: Display on the screen the numbers between 1 and 10 sorted in decreasing order.

```
for i=10:-1:1
    fprintf('\n Number: %d',i);
end
```

This is the same as `[10 9 8 7 6 5 4 3 2 1]`

Output

```
Number: 10
Number: 9
Number: 8
Number: 7
Number: 6
Number: 5
Number: 4
Number: 3
Number: 2
Number: 1
```

Exercise

Exercise: Display on the screen the numbers between 1 and 10 (included) squared.

Exercise

Exercise: Display on the screen the numbers between 1 and 10 (included) squared.

```
for i=1:10
      square = i * i;
      fprintf(‘\n The square of %d is %d ‘, i, square);
end
```

Modifying variables inside a loop

What will be the value of the variable ‘var’ at the end of the program execution?

```
var = 0;
for i=1:5
     var = var + 1;
end
fprintf(‘\n The value is %d ‘, var);
```

Modifying variables inside a loop

What will be the value of the variable ‘var’ at the end of the program execution?

```
var = 0;
for i=1:5
      var = var + 1;
end
fprintf(‘\n The value is %d ‘, var);
```

1st iteration

i = 1

var = 0 + 1 = 1

Modifying variables inside a loop

What will be the value of the variable ‘var’ at the end of the program execution?

```
var = 0;
for i=1:5
      var = var + 1;
end
fprintf(‘\n The value is %d ‘, var);
```

1st iteration

  i = 1

  var = 0 + 1 = 1

2ndt iteration

  i = 2

  var = 1 + 1 = 2

Modifying variables inside a loop

What will be the value of the variable ‘var’ at the end of the program execution?

```
var = 0;
for i=1:5
      var = var + 1;
end
fprintf(‘\n The value is %d ‘, var);
```

1st iteration
    i = 1
    var = 0 + 1 = 1
2nd iteration
    i = 2
    var = 1 + 1 = 2
3nd iteration
    i = 3
    var = 2 + 1 = 3

Modifying variables inside a loop

What will be the value of the variable ‘var’ at the end of the program execution?

```
var = 0;
for i=1:5
      var = var + 1;
end
fprintf(‘\n The value is %d ‘, var);
```

1st iteration
    i = 1
    var = 0 + 1 = 1
2nd iteration
    i = 2
    var = 1 + 1 = 2
3nd iteration
    i = 3
    var = 2 + 1 = 3
4rd iteration
    i = 4
    var = 5 + 1 = 4

Modifying variables inside a loop

What will be the value of the variable ‘var’ at the end of the program execution?

```
var = 0;
for i=1:5
     var = var + 1;
end
fprintf(‘\n The value is %d ‘, var);
```

1st iteration
    i = 1
    var = 0 + 1 = 1
2nd iteration
    i = 2
    var = 1 + 1 = 2
3nd iteration
    i = 3
    var = 2 + 1 = 3
4rd iteration
    i = 4
    var = 3 + 1 = 4
5th iteration
    i = 5
    var = 4 + 1 = 5

Modifying variables inside a loop

What will be the value of the variable ‘var’ at the end of the program execution?

```
var = 0;
for i=1:5
      var = var + 1;
end
fprintf(‘\n The value is %d ‘, var);
```

At the end of the program the value of *var* is 5

Modifying variables inside a loop

And in this case?

```
var = 0;
for i=1:5
      var = var + i;
end
fprintf(‘\n The value is %d ‘, var);
```

Modifying variables inside a loop

And in this case?

1st iteration

i = 1

var = 0 + 1 = 1

```
var = 0;
for i=1:5
      var = var + i;
end
fprintf(‘\n The value is %d ‘, var);
```

Modifying variables inside a loop

And in this case?

```
var = 0;
for i=1:5
      var = var + i;
end
fprintf('\n The value is %d ', var);
```

1st iteration
  i = 1
  var = 0 + 1 = 1
2nd iteration
  i = 2
  var = 1 + 2 = 3

Modifying variables inside a loop

And in this case?

```
var = 0;
for i=1:5
     var = var + i;
end
fprintf(‘\n The value is %d ‘, var);
```

1st iteration
- i = 1
- var = 0 + 1 = 1

2nd iteration
- i = 2
- var = 1 + 2 = 3

3nd iteration
- i = 3
- var = 3 + 3 = 6

Modifying variables inside a loop

And in this case?

```
var = 0;
for i=1:5
      var = var + i;
end
fprintf('\n The value is %d ', var);
```

1st iteration

    i = 1
    var = 0 + 1 = 1

2nd iteration

    i = 2
    var = 1 + 2 = 3

3nd iteration

    i = 3
    var = 3 + 3 = 6

4rd iteration

    i = 4
    var = 6 + 4 = 10

Modifying variables inside a loop

And in this case?

```
var = 0;
for i=1:5
     var = var + i;
end
fprintf(‘\n The value is %d ‘, var);
```

1st iteration

    i = 1
    var = 0 + 1 = 1

2nd iteration

    i = 2
    var = 1 + 2 = 3

3nd iteration

    i = 3
    var = 3 + 3 = 6

4rd iteration

    i = 4
    var = 6 + 4 = 10

5th iteration

    i = 5
    var = 10 + 5 = 15

Modifying variables inside a loop

And in this case?

```
var = 0;
for i=1:5
      var = var + i;
end
fprintf(‘\n The value is %d ‘, var);
```

At the end of the program the value of *var* is 15

Exercise

Write a program that asks the user to introduce two numbers and prints on screen all the numbers between them

Exercise

Write a program that asks the user to introduce two numbers and prints on screen all the numbers between them

```
varMin = input(‘Introduce a number’);
varMax = input(‘Introduce another number’);
for i=varMin:varMax
     fprintf(‘\n Number: %d‘,i);
end
```

Working with vectors

- *For* loops are specially useful when working with vectors. We can use them to:
  - Modify the values of the elements of the vector
  - Retrieve their values
  - Count how many of them satisfy certain condition
  - Find the maximun, minimun…
  - … in general to loop through the elements in the vector and work with them

Working with vectors

Exercise: Write a program that asks the user to introduce 20 numbers and stores them in a vector

Working with vectors

Exercise: Write a program that asks the user to introduce 20 numbers and stores them in a vector

```
vect = zeros(1,20);
for i=1:20
     vect(i) = input ('Introduce a number: ');
end
```

Working with vectors

Exercise: Write a program that asks the user to introduce 20 numbers and stores them in a vector

```
vect = zeros(1,20);
for i=1:20
     vect(i) = input ('Introduce
end
```

It is a good practice to initialize the vector before using it. We do it because the variable vect could have been used in a previous program and it might contain data from previous executions. When working with numbers we normally initialize filling it with zeros, but you can put anything else

Working with vectors

Exercise: Modify the previous program so that, once the user finishes introducing the values, it counts how many numbers in the vector are even

Working with vectors

Exercise: Modify the previous program so that, once the user finishes introducing the values, it counts how many numbers in the vector are even

```
vect = zeros(1,20);
for i=1:20
     vect(i) = input ('Introduce a number: ');
end
count = 0;
for i=1:20
    if (rem(vect(i),2) ==0)
          count = count + 1;
    end
end
fprintf('\There are %d even numbers', count)
```

Working with vectors

Exercise: Modify the previous program so that, once the user finishes introducing the values, it counts how many numbers in the vector are even

```
vect = zeros(1,20);
for i=1:20
     vect(i) = input ('Introduce a number: ');
end
count = 0;
for i=1:20
    if (rem(vect(i),2) ==0)
          count = count + 1;
    end
end
fprintf('\There are %d even numbers', count)
```

You can put ifs, switchs… any commands inside the for… even another for.

Working with vectors

Exercise: Modify the previous program so that, once the user finishes introducing the values, it says how many numbers in the vector are even

```
vect = zeros(1,20);
for i=1:20
     vect(i) = input ('Introduce a number: ');
end
count = 0;
for number=vect
    if (rem(number,2) ==0)
          count = count + 1;
    end
end
fprintf('\There are %d even numbers', count)
```

ANOTHER SOLUTION. IN THIS CASE WE USE vect DIRECTLY AS THE VECTOR TO ITERATE IN THE for

Working with vectors

Exercise: Modify the previous program so that, once the user finishes introducing the values, it says how many numbers in the vector are even

```
vect = zeros(1,20);
count = 0;
for i=1:20
    vect(i) = input ('Introduce a number: ');
    if (rem(vect(i),2) ==0)
        count = count + 1;
    end
end
fprintf('\There are %d even numbers', count);
```

You could even improve the solution by checking if the number is even when the user introduce it

Working with vectors

Exercise: Write a program that asks the user to introduce 5 characters one by one, and prints on screen the number of vowels introduced

Working with vectors

Exercise: Write a program that asks the user to introduce 5 characters one by one, and prints on screen the number of vowels introduced

```
vectCh = [];
 for i=1:5
     vectCh(i) = input('Introduce a character: ','s');
 end
 countvow = 0;
  for i = 1:5
     switch vectCh(i)
              case {'a','e','i','o','u'}
                       countvow = countvow + 1;
     end
end
fprintf('\n There are %d vowels', countvow);
```

Working with vectors

Exercise: Write a program that asks the user to introduce 5 characters one by one, and prints on screen the number of vowels introduced

```
vectCh = [];
for i=1:5
     vectCh(i) = input('Introduce a charac
end
countvow = 0;
 for i = 1:5
     switch vectCh(i)
              case {'a','e','i','o','u'}
                       countvow = countvow +
     end
end
fprintf('\n There are %d vowels', countvow);
```

We initialize the vector with an "empty vector". We cannot use zeros, as we are going to store characters, and the elements in the vector should all be of the same types (you can't have numbers and characters in the same vector)

Working with vectors

Exercise: Write a program that asks the user to introduce 5 characters one by one, and prints on screen the number of vowels introduced

```
vectCh = [];
 for i=1:5
      vectCh(i) = input('Introduce a character: ','s');
 end
 countvow = 0;
  for letter = vectCh
      switch letter
              case {'a','e','i','o','u'}
                      countvow = countvow + 1;
      end
end
fprintf('\n There are %d vowels', countvow);
```

Another possible solution

Working with vectors

Exercise: Write a program that asks the user to introduce 5 characters one by one, and prints on screen the number of vowels introduced

```
 countvow = 0;
for i=1:5
     ch = input('Introduce a character: ','s');
     switch ch
              case {'a','e','i','o','u'}
                        countvow = countvow + 1;
     end
end
fprintf('\n There are %d vowels', countvow);
```

Another possible solution. This time the problem didn't say anything about vectors, so we could solve the problem like this.

Working with vectors

Exercise: Write a program that asks the user to introduce 5 numbers and prints on screen the highest number introduced. Solve it using ***for*** (do not use the function *max*)

Working with vectors

```
vect = zeros(1,5);
for i=1:5
     vect(i) = input ('Introduce a number: ');
end
maxNum = vect (1);
for var = vect
    if (var > maxNum)
          maxNum = var;
    end
end
fprintf('\The maximun is %d', maxNum);
```

Before we start we consider as maximum the first number in the vector

Working with vectors

```
vect = zeros(1,5);
for i=1:5
     vect(i) = input ('Introduce a number: ');
end
maxNum = vect (1);
for var = vect
    if (var > maxNum)
          maxNum = var;
    end
end
fprintf('\The maximun is %d', maxNum);
```

At each iteration we compare the "maximun so far", which is stored in the variable maxNum, with a new element taken from the vector (var).
If this number is greater than the maximum so far, then it will be our new maximum

Working with vectors

An example of an execution of the program

```
vect = zeros(1,5);
 for i=1:5
      vect(i) = input ('Introduce a number: ');
 end
maxNum = vect (1);
 for var = vect
      if (var > maxNum)
             maxNum = var;
      end
 end
 fprintf('\The maximun is %d', maxNum);
```

Let's say the user has introduced the values 2 1 4 3 1, so the content of vect is [2 1 4 3 1]

Working with vectors

An example of an execution of the program

vect = [2 1 4 3 1]

```
vect = zeros(1,5);
for i=1:5
     vect(i) = input ('Introduce a number: ');
end
maxNum = vect (1);
for var = vect
     if (var > maxNum)
          maxNum = var;
     end
end
fprintf('\The maximun is %d', maxNum);
```

$\longrightarrow$ maxNum = 2

Working with vectors

An example of an execution of the program

```
vect = zeros(1,5);
 for i=1:5
      vect(i) = input ('Introduce a number: ');
 end
maxNum = vect (1);
 for var = vect
     if (var > maxNum)
           maxNum = var;
     end
 end
 fprintf('\The maximun is %d', maxNum);
```

vect = [**2** 1 4 3 1]

1st Iteration

var = 2

It is our maximun

Working with vectors

An example of an execution of the program

```
vect = zeros(1,5);
 for i=1:5
       vect(i) = input ('Introduce a number: ');
 end
maxNum = vect (1);
 for var = vect
     if (var > maxNum)
            maxNum = var;
     end
 end
 fprintf('\The maximun is %d', maxNum);
```

vect = [2 **1** 4 3 1]

$\longrightarrow$ 2nd Iteration

var = 1

1 is smaller than 2, nothing changes

Working with vectors

An example of an execution of the program

```
vect = zeros(1,5);
for i=1:5
     vect(i) = input ('Introduce a number: ');
end
maxNum = vect (1);
for var = vect
     if (var > maxNum)
           maxNum = var;
     end
end
fprintf('\The maximun is %d', maxNum);
```

vect = [2 1 **4** 3 1]

3rd Iteration
var = 4
4 is greater than 2,
maxNum = 4

Working with vectors

An example of an execution of the program

```
vect = zeros(1,5);
for i=1:5
     vect(i) = input ('Introduce a number: ');
end
maxNum = vect (1);
for var = vect
     if (var > maxNum)
          maxNum = var;
     end
end
fprintf('\The maximun is %d', maxNum);
```

vect = [2 1 4 **3** 1]

4th Iteration

var = 3

3 is smaller than 4, nothing changes

Working with vectors

An example of an execution of the program

vect = [2 1 4 3 **1**]

```
vect = zeros(1,5);
for i=1:5
     vect(i) = input ('Introduce a number: ');
end
maxNum = vect (1);
for var = vect
     if (var > maxNum)
          maxNum = var;
     end
end
fprintf('\The maximun is %d', maxNum);
```

$\longrightarrow$ 5th Iteration

var = 1

1 is smaller than 4, nothing changes

Working with vectors

An example of an execution of the program

```
vect = zeros(1,5);
for i=1:5
     vect(i) = input ('Introduce a number: ');
end
maxNum = vect (1);
for var = vect
    if (var > maxNum)
          maxNum = var;
    end
end
fprintf('\The maximun is %d', maxNum);
```

vect = [2 1 4 3 1]

$\longrightarrow$ The value of maxNum is 4

Working with vectors

Another way of solving the exercise

```
vect = zeros(1,5);
 for i=1:5
      vect(i) = input ('Introduce a number: ');
 end
maxNum = vect (1);
 for i = 2:length(vect)
     if (vect (i) > maxNum)
          maxNum = vect(i);
     end
 end
 fprintf('\The maximun is %d', maxNum);
```

The function length returns the length of the vector. In this case the variable i is going to take values between 2 and the length of the vector. Then we compare the number of the vector in the ith position, in the same way as we did before

Working with vectors

Another different way of solving the exercise.
This time we are not using vectors

```
num = input ('Introduce a number: ');
maxNum = num;
for i = 1:4
    num = input ('Introduce a number: ');
    if (num > maxNum)
        maxNum = num;
    end
end
fprintf('\The maximun is %d', maxNum);
```

We ask for a number which, initially, it will be our “maximun so far”

Now we ask 4 numbers more, comparing each of them with the “maximun so far” and updating the value of this latter if they are higher