

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

Dati interi e in virgola mobile

Le classi numeriche in MATLAB® includono numeri interi con e senza segno e numeri in virgola mobile a singola e doppia precisione. Per impostazione predefinita, MATLAB memorizza tutti i valori numerici in virgola mobile a doppia precisione. (La precisione e il tipo predefiniti non possono essere modificati). È possibile scegliere di memorizzare qualsiasi numero, o matrice di numeri, come numeri interi o a singola precisione. Gli array di numeri interi e a singola precisione offrono una memorizzazione più efficiente rispetto alla doppia precisione.

Tutti i tipi numerici supportano le operazioni di base degli array, come l'indicizzazione, il rimodellamento e le operazioni matematiche.

double

Array a doppia precisione

Descrizione

`double` è il tipo di dati numerici (classe) predefinito in MATLAB® che fornisce una precisione sufficiente per la maggior parte delle attività di calcolo. Le variabili numeriche sono memorizzate automaticamente come valori in virgola mobile a doppia precisione a 64 bit (8 byte). Ad esempio:

```
x = 10;  
whos x
```

Name	Size	Bytes	Class	Attributes
x	1x1	8	double	

MATLAB costruisce il tipo di dati `double` secondo lo standard IEEE® 754 per la doppia precisione. L'intervallo per un numero negativo di tipo `double` è compreso tra $-1,79769 \times 10^{308}$ e $-2,22507 \times 10^{-308}$, mentre quello per i numeri positivi è compreso tra $2,22507 \times 10^{-308}$ e $1,79769 \times 10^{308}$.

Per maggiori informazioni sui valori in virgola mobile a doppia precisione e a precisione singola, vedere [Floating-Point Numbers](#).

eps

Floating-point relative accuracy

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

Dati interi e in virgola mobile

Inf

Create array of all Inf values

Examples

Matrix of Inf Values

Create a 3-by-3 matrix of Inf values.

```
X = Inf(3)
```

X = 3×3

Inf	Inf	Inf
Inf	Inf	Inf
Inf	Inf	Inf

NaN

Create array of all NaN values

Examples

Matrix of NaN Values

Create a 3-by-3 matrix of NaN values.

```
X = NaN(3)
```

X = 3×3

NaN	NaN	NaN
NaN	NaN	NaN
NaN	NaN	NaN

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

Dati interi e in virgola mobile

Le classi numeriche in MATLAB® includono numeri interi con e senza segno e numeri in virgola mobile a singola e doppia precisione. Per impostazione predefinita, MATLAB memorizza tutti i valori numerici in virgola mobile a doppia precisione. (La precisione e il tipo predefiniti non possono essere modificati). È possibile scegliere di memorizzare qualsiasi numero, o matrice di numeri, come numeri interi o a singola precisione. Gli array di numeri interi e a singola precisione offrono una memorizzazione più efficiente rispetto alla doppia precisione.

Tutti i tipi numerici supportano le operazioni di base degli array, come l'indicizzazione, il rimodellamento e le operazioni matematiche.

▼ Tipo e valore della query

<code>allfinite</code>	Determine if all array elements are finite (Da R2022a)
<code>anynan</code>	Determine if any array element is NaN (Da R2022a)
<code>isinteger</code>	Determine whether input is integer array
<code>isfloat</code>	Determine if input is floating-point array
<code>isnumeric</code>	Determine whether input is numeric array
<code>isreal</code>	Determine whether array uses complex storage
<code>isfinite</code>	Determine which array elements are finite
<code>isinf</code>	Determine which array elements are infinite
<code>isnan</code>	Determinare quali elementi dell'array sono NaN

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

allfinite

Determine if all array elements are finite.
Since R2022a

Syntax

```
TF = allfinite(A)
```

Description

TF = allfinite(A) returns logical 1 (true) if all elements of A are finite. It returns 0 (false) if any element is not finite. If A contains complex numbers, allfinite(A) returns 1 if all elements have finite real and imaginary parts, and 0 otherwise.

▼ Determine If All Matrix Elements Are Finite

Create a matrix and determine if all elements are finite.

```
A = [0 0 3;0 0 3;0 0 NaN]
```

A = 3×3

```
0     0     3
0     0     3
0     0    NaN
```

```
TF = allfinite(A)
```

TF = logical
0

Examples

▼ Determine If All Vector Elements Are Finite

Create a row vector and determine if all elements are finite.

```
A = 1./[-2 -1 1e-23 0.1]
```

A = 1×4
10²³ ×

```
-0.0000 -0.0000 1.0000 0.0000
```

```
TF = allfinite(A)
```

TF = logical
1

Create another row vector and determine if all elements are finite.

```
B = 0./[-2 -1 0 0.1]
```

B = 1×4

```
0     0    NaN     0
```

```
TF = allfinite(B)
```

TF = logical
0

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

anynan

Determine if any array element is NaN
Since R2022a

Syntax

```
TF = anynan(A)
```

Description

`TF = anynan(A)` returns logical 1 (true) if at least one element of A is NaN. It returns 0 (false) if no element is NaN. If A contains complex numbers, `anynan(A)` returns 1 if at least one element has a real or imaginary part that is NaN.

Examples

▼ Determine If Vector Contains NaN

Create a row vector A. Determine if at least one element of A is NaN.

▼ Determine If Matrix Contains NaN

Create a matrix and determine if at least one of its elements is NaN.

```
A = [0 0 3;0 0 3;0 0 NaN]
```

```
A = 3x3
```

```
0     0     3
0     0     3
0     0    NaN
```

```
TF = anynan(A)
```

```
TF = logical
1
```

Examples

▼ Determine If Vector Contains NaN

Create a row vector A. Determine if at least one element of A is NaN.

```
A = 0./[-2 -1 0 1 2]
```

```
A = 1x5
```

```
0     0    NaN     0     0
```

```
TF = anynan(A)
```

```
TF = logical
1
```

`anynan` returns logical 1 (true) because at least one element of A is NaN.

Create another row vector B. Determine if at least one element of B is NaN.

```
B = [-2 -1 1 2]/0
```

```
B = 1x4
```

```
-Inf -Inf  Inf  Inf
```

```
TF = anynan(B)
```

```
TF = logical
0
```

`anynan` returns logical 0 (false) because no element of B is NaN.

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

isfinite

Determine which array elements are finite

Syntax

```
TF = isfinite(A)
```

Description

`TF = isfinite(A)` returns a logical array containing 1 (true) where the elements of the array A are finite, and 0 (false) where they are infinite or NaN. If A contains complex numbers, `isfinite(A)` contains 1 for elements with finite real and imaginary parts, and 0 for elements where either part is infinite or NaN.

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

isfinite

Determine which array elements are finite

Syntax

```
TF = isfinite(A)
```

Description

`TF = isfinite(A)` returns a logical array containing 1 (true) where the elements of the array `A` are finite, and 0 (false) where the elements are either infinite or NaN.

Examples

▼ Determine Finite Real Elements

Create a row vector and determine the finite real elements.

```
A = 1./[-2 -1 0 1 2]
```

```
A = 1x5
```

```
-0.5000 -1.0000 Inf 1.0000 0.5000
```

```
TF = isfinite(A)
```

```
TF = 1x5 logical array
```

```
1 1 0 1 1
```

or elements

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

isinf

Determine which array elements are infinite

Syntax

```
TF = isinf(A)
```

Description

`TF = isinf(A)` returns a logical array containing 1 (true) where the elements of the array `A` are `Inf` or `-Inf`, and 0 (false) where they are not. If `A` contains complex numbers, `isinf(A)` contains 1 for elements with infinite real or imaginary part, and 0 for elements where both real and imaginary parts are finite or NaN.

Examples

▼ Determine Infinite Real Elements

Create a row vector and determine the infinite elements.

```
A = 1./[-2 -1 0 1 2]
```

```
A = 1x5
```

```
-0.5000 -1.0000 Inf 1.0000 0.5000
```

```
TF = isinf(A)
```

```
TF = 1x5 Logical array
```

```
0 0 1 0 0
```


ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

isnan

Determinare quali elementi dell'array sono NaN

Sintassi

```
TF = isnan(A)
```

Descrizione

TF = isnan(A) restituisce un array logico contenente 1 (true) dove gli elementi di A sono NaN e 0 (false) dove non lo sono. Se A contiene numeri complessi, isnan(A) contiene 1 per gli elementi la cui parte reale o immaginaria è NaN e 0 per gli elementi dove sia le parti reali che quelle immaginarie non sono NaN.

Esempi

▼ Determinazione degli elementi NaN in numeri reali

Creare un vettore riga e determinare quali elementi sono NaN.

```
A = 0./[-2 -1 0 1 2]
```

```
A = 1x5
```

```
0    0    NaN    0    0
```

```
TF = isnan(A)
```

```
TF = 1x5 logical array
```

```
0    0    1    0    0
```

▼ Sostituzione degli elementi NaN

Creare un array e trovare gli elementi con valori NaN.

```
A = [1,3,5,7,NaN,10,NaN,4,6,8]
```

```
A = 1x10
```

```
1    3    5    7    NaN    10    NaN    4    6    8
```

```
TF = isnan(A)
```

```
TF = 1x10 logical array
```

```
0    0    0    0    1    0    1    0    0    0
```

Indicizzare su A con TF per accedere agli elementi di A che sono NaN. Sostituire i valori NaN con 0.

```
A(TF) = 0
```

```
A = 1x10
```

```
1    3    5    7    0    10    0    4    6    8
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

Infinity and NaN

Infinity

MATLAB® represents infinity by the special value `Inf`. Infinity results from operations like division by zero and overflow, which lead to results too large to represent as conventional floating-point values. MATLAB also provides a function called `Inf` that returns the IEEE® arithmetic representation for positive infinity as a `double` scalar value.

Several examples of statements that return positive or negative infinity in MATLAB are shown here.

<pre>x = 1/0 x = Inf</pre>	<pre>x = 1.e1000 x = Inf</pre>
<pre>x = exp(1000) x = Inf</pre>	<pre>x = log(0) x = -Inf</pre>

Use the `isinf` function to verify that `x` is positive or negative infinity:

```
x = log(0);

isinf(x)
ans =
    1
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

Infinity and NaN

NaN

MATLAB represents values that are not real or complex numbers with a special value called NaN, which stands for "Not a Number". Expressions like `0/0` and `inf/inf` result in NaN, as do any arithmetic operations involving a NaN:

```
x = 0/0  
x =
```

```
NaN
```

You can also create NaNs by:

```
x = NaN;
```

```
whos x  
Name      Size      Bytes  Class  
  
x          1x1           8  double
```

The `NaN` function returns one of the IEEE arithmetic representations for NaN as a double scalar value. The exact bit-wise hexadecimal representation of this NaN value is,

```
format hex  
x = NaN  
  
x =  
  
fff8000000000000
```

Always use the `isnan` function to verify that the elements in an array are NaN:

```
isnan(x)  
ans =  
  
1
```

MATLAB preserves the "Not a Number" status of alternate NaN representations and treats all of the different representations of NaN equivalently. However, in some special cases (perhaps due to hardware limitations), MATLAB does not preserve the exact bit pattern of alternate NaN representations throughout an entire calculation, and instead uses the canonical NaN bit pattern defined above.

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

Infinity and NaN

NaN

Logical Operations on NaN

Because two NaNs are not equal to each other, logical operations involving NaN always return false, except for a test for inequality, (NaN ~= NaN):

```
NaN > NaN  
ans =  
0
```

```
NaN ~= NaN  
ans =  
1
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Tipi numerici

Empty Matrices

If you construct a matrix using empty matrix elements, the empty matrices are ignored in the resulting matrix:

```
A = [5.36; 7.01; []; 9.44]
A =
    5.3600
    7.0100
    9.4400
```

Concatenation Examples

Combining Character and Double Types

Combining character values with double values yields a character matrix. MATLAB® converts the double elements in this example to their character equivalents:

```
x = ['A' 'B' 'C' 68 69 70]
x =
    ABCDEF

class(x)
ans =
    char
```

Combining Logical and Double Types

Combining logical values with double values yields a double matrix. MATLAB converts the logical true and false elements in this example to double:

```
x = [true false false pi sqrt(7)]
x =
    1.0000         0         0    3.1416    2.6458

class(x)
ans =
    double
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Testo in array di caratteri e in array di stringhe

Gli array di caratteri e gli array di stringhe consentono di memorizzare i dati di testo in MATLAB®.

- Un array di caratteri è una sequenza di caratteri così come un array numerico è una sequenza di numeri. Un uso tipico è quello di memorizzare brevi porzioni di testo come *vettori di caratteri*, come `c = 'Hello World'`.
- Un array di stringhe è un contenitore per porzioni di testo. Gli array di stringhe forniscono un insieme di funzioni per lavorare con il testo come dati. È possibile creare stringhe utilizzando le virgolette doppie, come `str = "Greetings friend"`. Per convertire i dati in array di stringhe, utilizzare la funzione `string`.

Funzioni

▼ Creazione, concatenazione e conversione

Array di stringhe

<code>string</code>	String array
<code>strings</code>	Create string array with no characters
<code>join</code>	Combine strings
<code>plus</code>	Add numbers, append strings

Array di caratteri

<code>char</code>	Character array
<code>cellstr</code>	Convert to cell array of character vectors
<code>blanks</code>	Create character array of blanks
<code>newline</code>	Creare il carattere nuova riga

Array di stringhe o di caratteri

<code>compose</code>	Format data into multiple strings
<code>sprintf</code>	Format data into string or character vector
<code>strcat</code>	Concatenare le stringhe orizzontalmente
<code>append</code>	Combine strings

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Testo in array di caratteri e in array di stringhe

Gli array di caratteri e gli array di stringhe consentono di memorizzare i dati di testo in MATLAB®.

- Un array di caratteri è una sequenza di caratteri così come un array numerico è una sequenza di numeri. Un uso tipico è quello di memorizzare brevi porzioni di testo come *vettori di caratteri*, come `c = 'Hello World'`.
- Un array di stringhe è un contenitore per porzioni di testo. Gli array di stringhe forniscono un insieme di funzioni per lavorare con il testo come dati. È possibile creare stringhe utilizzando le virgolette doppie, come `str = "Greetings friend"`. Per convertire i dati in array di stringhe, utilizzare la funzione `string`.

Funzioni

▼ Creazione, concatenazione e conversione

Conversione degli argomenti di input

<code>convertCharsToStrings</code>	Convert character arrays to string arrays, leaving other arrays unaltered
<code>convertStringsToChars</code>	Convert string arrays to character arrays, leaving other arrays unaltered
<code>convertContainedStringsToChars</code>	Convert string arrays at any level of cell array or structure

Conversione tra numeri e stringhe

<code>double</code>	Array a doppia precisione
<code>string</code>	String array
<code>str2double</code>	Convert strings to double precision values
<code>num2str</code>	Convert numbers to character array

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Testo in array di caratteri e in array di stringhe

Gli array di caratteri e gli array di stringhe consentono di memorizzare i dati di testo in MATLAB®.

- Un array di caratteri è una sequenza di caratteri così come un array numerico è una sequenza di numeri. Un uso tipico è quello di memorizzare brevi porzioni di testo come *vettori di caratteri*, come `c = 'Hello World'`.
- Un array di stringhe è un contenitore per porzioni di testo. Gli array di stringhe forniscono un insieme di funzioni per lavorare con il testo come dati. È possibile creare stringhe utilizzando le virgolette doppie, come `str = "Greetings friend"`. Per convertire i dati in array di stringhe, utilizzare la funzione `string`.

Funzioni

▼ Determinazione del tipo e delle proprietà

Tipo di dati

<code>ischar</code>	Determine if input is character array
<code>iscellstr</code>	Determine if input is cell array of character vectors
<code>isstring</code>	Determine if input is string array
<code>isStringScalar</code>	Determine if input is string array with one element

Proprietà del testo

<code>strlength</code>	Lengths of strings
<code>isstrprop</code>	Determine which characters in input strings are of specified category
<code>isletter</code>	Determine which characters are letters
<code>isspace</code>	Determine which characters are space characters

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Testo in array di caratteri e in array di stringhe

Gli array di caratteri e gli array di stringhe consentono di memorizzare i dati di testo in MATLAB®.

- Un array di caratteri è una sequenza di caratteri così come un array numerico è una sequenza di numeri. Un uso tipico è quello di memorizzare brevi porzioni di testo come *vettori di caratteri*, come `c = 'Hello World'`.
- Un array di stringhe è un contenitore per porzioni di testo. Gli array di stringhe forniscono un insieme di funzioni per lavorare con il testo come dati. È possibile creare stringhe utilizzando le virgolette doppie, come `str = "Greetings friend"`. Per convertire i dati in array di stringhe, utilizzare la funzione `string`.

Funzioni

▼ Ricerca e sostituzione

Ricerca

<code>contains</code>	Determine if pattern is in strings
<code>matches</code>	Determine if pattern matches strings (Da R2019b)
<code>count</code>	Count occurrences of pattern in strings
<code>endsWith</code>	Determine if strings end with pattern
<code>startsWith</code>	Determine if strings start with pattern
<code>strfind</code>	Find strings within other strings
<code>sscanf</code>	Read formatted data from strings

Sostituzione

<code>replace</code>	Find and replace one or more substrings
<code>replaceBetween</code>	Replace substrings between start and end points
<code>strrep</code>	Find and replace substrings

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Testo in array di caratteri e in array di stringhe

Gli array di caratteri e gli array di stringhe consentono di memorizzare i dati di testo in MATLAB®.

- Un array di caratteri è una sequenza di caratteri così come un array numerico è una sequenza di numeri. Un uso tipico è quello di memorizzare brevi porzioni di testo come *vettori di caratteri*, come `c = 'Hello World'`.
- Un array di stringhe è un contenitore per porzioni di testo. Gli array di stringhe forniscono un insieme di funzioni per lavorare con il testo come dati. È possibile creare stringhe utilizzando le virgolette doppie, come `str = "Greetings friend"`. Per convertire i dati in array di stringhe, utilizzare la funzione `string`.

Funzioni

▼ Unione e divisione

<code>join</code>	Combine strings
<code>plus</code>	Add numbers, append strings
<code>split</code>	Split strings at delimiters
<code>splitlines</code>	Split strings at newline characters
<code>strjoin</code>	Join strings in array
<code>strsplit</code>	Split string or character vector at specified delimiter
<code>strtok</code>	Selected parts of strings
<code>extract</code>	Extract substrings from strings (<i>Da R2020b</i>)
<code>extractAfter</code>	Extract substrings after specified positions
<code>extractBefore</code>	Extract substrings before specified positions
<code>extractBetween</code>	Extract substrings between start and end points

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Testo in array di caratteri e in array di stringhe

Gli array di caratteri e gli array di stringhe consentono di memorizzare i dati di testo in MATLAB®.

- Un array di caratteri è una sequenza di caratteri così come un array numerico è una sequenza di numeri. Un uso tipico è quello di memorizzare brevi porzioni di testo come *vettori di caratteri*, come `c = 'Hello World'`.
- Un array di stringhe è un contenitore per porzioni di testo. Gli array di stringhe forniscono un insieme di funzioni per lavorare con il testo come dati. È possibile creare stringhe utilizzando le virgolette doppie, come `str = "Greetings friend"`. Per convertire i dati in array di stringhe, utilizzare la funzione `string`.

Funzioni

▼ Confronti

<code>matches</code>	Determine if pattern matches strings (<i>Da R2019b</i>)
<code>strcmp</code>	Confrontare stringhe
<code>strcmpi</code>	Compare strings (case insensitive)
<code>strncmp</code>	Compare first n characters of strings (case sensitive)
<code>strncmpi</code>	Compare first n characters of strings (case insensitive)

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

There are two ways to represent text in MATLAB®. You can store text in string arrays and in character vectors. MATLAB displays strings with double quotes and character vectors with single quotes.

Represent Text with String Arrays

You can store any 1-by-n sequence of characters as a string, using the `string` data type. Enclose text in double quotes to create a string.

```
str = "Hello, world"
```

```
str =  
"Hello, world"
```

Though the text "Hello, world" is 12 characters long, `str` itself is a 1-by-1 string, or *string scalar*. You can use a string scalar to specify a file name, plot label, or any other piece of textual information.

To find the number of characters in a string, use the `strlength` function.

```
n = strlength(str)
```

```
n = 12
```

If the text includes double quotes, use two double quotes within the definition.

```
str = "They said, ""Welcome!"" and waved."
```

```
str =  
"They said, "Welcome!" and waved."
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Represent Text with String Arrays

To add text to the end of a string, use the plus operator, `+`. If a variable can be converted to a string, then `plus` converts it and appends it.

```
fahrenheit = 71;  
celsius = (fahrenheit-32)/1.8;  
tempText = "temperature is " + celsius + "C"
```

```
tempText =  
"temperature is 21.6667C"
```

You can also concatenate text using the `append` function.

```
tempText2 = append("Today's ",tempText)
```

```
tempText2 =  
"Today's temperature is 21.6667C"
```

The `string` function can convert different types of inputs, such as numeric, datetime, duration, and categorical values. For example, convert the output of `pi` to a string.

```
ps = string(pi)
```

```
ps =  
"3.1416"
```

You can store multiple pieces of text in a string array. Each element of the array can contain a string having a different number of characters, without padding.

```
str = ["Mercury","Gemini","Apollo";...  
      "Skylab","Skylab B","ISS"]
```

```
str = 2x3 string  
    "Mercury"    "Gemini"    "Apollo"  
    "Skylab"     "Skylab B"   "ISS"
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Represent Text with String Arrays

`str` is a 2-by-3 string array. You can find the lengths of the strings with the `strlength` function.

```
N = strlength(str)
```

N = 2×3

7	6	6
6	8	3

String arrays are supported throughout MATLAB and MathWorks® products. Functions that accept character arrays (and cell arrays of character vectors) as inputs also accept string arrays.

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Represent Text with Character Vectors

To store a 1-by-n sequence of characters as a character vector, using the `char` data type, enclose it in single quotes.

```
chr = 'Hello, world'
```

```
chr =  
'Hello, world'
```

The text 'Hello, world' is 12 characters long, and `chr` stores it as a 1-by-12 character vector.

```
whos chr
```

Name	Size	Bytes	Class	Attributes
chr	1x12	24	char	

If the text includes single quotes, use two single quotes within the definition.

```
chr = 'They said, ''Welcome!'' and waved.'
```

```
chr =  
'They said, 'Welcome!' and waved.'
```

Character vectors have two principal uses:

- To specify single pieces of text, such as file names and plot labels.
- To represent data that is encoded using characters. In such cases, you might need easy access to individual characters.

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

String arrays store pieces of text and provide a set of functions for working with text as data. You can index into, reshape, and concatenate strings arrays just as you can with arrays of any other type. You also can access the characters in a string and append text to strings using the plus operator. To rearrange strings within a string array, use functions such as `split`, `join`, and `sort`.

Create String Arrays from Variables

MATLAB® provides string arrays to store pieces of text. Each element of a string array contains a 1-by-n sequence of characters.

You can create a string using double quotes.

```
str = "Hello, world"
```

```
str =  
"Hello, world"
```

As an alternative, you can convert a character vector to a string using the `string` function. `chr` is a 1-by-17 character vector. `str` is a 1-by-1 string that has the same text as the character vector.

```
chr = 'Greetings, friend'
```

```
chr =  
'Greetings, friend'
```

```
str = string(chr)
```

```
str =  
"Greetings, friend"
```


ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Create String Arrays from Variables

Create a string array containing multiple strings using the `[]` operator. `str` is a 2-by-3 string array that contains six strings.

```
str = ["Mercury", "Gemini", "Apollo";  
       "Skylab", "Skylab B", "ISS"]  
  
str = 2x3 string  
    "Mercury"    "Gemini"    "Apollo"  
    "Skylab"    "Skylab B"  "ISS"
```

Find the length of each string in `str` with the `strlength` function. Use `strlength`, not `length`, to determine the number of characters in strings.

```
L = strlength(str)
```

```
L = 2x3
```

```
7    6    6  
6    8    3
```

As an alternative, you can convert a cell array of character vectors to a string array using the `string` function. MATLAB displays strings in string arrays with double quotes, and displays character vectors in cell arrays with single quotes.

```
C = {'Mercury', 'Venus', 'Earth'}
```

```
C = 1x3 cell  
    {'Mercury'}    {'Venus'}    {'Earth'}
```

```
str = string(C)
```

```
str = 1x3 string  
    "Mercury"    "Venus"    "Earth"
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Create String Arrays from Variables

In addition to character vectors, you can convert numeric, datetime, duration, and categorical values to strings using the `string` function.

Convert a numeric array to a string array.

```
X = [5 10 20 3.1416];  
string(X)
```

```
ans = 1x4 string  
    "5"    "10"    "20"    "3.1416"
```

Convert a datetime value to a string.

```
d = datetime('now');  
string(d)
```

```
ans =  
"12-Feb-2024 23:24:55"
```

Also, you can read text from files into string arrays using the `readtable`, `textscan`, and `fscanf` functions.

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Create Empty and Missing Strings

String arrays can contain both empty and missing values. An empty string contains zero characters. When you display an empty string, the result is a pair of double quotes with nothing between them (""). The missing string is the string equivalent to NaN for numeric arrays. It indicates where a string array has missing values. When you display a missing string, the result is <missing>, with no quotation marks.

Create an empty string array using the `strings` function. When you call `strings` with no arguments, it returns an empty string. Note that the size of `str` is 1-by-1, not 0-by-0. However, `str` contains zero characters.

```
str = strings
```

```
str =  
""
```

Create an empty character vector using single quotes. Note that the size of `chr` is 0-by-0.

```
chr = ''
```

```
chr =
```

```
0x0 empty char array
```

Create a string array where every element is an empty string. You can preallocate a string array with the `strings` function.

```
str = strings(2,3)
```

```
str = 2x3 string  
"" "" ""  
"" "" ""
```

To create a missing string, convert a missing value using the `string` function. The missing string displays as <missing>.

```
str = string(missing)
```

```
str =  
<missing>
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Create Empty and Missing Strings

You can create a string array with both empty and missing strings. Use the `ismissing` function to determine which elements are strings with missing values. Note that the empty string is not a missing string.

```
str(1) = "";  
str(2) = "Gemini";  
str(3) = string(missing)
```

```
str = 1x3 string  
    ""    "Gemini"    <missing>
```

```
ismissing(str)
```

```
ans = 1x3 logical array
```

```
0    0    1
```

Compare a missing string to another string. The result is always 0 (false), even when you compare a missing string to another missing string.

```
str = string(missing);  
str == "Gemini"
```

```
ans = logical  
0
```

```
str == string(missing)
```

```
ans = logical  
0
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Access Elements of String Array

String arrays support array operations such as indexing and reshaping. Use array indexing to access the first row of `str` and all the columns.

```
str = ["Mercury", "Gemini", "Apollo";  
       "Skylab", "Skylab B", "ISS"];  
str(1,:)
```

```
ans = 1x3 string  
      "Mercury"    "Gemini"    "Apollo"
```

Access the second element in the second row of `str`.

```
str(2,2)
```

```
ans =  
"Skylab B"
```

Assign a new string outside the bounds of `str`. MATLAB expands the array and fills unallocated elements with missing values.

```
str(3,4) = "Mir"
```

```
str = 3x4 string  
      "Mercury"    "Gemini"    "Apollo"    <missing>  
      "Skylab"     "Skylab B"  "ISS"       <missing>  
      <missing>    <missing>    <missing>    "Mir"
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Access Characters Within Strings

You can index into a string array using curly braces, {}, to access characters directly. Use curly braces when you need to access and modify characters within a string element. Indexing with curly braces provides compatibility for code that could work with either string arrays or cell arrays of character vectors. But whenever possible, use string functions to work with the characters in strings.

Access the second element in the second row with curly braces. `chr` is a character vector, not a string.

```
str = ["Mercury", "Gemini", "Apollo";  
       "Skylab", "Skylab B", "ISS"];  
chr = str{2,2}
```

```
chr =  
'Skylab B'
```

Access the character vector and return the first three characters.

```
str{2,2}(1:3)
```

```
ans =  
'Sky'
```

Find the space characters in a string and replace them with dashes. Use the `isspace` function to inspect individual characters within the string. `isspace` returns a logical vector that contains a true value wherever there is a space character. Finally, display the modified string element, `str(2,2)`.

```
TF = isspace(str{2,2})
```

```
TF = 1x8 logical array
```

```
0 0 0 0 0 0 1 0
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Access Characters Within Strings

```
str{2,2}(TF) = "-";  
str(2,2)
```

```
ans =  
"Skylab-B"
```

Note that in this case, you can also replace spaces using the `replace` function, without resorting to curly brace indexing.

```
replace(str(2,2)," ","-")
```

```
ans =  
"Skylab-B"
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Concatenate Strings into String Array

Concatenate strings into a string array just as you would concatenate arrays of any other kind.

Concatenate two string arrays using square brackets, [].

```
str1 = ["Mercury", "Gemini", "Apollo"];  
str2 = ["Skylab", "Skylab B", "ISS"];  
str = [str1 str2]
```

```
str = 1x6 string  
    "Mercury"    "Gemini"    "Apollo"    "Skylab"    "Skylab B"    "ISS"
```

Transpose str1 and str2. Concatenate them and then vertically concatenate column headings onto the string array. When you concatenate character vectors into a string array, the character vectors are automatically converted to strings.

```
str1 = str1';  
str2 = str2';  
str = [str1 str2];  
str = [{"Mission:" , "Station:"} ; str]
```

```
str = 4x2 string  
    "Mission:"    "Station:"  
    "Mercury"     "Skylab"  
    "Gemini"      "Skylab B"  
    "Apollo"      "ISS"
```


ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Append Text to Strings

To append text to strings, use the plus operator, +. The plus operator appends text to strings but does not change the size of a string array.

Append a last name to an array of names. If you append a character vector to strings, then the character vector is automatically converted to a string.

```
names = ["Mary"; "John"; "Elizabeth"; "Paul"; "Ann"];  
names = names + ' Smith'
```

```
names = 5x1 string  
"Mary Smith"  
"John Smith"  
"Elizabeth Smith"  
"Paul Smith"  
"Ann Smith"
```

Append different last names. You can append text to a string array from a string array or from a cell array of character vectors. When you add nonscalar arrays, they must be the same size.

```
names = ["Mary"; "John"; "Elizabeth"; "Paul"; "Ann"];  
lastnames = ["Jones"; "Adams"; "Young"; "Burns"; "Spencer"];  
names = names + " " + lastnames
```

```
names = 5x1 string  
"Mary Jones"  
"John Adams"  
"Elizabeth Young"  
"Paul Burns"  
"Ann Spencer"
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Text in String and Character Arrays

Create String Arrays

Append Text to Strings

Append a missing string. When you append a missing string with the plus operator, the output is a missing string.

```
str1 = "Jones";  
str2 = string(missing);  
str1 + str2
```

```
ans =  
<missing>
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Analyze Text Data with String Arrays

Compare Character Vectors

You can compare character vectors and cell arrays of character vectors to each other. Use the `strcmp` function to compare two character vectors, or `strncmp` to compare the first *N* characters. You also can use `strcmpi` and `strncmpi` for case-insensitive comparisons.

Compare two character vectors with the `strcmp` function. `chr1` and `chr2` are not equal.

```
chr1 = 'hello';  
chr2 = 'help';  
TF = strcmp(chr1,chr2)
```

```
TF = logical  
    0
```

Note that the MATLAB `strcmp` differs from the C version of `strcmp`. The C version of `strcmp` returns 0 when two character arrays are the same, not when they are different.

Compare the first two characters with the `strncmp` function. `TF` is 1 because both character vectors start with the characters `he`.

```
TF = strncmp(chr1,chr2,2)
```

```
TF = logical  
    1
```

ESERCITAZIONI

MATLAB – Nozioni fondamentali



Tipi di dati

Caratteri e stringhe

Argomenti

Analyze Text Data with String Arrays

Compare Character Vectors

Compare two cell arrays of character vectors. `strcmp` returns a logical array that is the same size as the cell arrays.

```
C1 = {'pizza'; 'chips'; 'candy'};  
C2 = {'pizza'; 'chocolate'; 'pretzels'};  
strcmp(C1,C2)
```

ans = 3x1 logical array

```
1  
0  
0
```

Riferimenti Bibliografici

[1] <https://it.mathworks.com>