UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Fondamenti di Informatica

Ing. Alba Amato, PhD

alba.amato@unina2.it

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

IL LINGUAGGIO Matlab

Lucidi tratti da:

Alla scoperta dei fondamenti dell'informatica. Un viaggio nel mondo dei bit
di Angelo Chianese, Vincenzo Moscato, Antonio Picariello

capitolo 7 -par. 7.1, 7.2, 7.3

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Caratteristiche Generali

- ***MATLAB*** è un vero e proprio ambiente di progetto rivolto principalmente **allo sviluppo di programmi per l'analisi numerica**. Tale ambiente utilizza un linguaggio di programmazione con una **sintassi simile a quella del C**, e mette a disposizione del programmatore tutti i più noti "tipi elementari" di dato:
  - dati di natura numerico: o tipo intero, o tipo reale,
  - dati di natura alfanumerica: o caratteri, o stringhe.
- In particolare l'instanziazione di una variabile numerica avviene attraverso la sintassi: **var=val**
  - dove **var** e **val** rappresentano rispettivamente il nome ed il valore della variabile numerica.
- Mentre l'instanziazione di una variabile alfanumerica avviene attraverso la sintassi: **var='val'**
  - dove **var** e **val** rappresentano rispettivamente il nome ed il valore della variabile alfanumerica. Una variabile alfanumerica è contraddistinta dalla presenza degli apici, all'inizio e alla fine del valore della variabile stessa.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Caratteristiche Generali

- Altra caratteristica fondamentale di MATLAB è il concetto di *funzione*. Ogni programma, sottoprogramma, procedura o funzione del linguaggio è sempre trattata come una funzione costituita o meno da un insieme di parametri di ingresso-uscita.
- Ogni dichiarazione di funzione MATLAB è preceduta dalla parola chiave **function**.
- Per quanto riguarda le istruzioni, il linguaggio MATLAB fornisce sia ***enunciati o statement semplici che di controllo***.
- Matlab è un programma con licenza commerciale, quindi a pagamento.
- TUTTAVIA è lo standard industriale in molti campi, ed è molto utilizzato.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Istruzioni Semplici

- Quelli semplici sono:

  - l'*assegnazione*, che fornisce ad una variabile il valore che si ottiene calcolando il risultato di un'espressione composta in generale di costanti, variabili, operatori e funzioni; essa si indica (se var è la variabile che riceve il valore ed E è l'espressione che lo fornisce) con un sintassi del tipo: **var=E**

  - l'*attivazione di funzioni*, eventualmente con il passaggio dei dati da elaborare e con la ricezione di uno più risultati. Si noti che MATLAB, come del resto altri linguaggi, mette a disposizione un insieme molto ampio di funzioni predefinite quali ad esempio quelle per il calcolo numerico oppure quelle per la lettura e la scrittura di dati sui supporti di ingresso e uscita.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Istruzioni di controllo

- Gli enunciati di controllo si dividono al loro volta in:
  - enunciati composti, che si ottengono disponendo in sequenza enunciati semplici, di selezione e di iterazione;
  - enunciati di selezione;
  - enunciati di iterazione.
- Come anticipato il linguaggio MATLAB prevede la costruzione di programmi con la struttura di funzioni composte da un unico blocco di istruzioni, costituente la *sezione esecutiva*.
- A differenza di altri linguaggi, manca nel corpo di una funzione MATLAB la *sezione dichiarativa* in quanto **tutte le variabili sono automaticamente trattate e riconosciute o come set di reali o di caratteri**.
- Una funzione può richiamare o essere richiamata da altre funzioni e può essere o meno caratterizzata dalla presenza di parametri di ingresso e di uscita

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Esempio

programma MATLAB che richiama due funzioni.

```
function nome_programma
        enunciato 1 del programma
                ..........................
        funzioneA (lista_parametri_ingresso)
                ..........................
        [lista_paramteri_ucita] = funzioneB(lista_parametri_ingresso)
                ..........................
        enunciato N del programma

function funzioneA(lista_parametri_ingresso)
        enunciato 1 della funzioneA
        ........................
        enunciato N1 della funzioneA

function [lista_paramteri_ucita] = funzioneB(lista_parametri_ingresso)
        enunciato 1 della funzioneB
        ........................
        enunciato N2 della funzioneB
```

- La funzione A è un esempio di funzione avente solo parametri di ingresso.
- La funzione B è un esempio di funzione avente parametri sia di ingresso che di uscita.
- Il programma principale è esso stesso una funzione senza parametri.
- Tutte le variabili usate all'interno di una funzione MATLAB sono ad essa locali, ossia hanno una visibilità limitata alla sola funzione in cui sono usate.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Vocabolario

- Il ***vocabolario*** del linguaggio è costituito da sequenze di lunghezza finita di caratteri trattate come singole entità logiche. Tali entità sono raggruppate in cinque classi differenti per finalità e caratteristiche.
- Le 5 classi sono:
  - separatori;
  - identificatori e parole riservate;
  - simboli speciali (operatori, delimitatori, frasi di commento);
  - numeri interi e decimali;
  - sequenze di caratteri racchiuse tra apici;
- Si noti che le terza e la quarta classe contengono le costanti (numeriche e alfanumeriche) gestite dal linguaggio.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Separatori

- Ciascuna entità lessicale è separata dalla successiva mediante opportuni ***separatori***.
- I caratteri *spazio* ed *ENTER* (fine linea) sono considerati separatori espliciti, mentre gli operatori (aritmetici e di relazione), il punto e virgola e l'operatore dell'assegnazione (=) vengono implicitamente identificati come separatori.
- Così le frasi che seguono presentano due entità lessicali distinte:
- **alfa = 10**
- **if contatore**
- mentre le frasi seguenti ne presentano soltanto una:
- **ifcontatore**
- **whilecondizione**

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Identificatori e Parole Chiave

- Gli ***identificatori*** permettono di indicare i nomi di programmi, costanti, variabili e funzioni.
- Un identificatore è una stringa di caratteri che deve soddisfare i seguenti vincoli:
  - deve essere composta da lettere, cifre e dal carattere '_';
  - il primo carattere deve essere una lettera;
  - deve essere costituito da un numero limitato di caratteri;
  - non deve ovviamente contenere al suo interno lo spazio.
- L'esempio sottolinea il fatto che MATLAB è "case sensitive" ovvero fa differenza tra lettere maiuscole e minuscole per cui ALFA e Alfa sono considerati distinti.

Esempi di identificatori sono:

A
a
ALFA
Alfa
SOL1
PI_greco
equaz_2_grado

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Parole Riservate o Chiavi

- Alcuni identificatori hanno un ben preciso e congelato significato nell'ambito del linguaggio e pertanto prendono il nome di ***parole riservate*** o ***chiavi***.
- Tali parole sono riservate per usi particolari e non sono ridefinibili in altro modo dal programmatore in quanto sono “le chiavi” che guidano nella costruzione delle frasi del linguaggio.
- Esempi di parole chiavi del linguaggio sono: **if**, **function**, **while**, **else**, **end**.
- Accanto agli identificatori riservati, esiste un insieme di identificatori con significato ormai standard, che costituiscono l'ambiente di lavoro del linguaggio.
- In MATLAB ne esistono tantissimi, la maggior parte di questi fa riferimento a funzioni matematiche. Alcuni esempi sono: **abs**, **max**, **min**, **sin**, **cos, sqrt, mean**, etc…

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Simboli Speciali

- I simboli speciali sono o semplici caratteri o coppie adiacenti di essi ed indicano le operazioni predefinite presenti nel linguaggio e tutta la punteggiatura richiesta dalla sintassi. Essi sono i seguenti:

+ - * /

(operatori aritmetici)

< > == <= >= ~=

(operatori di confronto)

, ; . ‘ : .. …

(delimitatori e punteggiatura)

( ) [ ] { }

(parentesi)

& | ~ ()

operatori logici)

%

(commenti)

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Simboli Speciali

- **Operatori Aritmetici**
- Gli operatori aritmetici sono i classici operatori binari +,- ,*,/ e l'operatore di modulo %. La divisione tra interi tronca la parte frazionaria, mentre l'espressione x%y fornisce il resto della divisione di x per y. L'operatore di modulo può essere applicato solo a tipi interi. Gli operatori + e – hanno la stessa priorità, priorità inferiore a * e / (che invece hanno identica priorità).
- **Operatori Relazionali**
- Gli operatori relazionali sono:

  > >= < <=
- mentre gli operatori di eguaglianza (e disuguaglianza) sono:

  == ~=
- Gli operatori di relazione hanno priorità maggiore rispetto a quelli di eguaglianza/disuguaglianza. Gli operatori relazionali hanno invece priorità minore rispetto agli operatori aritmetici.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Simboli Speciali

- **Operatori Logici**
- Gli operatori logici sono:

  & (and), | (or) e ~ (not)
- **&** ha priorità maggiore di **|** ed entrambi hanno priorità inferiore agli operatori relazionali e di uguaglianza.
- **I delimitatori**
- Il linguaggio MATLAB non ha dei delimitatori canonici per delimitare le frasi del programma. L'unico carattere che ha una funzione di questo tipo è il carattere ';' (punto e virgola). **Tale carattere viene utilizzato come termine di ogni frase del linguaggio. Se non lo si inserisce al termine della frase, allora viene prodotta in automatico la stampa del valore delle variabili calcolate nella istruzione; se invece si chiude la frase con il punto e virgola tale stampa viene disabilitata.**

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Frasi Di Commento

- Come tutti gli altri linguaggi di programmazione anche MATLAB consente di introdurre nel programma frasi prive di ogni valore esecutivo o dichiarativo che consentono di migliorare la leggibilità e la chiarezza del programma. Esse servono unicamente ad uno scambio di messaggi tra le persone.
- Tali frasi prendono il nome di frasi di commento. Nel linguaggio una frase che inizia con il simbolo speciale '%' è un commento

```
% la variabile alfa deve avere valore negativo
% i>=10 implica una condizione di errore
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Le Costanti Numeriche

- I ***numeri*** trattati dal linguaggio sono di due tipi, interi e reali. Un numero intero è una sequenza di cifre eventualmente preceduta da un segno. Un numero reale ha una parte decimale ed eventualmente un fattore di scala.
- Sono costanti intere:

  10
  300
  -1000

Sono costanti reali:

3.400
10.20
+1.99E+30
-30.008
-0.7E-10
9.02E3

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Le Costanti Stringhe

- Le ***costanti stringa di caratteri*** sono sequenze di caratteri racchiuse da una coppia di caratteri ' (apici). Il valore della stringa è dato dalla sequenza di caratteri esclusi gli apici che fungono da parentesi.
- Una costante senza caratteri (una coppia di apici) rappresenta la stringa a lunghezza nulla.
- Sono costanti stringhe di caratteri:

  'ALFABETO'

  'ciao'

  'Minuscolo'

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Struttura del Programma

- Da un punto di vista testuale, il programma è visto come un insieme di frasi a cui viene costituito da una *intestazione* che ne riassume molto brevemente il significato ed un *blocco* che può essere visto come una entità sintattica che contiene la parte elaborativa del programma. Ogni intestazione è sempre preceduta dalla parola chiave **function**.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

La dichiarazione e gestione dei tipi

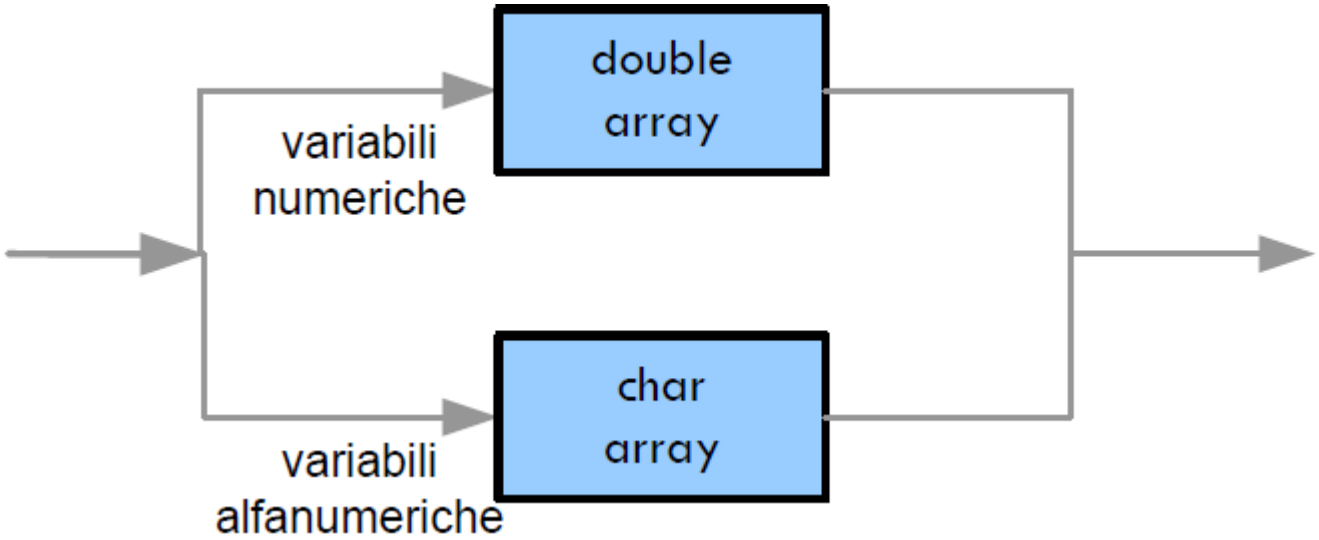
- In MATLAB non esiste il concetto di ***dichiarazione*** "esplicita" di tipo.
- Ogni variabile appartiene, in maniera implicita, ad uno dei tipi elementari messi a disposizione dal linguaggio.
- A differenza dei classici linguaggi di programmazione, il linguaggio MATLAB utilizza due soli tipi elementari di dato, o meglio, due sole classi elementari di oggetti:
  - *double array* (array bidimensionale di numeri reali);
  - *char array* (array bidimensionale di caratteri).

MATLAB deriva da "matrix laboratory". Questo perché tratta quasi tutti i dati come come una matrice. In particolare le variabili sono per MATLAB delle matrici 1x1.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

La dichiarazione e gestione dei tipi

- La classe *double array* permette la gestione di tutti dati di tipo numerico come interi, reali a singola precisione e reali a doppia precisione.
- La classe *char array* permette la gestione di tutti i dati di tipo alfanumerico come caratteri e stringhe.
- Un oggetto di tali classi ha una struttura simile a quella di una matrice.



UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Semplici: il tipo intero

- Il ***tipo intero*** contiene il sottoinsieme dei numeri interi compresi nell'intervallo [-MAXINT, MAXINT] con MAXINT valore costante predefinito.
- Il valore di MAXINT dipende dalla rappresentazione interna dei numeri interi ed è quindi dipendente dal particolare sistema di calcolo.
- Sulle variabili di tipo intero sono definite le operazioni di somma, sottrazione, divisione e moltiplicazione più moltissime altre funzioni tra cui ricordiamo: la funzione resto in modulo **MOD**, la funzione valore assoluto **ABS**, la funzione segno **SIGN** e così via.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Semplici: il tipo reale

- Il ***tipo reale*** contiene un intervallo di estremi predefiniti dell'insieme dei numeri reali.
- Sui valori del tipo reale sono definite le operazioni di somma, sottrazione, divisione e di moltiplicazione, più un insieme ampio di funzioni più avanzate.
- Sono inoltre definite le due funzioni di conversione da reale a intero **floor(x)**, **ceil(x)** e **round(x)**.

  floor (0.6) = 0 restituisce la parte intera di x

  ceil (0.6) = 1 restituisce la parte intera di x+1

  round(0.6) = 1 calcola l'intero mediante un arrotondamento

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Semplici: il tipo char

- Le costanti che compongono il ***tipo char*** formano un insieme finito e ordinato di caratteri.
  - Un carattere è denotato da uno dei simboli di tale tabella racchiuso tra apici (ad esempio **'m'**).
- L'ordinamento può variare a seconda del sistema di calcolo e dipende dal codice usato per la rappresentazione in memoria.
- Solitamente viene impiegato il codice ASCII a sette bit, per cui l'ordinamento viene stabilito dalla posizione del carattere all'interno della tabella introdotta dall'ASCII.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Semplici: il tipo booleano

- In MATLAB non esiste il tipo booleano, ma questo può essere simulato con una variabile intera che può assumere due soli valori (0 e 1): uno corrispondente a FALSE, l'altro a TRUE.
- Sui valori logici sono definite le operazioni di NOT (**~**), AND (**&**) ed OR (**!**).

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Strutturati: il tipo array

- La struttura ***array*** è composta da un insieme di elementi tutti dello stesso tipo e con un unico nome collettivo. Può avere una o più dimensioni fino ad un massimo prefissato. Ad ogni elemento dell'array è possibile accedere mediante un indice (o un numero di indici uguale alle dimensioni stabilite) che ne individua la posizione all'interno della struttura.
- In MATLAB il numero di elementi dell'array non è definito a priori e può cambiare nel corso del programma. Un array monodimensionale di interi o reali coincide con un oggetto della classe *double array* di dimensione 1xN, con N pari al numero di elementi contenuti nell'array.
- Un array monodimensionale di caratteri coincide con un oggetto della classe *char array* di dimensione 1xN, con N pari al numero di elementi contenuti nell'array.
- Gli elementi del vettore vanno racchiusi tra parentesi quadre e separati da uno spazio bianco o da una virgola.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Strutturati: il tipo array

a=[1 2 3];
b=[7,2,8,6];

Nel primo caso si è creato il vettore **a** di tre elementi con valori 1, 2 e 3; nel secondo caso si è creato il vettore **b** con quattro elementi 7, 2, 8 e 6.

- Un matrice di caratteri coincide con un oggetto della classe *char array* di dimensione MxN, con M pari al numero di righe della matrice e N pari al numero di colonne. Gli elementi della matrice vanno racchiusi tra parentesi quadre, gli elementi delle righe sono separati da uno spazio bianco o da una virgola, mentre ogni riga è separata dalla successiva mediante il carattere punto e virgola ';'.
- Con: **a=[1 2 3 4; 5 6 7 8; 9 10 11 12];** si crea una matrice di tre righe e quattro colonne i cui elementi hanno i valori indicati.

$$a \equiv \begin{vmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{vmatrix}$$

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Strutturati: il tipo array

- La selezione di uno dei componenti dell'array si effettua facendo seguire al nome dell'array il valore dell'indice, o degli indici separati da virgola nel caso di array a più dimensioni, racchiusi tra una coppia di parentesi tonde.
- Sono esempi di funzioni di accesso:

  a(3,(I*J+2))
  a(I,J)
  a(3)

- Nel caso di array con informazioni strutturate, si devono comporre le varie funzioni di accesso. Le operazioni consentite sugli array sono quelle che si possono fare sui componenti l'array stesso.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Strutturati: il tipo array

- In MATLAB anche i tipi semplici sono visti come casi particolari di quelli strutturati. In MATLAB le variabili intere, reali e caratteri sono gestite mediante un oggetto della classe *double array* di dimensione 1x1, mentre i caratteri mediante un oggetto della classe *char array* di dimensione 1x1

```
a=1;
Name    Size      Class
a       1x1       double array
a=1.5;
Name    Size      Class
a       1x1       double array
a='a';
Name    Size      Class
a       1x1       char array
```

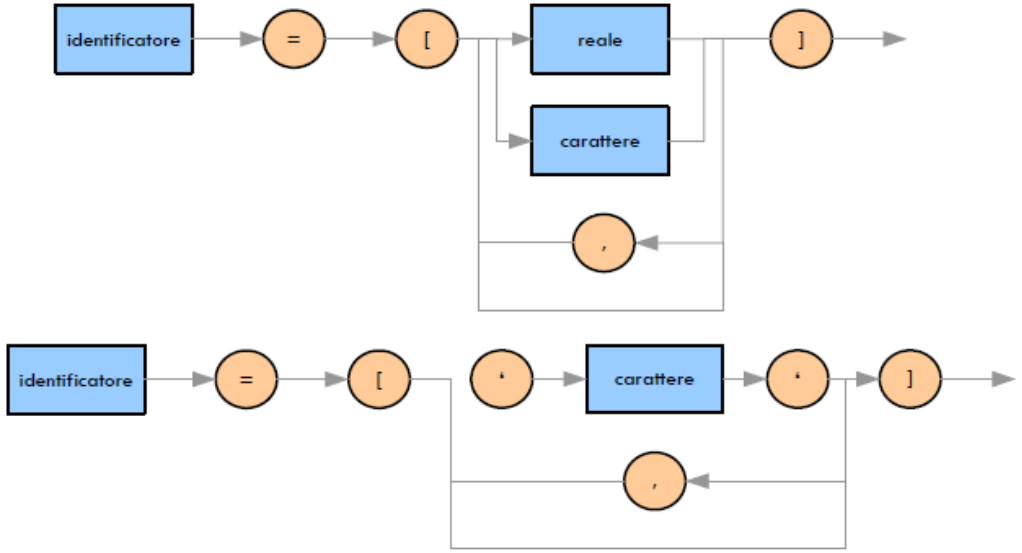


Figura 8 – Carta sintattica per la costruzione di array monodimensionali

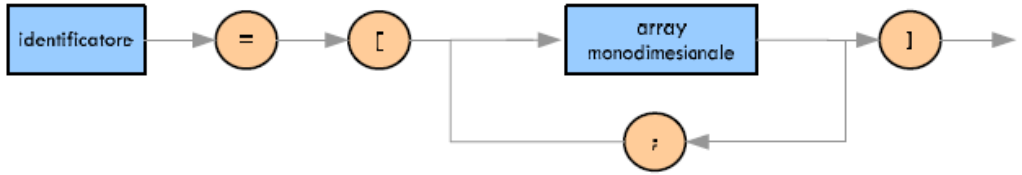


Figura 9 – Carta sintattica per per la costruzione di array bidimensionali

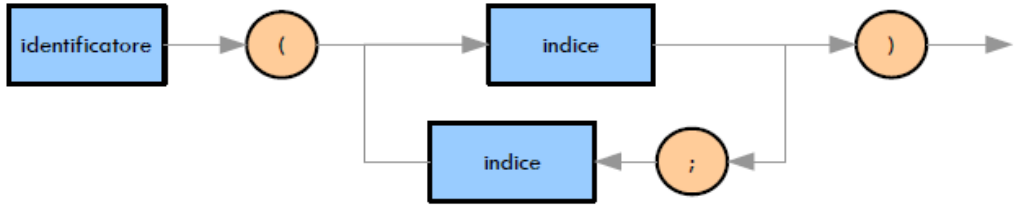
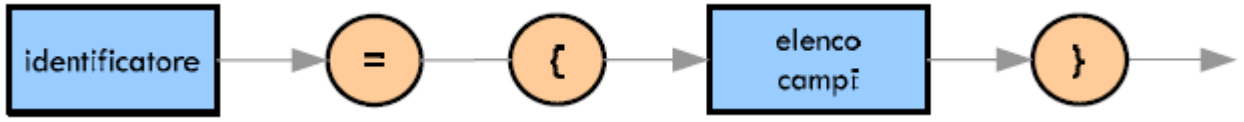


Figura 10 – Carta sintattica l'accesso agli elementi di array

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Strutturati: il tipo record

- La struttura ***record*** è composta da un numero prefissato di componenti, anche di tipo differente.
- Ogni componente, detto anche campo del record, ha un suo nome e tipo.
- La definizione di un record prevede, allora, l'elencazione delle variabili, sia semplici che a loro volta strutturate, costituenti i campi componenti.
- Le operazioni consentite sul record sono quelle che si possono fare sui campi.
- In MATLAB la definizione di un record avviene definendo il nome del record e ponendo tra parentesi graffe le variabili costituenti i campi del record separate da virgole. Il valore delle variabili deve essere noto all'atto della definizione del record. **MATLAB tratta i record come oggetti della classe non elementare *cell array* (array di celle). Una cella (campo del record) può essere a sua volta o di tipo *char array* o di tipo *double array*.**
- Esempio: definizione del record '**agenda**'.
- L'accesso ad un campo si effettua specificando il nome del record e l'indice del campo compreso tra parentesi graffe. Se un campo è un'informazione strutturata, allora si deve per esso applicare la funzione di accesso caratteristica del tipo di appartenenza.




```
Nominativo='Angelo Chianese';
Indirizzo='via Claudio 121';
citta='Napoli';
tel='39081777777';
agenda={Nominativo,Indirizzo,citta,tel}
```

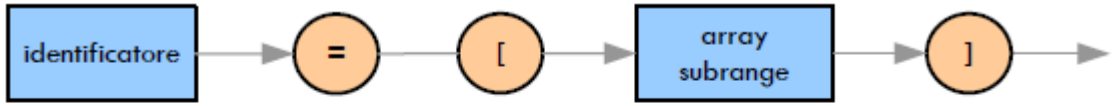
```
agenda{1}
agenda{3}
```

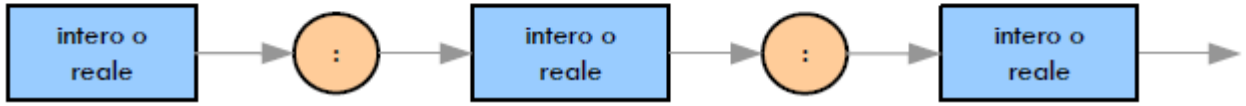
UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Strutturati: il tipo Intervallo

- Tratta solo intervalli di natura numerica.
- Per definire un intervallo devono essere in generale specificati l'estremo iniziale (***ei***), quello finale (***ef***) ed un passo (***p***): in tale caso si generano tutti i valori che si ottengono calcolando **x = x + *p*** con valore iniziale di **x** uguale a ***ei*** e valore finale uguale a ***ef***.
- Non è obbligatorio specificare il passo: nel caso venga omesso assume automaticamente il valore unitario.
- La sintassi per la generazione di un intervallo è la seguente: **t=[ei:passo:ef]**
- Con il tipo intervallo si introduce di fatto un con tanti valori quanti sono quelli determinati dalla dichiarazione di intervallo.

Istruzione	Intervallo generato
t=[1:10]	1 2 3 4 5 6 7 8 9 10
t=[0:0.1:1]	0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1





UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Tipi Strutturati: il tipo file

- Un ***file*** è una sequenza di componenti tutti dello stesso tipo che risiede in modo permanente sui supporti di massa.
- Un file può essere visto come un dato strutturato utilizzabile per la registrazione di un qualsiasi numero di elementi.
- In MATLAB non esiste un tipo file, le operazioni di I/O da e verso file sono ottenute aprendo dei veri e propri canali di comunicazione tra il programma e la memoria di massa.

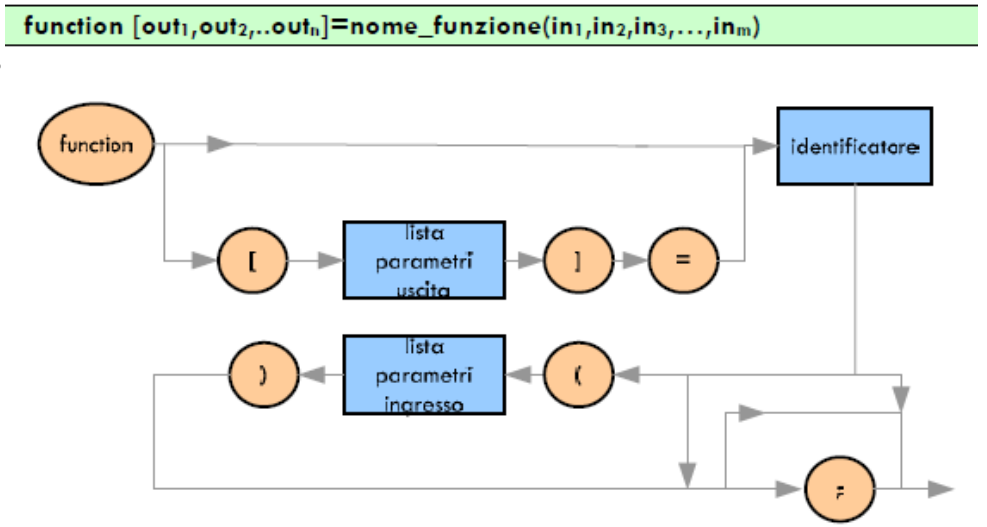
UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

La dichiarazione di Funzione

- MATLAB, come già anticipato, utilizza il concetto di ***funzione*** per la creazione di programmi e sottoprogrammi.
- La dichiarazione di funzione si compone di due parti, l'*intestazione* e il *blocco* di enunciati. L'intestazione è composta dalla parola chiave **function**, l'identificatore della funzione e due liste: la prima contenente gli eventuali parametri di uscita separati da virgola e racchiusi in parentesi quadre, la seconda contenente gli eventuali parametri di ingresso separati da virgola e racchiusi in parentesi tonde. Tali liste possono essere omesse nel caso di mancanza dei parametri di ingresso o uscita.

Esempio di intestazione di funzione a ***n*** uscite e a ***m*** ingressi.

function [out1,out2,..outn]=nome_funzione(in1,in2,in3,…,inm)



i parametri attuali sono forniti all'atto del richiamo e rappresentano i dati effettivamente elaborati;
i parametri formali sono quelli usati all'interno della procedura per poterla scrivere.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

La specifica dell'algoritmo

- La ***specificazione delle azioni elaborative del programma*** si traduce in un enunciato o statement composto.
- Lo statement composto è formato da una sequenza di statement.
- Gli statement possono essere o meno separati dal carattere punto e virgola. Gli statement semplici che compongono la specificazione dell'algoritmo denotano azioni elaborative che devono essere svolte dall'esecutore macchina.
- Gli statement sono:
  - *assegnazione,*
  - *if … else,*
  - *while,*
  - *switch … case,*
  - *for,*
  - *richiamo funzione.*

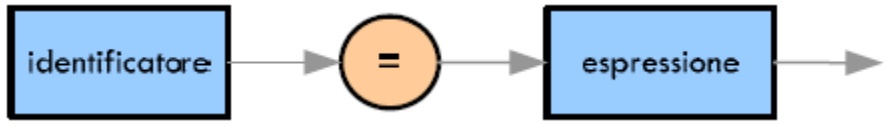
UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Assegnazione

- L'enunciato di assegnazione è la più elementare forma di enunciato. Esso prescrive che venga dapprima calcolato il valore dell'*espressione* che si trova a destra del '=', e che tale valore venga poi assegnato alla *variabile* che si trova a sinistra dello stesso simbolo. La variabile e il valore dell'espressione devono essere dello stesso tipo con l'eccezione che, se la variabile è reale, allora il tipo dell'espressione può anche essere intero.
- Se in una espressione sono presenti più operandi, occorre specificare la priorità della loro esecuzione nel caso in cui non sia esplicitata mediante l'introduzione nell'espressione delle parentesi. In particolare, il linguaggio prevede quattro classi di operatori a ciascuna delle quali è assegnata una precedenza.
- In mancanza di parentesi, deve essere svolta prima l'operazione con operatore con precedenza maggiore.
- Ciascun operatore è applicabile solo a particolari tipi di operandi e il risultato della operazione risulterà del tipo stabilito dal linguaggio.

Gli operatori in ordine di precedenza decrescente sono:
- l'operatore di negazione logica;
- gli operatori detti moltiplicativi;
- gli operatori detti additivi;
- gli operatori di relazione.

```
alfa = 3.569
circonferenza = 2 * PI_GRECO * raggio
area_rettangolo = base * altezza
i = i - 1
```

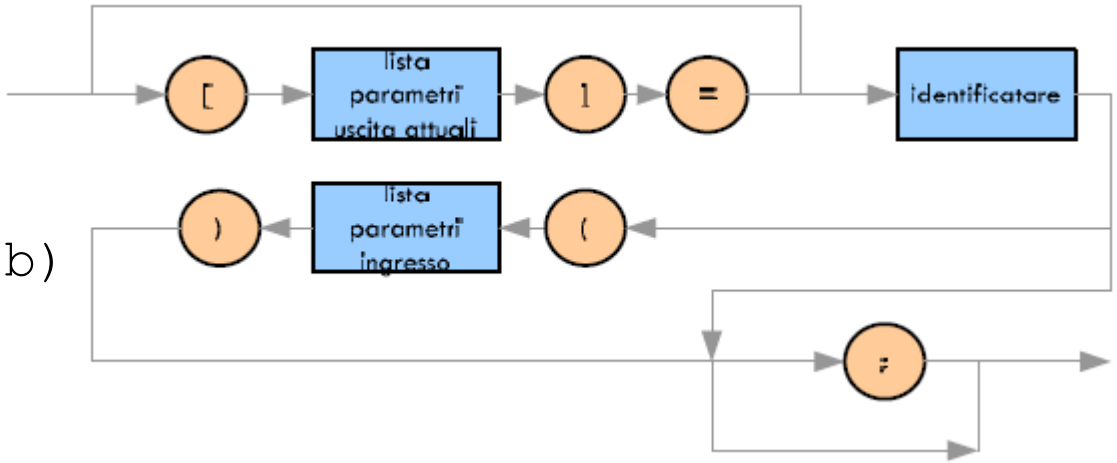


UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Richiamo della Funzione

- Il ***richiamo della funzione*** determina l'esecuzione della funzione indicata. Affinché il richiamo sia corretto, si deve fornire una lista di parametri uguale, in numero e tipo, a quella specificata nella dichiarazione della funzione. La corrispondenza si stabilisce per ordine, nel senso che al primo parametro attuale è associato il primo formale, e così via.
- **[parametri di uscita attuali]=nome_funzione(parametri di ingresso attuali)**

```
function s = somma(a, b)
s = a+b
```



UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Funzioni Predefinite

- Il linguaggio mette a disposizione un ampio set di funzioni predefinite applicabili a tipi semplici e strutturati, per lo più di natura matematica. Tali funzioni possono essere introdotte all'interno di un programma, alcuni esempi sono:

  cos(x)
  sqrt(x)
  mean(v)
  max(v)
  det(A)

- Di particolare interesse sono le funzioni di conversione tra caratteri e reali e tra reali e caratteri:

  char(126) = ~
  double('a') = 97

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Funzioni per I/O

- Il linguaggio fornisce alcune funzioni per le operazioni di lettura e scrittura dei valori delle variabili dai file INPUT e OUTPUT. In genere il file di INPUT rappresenta la tastiera del sistema di calcolo; quello di OUTPUT il monitor.
- Essi possono essere visti come organizzati in un testo costituito da sequenze di RIGHI separati tra loro dal carattere di fine rigo CR (Carriage Return). La sequenza di righi è terminata dal carattere di fine file. All'interno di ogni rigo sono distribuite le rappresentazioni delle informazioni opportunamente separate usando il carattere spazio.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Funzioni per I/O: Lettura

- In MATLAB la lettura da un file avviene aprendo un canale di comunicazione con la memoria di massa attraverso il concetto di 'apertura' di un file. Un file deve essere aperto prima di essere utilizzato. L'apertura di un file avviene attraverso la funzione: **[fid,message]=fopen (filename, permission)**
- dove **fid** è l'identificativo del file, **message** è il messaggio associato all'apertura del file, **filename** è il nome del file (path su disco del file) e **permission** è la modalità con cui si apre il file (in questo caso lettura).
- Dopo l'apertura, con l'identificatore del file è possibile eseguire le operazioni di lettura che avvengono con l'utilizzo della funzione: **[A] = fscanf(fid, format)**
- dove **fid** è l'identificativo del file, **format** indica il tipo degli elementi da leggere ed **A** è la matrice in cui si inseriscono gli elementi letti. La variabile format può assumere i seguenti valori a seconda del tipo di dato che si vuole leggere:

%d interi
%f, %g reali
%c caratteri
%s stringhe

esempio di lettura da file:

```
[fid,message]=fopen ('c:\file.txt','r);
[v_input]=fscanf(fid,'%s')
v_input = ciao
```

La lettura di informazioni da standard input (tastiera) avviene attraverso la funzione:

var_input=input(msg,['s'])

dove **var_input** rappresenta la variabile in cui verrà memorizzato il valore immesso da tastiera e **msg** è un messaggio visualizzato a video. Il parametro **'s'** va specificato solo per le variabili di tipo stringa.

```
var_input=input('Inserisci un numero:')
var_input=input('Inserisci una stringa:','s')
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Funzioni per I/O: Scrittura

- Analogamente all'operazione di lettura l'apertura di un file avviene attraverso la funzione: **[fid,message]=fopen (filename, permission)**
- dove **fid** è l'identificativo del file, **message** è il messagio associato all'apertura del file, **filename** è il nome del file (path su disco del file) e **permission** è la modalità con cui si apre il file (in questo caso scrittura).
- A questo punto attraverso l'identificatore del file è possibile eseguire le operazioni di scrittura che avvengono con l'utilizzo della funzione: **[count] = fprintf (fid, format,A)**
- dove **fid** è l'identificativo del file, **format** indica il tipo degli elementi da scrivere, **A** è la matrice in cui sono presenti gli elementi da scrivere e **count** conta il numero di elementi effettivamente scritti su file. La variabile format assume gli stessi valori previsti nella lettura.

```
[fid,message]=fopen ('c:\file.txt','w');
stringa='ciao';
[count]=fprintf(fid,stringa,'%s')
count = 4
[fid,message]=fopen ('c:\file.txt','r');
[v_input]=fscanf(fid,'%s')
v_input = ciao'
```

%d interi
%f, %g reali
%c caratteri
%s stringhe

La scrittura di informazioni su standard output (monitor) avviene attraverso la funzione stessa **fprintf** senza specificare l'identificativo del file:

fprintf ('%s','Messaggio visualizzato a video')

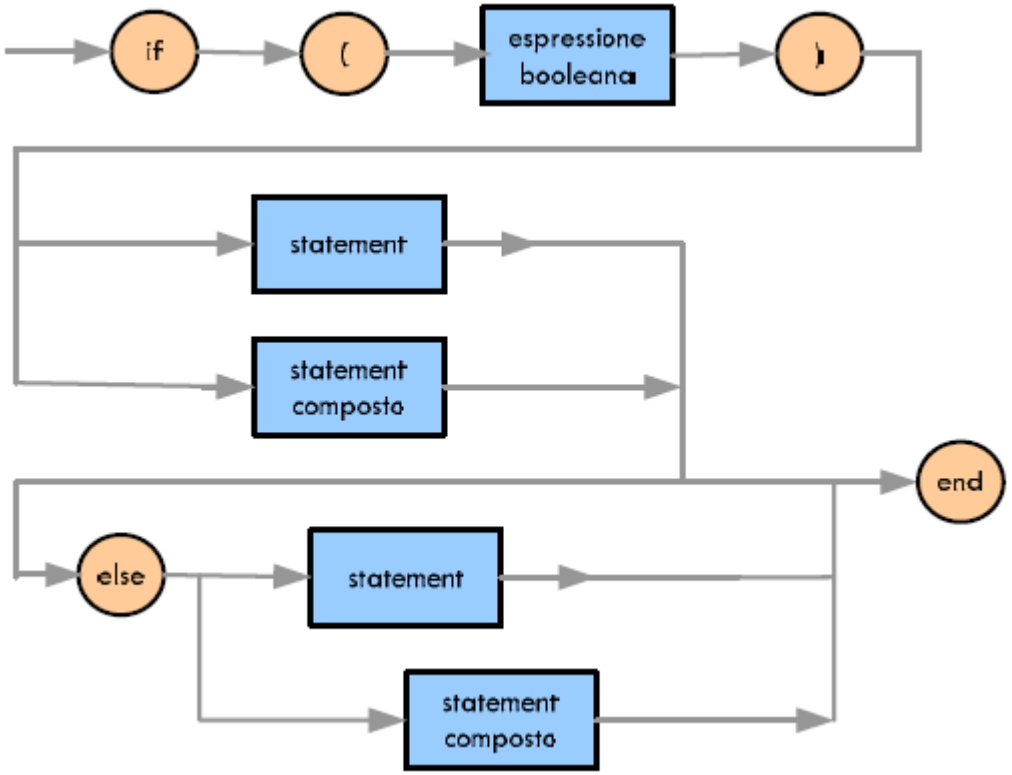
UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Il costrutto IF-ELSE

- Nel liguaggio MATLAB ogni costrutto **if-else** va terminato con la parola chiave **end**.
- Sono esempi di IF:

```
if (n>10)
    n=n+1;
else
    n=n-1;
end
```

```
if (n>10)
    n=0;
end;
n=n-1;
```



UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Il costrutto SWITCH-CASE

- L'enunciato ***switch-case*** permette di scegliere nel caso in cui le alternative siano più di due. Esso consiste di un'espressione (di tipo numerico o carattere), detta *selettore*, e di una lista di enunciati, ciascuno dei quali identificato da uno o più valori costanti appartenenti al tipo del selettore preceduti dalla parola **case**.
- L'enunciato scelto è quello identificato dalla costante che è uguale al valore calcolato del selettore.
- Una volta eseguito l'enunciato scelto si esce dalla struttura **switch**.
- La scansione dei valori costanti, per cercare quello con valore uguale al selettore, avviene in modo sequenziale per cui vanno disposti per ultimi gli enunciati che hanno la minore probabilità di essere scelti.
- Qualora il valore calcolato dal selettore non è uguale ai valori costanti indicati viene eseguito l'enunciato preceduto dalla parola chiave **otherwise** se presente, altrimenti si esce dallo struttura

```
SWITCH(colore_semaforo)
CASE 'rosso'
      msg='stop';
CASE 'verde'
      msg='vai';
CASE 'giallo'
      msg='attenzione';
OTHERWISE
      msg='semaforo rotto';
END
```

```
SWITCH(numero mese)
CASE {1,3,5,7,8,10,12}
      msg='mese di 31 giorni';
CASE {4,6,9,11}
      msg='mese di 30 giorni';
CASE 2
      msg='mese di 28 o 29 giorni';
OTHERWISE
      msg='mese non valido';
END
```

```
SWITCH(valcifra)
CASE      0      carcifra='0';
CASE      1      carcifra='1';
CASE      2      carcifra='2';
CASE      3      carcifra='3';
CASE      4      carcifra='4';
CASE      5      carcifra='5';
CASE      6      carcifra='6';
CASE      7      carcifra='7';
CASE      8      carcifra='8';
CASE      9      carcifra='9';
END
```

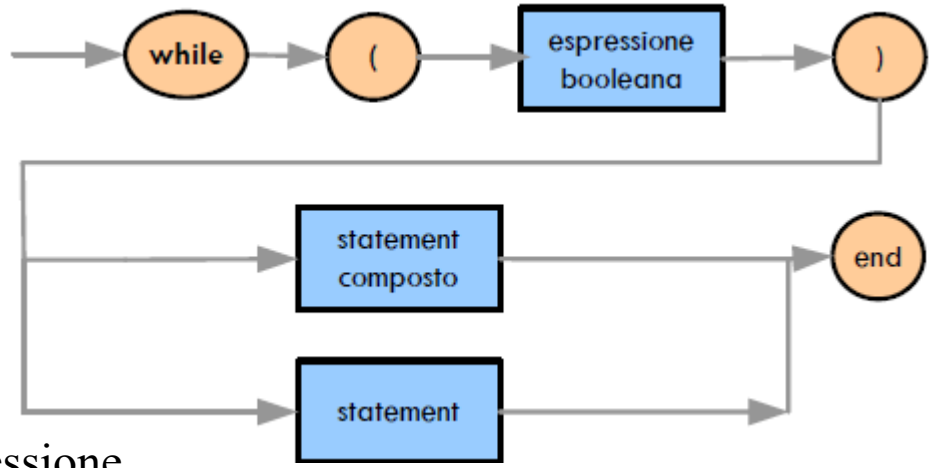
utilità dello switch per creare la corrispondenza tra i valori numerici delle cifre (variabile valcifra) e il relativo carattere ASCII (variabile carcifra).

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Il costrutto WHILE

- Nel liguaggio MATLAB ogni costrutto **while** va terminato con la parola chiave **end**.
- Sono esempi di **while** :

```
somma=0;
while (n<=20)
    somma=somma+vettore(n);
    n=n+1;
end
```



È possibile uscire da un ciclo WHILE in maniera incondizionata, cioè, in maniera indipendente dal valore di verità dell'espressione, attraverso la parola chiave BREAK

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Il costrutto FOR

- Il ***for*** è un enunciato iterativo enumerativo, detto anche ciclico, che deve essere usato ogni qualvolta il numero di ripetizioni è noto a priori. Si presenta nel seguente modo:

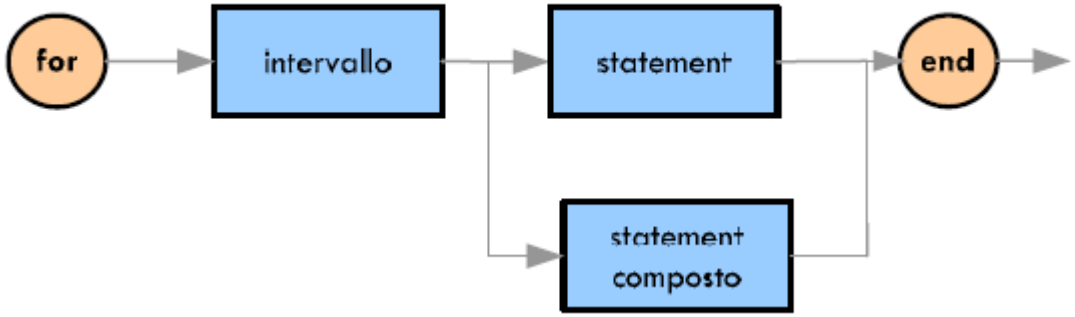
```
for t = Tin : Tfin
   S
end
```

- In MATLAB è possibile, così come avviene per la generazione degli, intervalli specificare anche il passo di iterazione, in tale caso il costrutto assume la seguente forma:

```
for t = Tin :p: Tfin
   S
end
```

- dove **p** rapprenda il passo di iterazione (se non viene specificato è di default posto al valore 1).
- E' possibile generare iterazioni in modo decrescente.

```
for t = 10 :-1:1
   S
end
```



UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***Lo scambio di valore***
- **Descrizione del problema**
- Scrivere una funzione che effettui lo scambio di valore tra due informazioni di un tipo *T*.
- Esempio:
- input: x=3, y=5
- output: x=5, y=3

```
z=x;
x=y;
y=z;
```

Descrizione delle funzioni

Per la realizzazione dell'algoritmo si utilizza una sola funzione **scambia** con le seguenti caratteristiche:

- Paramentri di input: [x,y] informazioni prima dello scambio
- Parametri di output: [x,y] informazioni dopo lo scambio
- Variabili locali: [z] informazione necessaria allo scambio

Implementazione

```
% FUNZIONE PER LO SCAMBIO DI VALORE
% La funzione [x,y]=scambia(x,y)
% permette di scambiare il valore delle variabili x ed y.
% Esempio:
%           x=3, y=5
%           [x,y]=scambia(x,y)
%           x=5, y=3

function [x,y]=scambia(x,y) % Intestazione funzione
% effettua scambia per mezzo di una terza variabile
    z=x;
    x=y;
    y=z;
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

Si nota come tutta la documentazione che precede l'intestazione nel codice può essere utilizzata come help della funzione.

```
» help scambia
 FUNZIONE PER LO SCAMBIO DI VALORE
 La funzione [x,y]=scambia(x,y)
 permette di scambiare il valore delle variabili x ed y.
 Esempio:
                    x=3, y=5
                    [x,y]=scambia(x,y)
                    x=5, y=3
» x=9;
» y=19;
» [x,y]=scambia(x,y)
  x = 19
  y = 9
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***L'inserimento in un vettore***
- **Descrizione del problema**
- Scrivere una funzione per l'inserimento dell'informazione *info* nella posizione *posiz* di un vettore di *n* elementi di un certo tipo. Esempio:

input: n=5, v= [10 50 20 40 35], info=66, posiz=3
output: n=6, v=[10 50 66 20 40 35]

- utilizzando la sintassi di Matlab, l'algoritmo assume la seguente forma :

```
for i=n:-1:posiz
v(i+1)=v(i);
end;
v(posiz)=info;
n=n+1;
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Per la realizzazione dell'algoritmo si utilizza una sola funzione **inserimento** con le seguenti caratteristiche:

- Parametri di Input: [v, n, info, posiz] rispettivamente vettore, riempimento del vettore, elemento da inserire e rispettiva posizione,
- Parametri di Output: [v, n] vettore e riempimento dopo l'inserimento
- Variabili locali: [i] contatore di ciclo

Implementazione

```
% FUNZIONE PER L'INSERIMENTO IN UN VETTORE
% La funzione [v,n]=inserimento(v,n,info,posiz)
% permette di inserire nel vettore v di n elementi l'elemento info
% alla posizione posiz.
% Esempio:
%              v=[1 2 4 5], n=4, info=3, posiz=3
%              [v,n]=inserimento(v,n,info,posiz)
%              v=[1 2 3 4 5], n=5

function [v,n]=inserimento(v,n,info,posiz)
% sposta gli elementi del vettore di un posto a destra a partire dall'ultimo fino
% a quello in posizione posiz
for i=n:-1:posiz
  v(i+1)=v(i);
end;
% inserisci il nuovo elemento nella posizione specificata
v(posiz)=info;
% aumenta di una unità il numero di elementi nel vettore
n=n+1;
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help inserimento
  FUNZIONE PER L'INSERIMENTO IN UN VETTORE
  La funzione [v,n]=inserimento(v,n,info,posiz)
  permette di inserire nel vettore v di n elementi l'elemento info
  alla posizione posiz.
  Esempio:
                v=[1 2 4 5], n=4, info=3, posiz=3
                [v,n]=inserimento(v,n,info,posiz)
                v=[1 2 3 4 5], n=5
» v=[10 50 20 40 35];
» n=5;
» info=66;
» posiz=3;
» [v,n]=inserimento(v,n,info,posiz)
  v =10    50    66    40    35
  n = 6
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***L'eliminazione in un vettore***
- **Descrizione del problema**
- Scrivere una funzione per l'eliminazione dell'elemento in una posizione *posiz* data di un vettore di *n* elementi di un certo tipo. Esempio:

input: n=5, v= [22 50 30 16 10], posiz=2
output: n=4, v=[22 30 16 10]

- Utilizzando la sintassi di Matlab, l'algoritmo assume la seguente forma:

```
for i=posiz+1:1:n
v(i-1)=v(i);
end;
n=n-1;
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

Per la realizzazione dell'algoritmo si utilizza una sola funzione **eliminazione** con le seguenti caratteristiche:

- Parametri di Input: [v, n, posiz] vettore di partenza, riempimento del vettore e posizione dell'elemento da eliminare

- Parametri di Output: [v,n] vettore e suo riempimento dopo l'eliminazione

- Variabili locali: i contatore di ciclo

Implementazione

```
% FUNZIONE PER L'ELIMINAZIONE IN UN VETTORE
% La funzione [v,n]=eliminazione(v,n,posiz)
% permette di eliminare dal vettore v di n elementi l'elemento
% alla posizione posiz.
% Esempio:
%              v=[1 2 4 3], n=4, posiz=3
%              [v,n]=eliminazione(v,n,posiz)
%              v=[1 2 3], n=3

function [v,n]=eliminazione(v,n,posiz)
% sposta gli elementi del vettore di un posto a sinistra a partire
% da quello successivo alla posizione posiz fino all'ultimo
for i=posiz+1:1:n
  v(i-1)=v(i);
end;
% diminuisce di una unità il numero di elementi nel vettore
n=n-1;
% aggiorna il vettore in maniera da considerare solo i primi n elementi
v=v(1:n);
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help eliminazione
FUNZIONE PER L'ELIMINAZIONE IN UN VETTORE
La funzione [v1,n]=eliminazione(v,n,posiz)
permette di eliminare dal vettore v di n elementi l'elemento
alla posizione posiz.
     Esempio:
               v=[1 2 4 3], n=4, posiz=3
               [v,n]=eliminazione(v,n,posiz)
               v=[1 2 3], n=3
» v=[22 50 30 16 10];
» n=5;
» posiz=2;
» [v,n]=eliminazione(v,n,posiz)
v = 22   30   16   10
n = 4
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***L'eliminazione di una colonna da una matrice***
- **Descrizione del problema**
- Scrivere una funzione per l'eliminazione di un data colonna di una matrice di *N* righe ed *M* colonne di un certo tipo. Esempio:

input: N=3, M=4, A= [10 22 33 50; 20 54 80 41; 30 10 23 31], col=2
output: N=3, M=3, A=[10 33 50; 20 80 41; 30 23 31]

- **Descrizione dell'algoritmo**
- L'eliminazione di una colonna in una matrice si effettua applicando l'algoritmo di eliminazione di un elemento da un vettore a tutte le righe della matrice. Utilizzando la sintassi di Matlab, l'algoritmo assume la seguente forma:

```
for j=col+1:1:M
for i=1:N
A(i,j-1)=A(i,j);
end;
end;
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

Per la realizzazione dell'algoritmo si utilizza una sola funzione **elimina_colonna** con le seguenti caratteristiche:

- Parametri di Input: [A,N,M,colonna], matrice, numero di righe, numero di colonne ed indice della colonna da eliminare

- Parametri di Output: [A,m] matrice e numero di colonne dopo l'eliminazione

- Variabili locali: i, j indici di riga e colonna

Implementazione

```
% FUNZIONE PER L'ELIMINAZIONE DI UNA COLONNA IN UNA MATRICE
% La funzione [A]=elimina_colonna(A,N,M,col)
% permette di eliminare la colonna col dalla matrice A di N righe e M colonne.
% Esempio:
%              A=[1 2 3; 4 5 6], N=2, M=3, col=3
%              [A]=elimina_colonna(A,N,M,col)
%              A=[1 2; 4 5]

function [A]=elimina_colonna(A,N,M,col)
% applica a tutte le righe della matrice l'algoritmo per l'eliminazione
% di un elemento nella posizione col assegnata
for j=col+1:1:M
  for i=1:N
      A(i,j-1)=A(i,j);
  end;
end;
% diminuisce di una unità il numero di colonne
M=M-1;
% aggiorna la matrice
A=A(1:N,1:M);
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help elimina_colonna
  FUNZIONE PER L'ELIMINAZIONE DI UNA COLONNA IN UNA MATRICE
  La funzione [A]=elimina_colonna(A,N,M,col)
  permette di eliminare la colonna col dalla matrice A di N righe e M colonne.
  Esempio:
                A=[1 2 3; 4 5 6], N=2, M=3, col=3
                [A]=elimina_colonna(A,N,M,col)
                A=[1 2; 4 5]
» A=[10 22 33 50; 20 54 80 41; 30 10 23 31];
» N=3;
» M=4;
» col=2;
» [A]=elimina_colonna(A,N,M,col)
A =
   10   33   50
   20   80   41
   30   23   31
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***L'eliminazione di una riga da una matrice***
- **Descrizione del problema**
- Scrivere una funzione per l'eliminazione di un data riga di una matrice di *N* righe ed *M* colonne di un certo tipo. Esempio:

  input: N=3, M=4, A= [10 22 33 50; 20 54 80 41; 30 10 23 31], riga=1
  output: N=2, M=4, A=[20 80 54 41; 30 10 23 31]
- L'eliminazione di una riga in una matrice si effettua applicando l'algoritmo di eliminazione di un elemento da un vettore a tutte le colonne della matrice.
- Utilizzando la sintassi di Matlab, l'algoritmo assume la seguente forma:

```
for i=riga+1:1:N
for j=1:M
A(i-1,j)=A(i,j);
end;
end;
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

Per la realizzazione dell'algoritmo si utilizza una sola funzione **elimina_riga** con le seguenti caratteristiche:

- Parametri di Input: [A, N, M, riga] matrice, numero di righe, numero di colonne ed indice della riga da eliminare

- Parametri di Output: [A,n] matrice e numero di righe dopo l'eliminazione

- Variabili locali: i, j indici di riga e colonna

Implementazione

```
% FUNZIONE PER L'ELIMINAZIONE DI UNA RIGA IN UNA MATRICE
% La funzione [A]=elimina_riga(A,N,M,riga)
% permette di eliminare la riga assegnata dalla matrice A di N righe e M colonne.
% Esempio:
%              A=[1 2 3; 4 5 6], N=2, M=3, riga=1
%              [A]=elimina_riga(A,N,M,riga)
%              A=[4 5 6]

function [A]=elimina_riga(A,N,M,riga)
% applica a tutte le colonne della matrice l'algoritmo per l'eliminazione
% di un elemento nella posizione riga assegnata
for i=riga+1:1:N
  for j=1:M
    A(i-1,j)=A(i,j);
  end;
end;
% diminuisce di una unità il numero di righe
N=N-1;
% aggiorna la matrice
A=A(1:N,1:M);
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help elimina_riga
  FUNZIONE PER L'ELIMINAZIONE DI UNA RIGA IN UNA MATRICE
  La funzione [A]=elimina_riga(A,N,M,riga)
  permette di eliminare la riga assegnata dalla matrice A di N righe e M colonne.
  Esempio:
                A=[1 2 3; 4 5 6], N=2, M=3, riga=1
                [A]=elimina_riga(A,N,M,riga)
                A=[4 5 6]
» A=[10 22 33 50; 20 54 80 41; 30 10 23 31];
» N=3;
» M=4;
» riga=1;
» [A]=elimina_riga(A,N,M,riga)
A =
   20   54   80   41
   30   10   23   31
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Scrivere una funzione per la ricerca dell'informazione i*nfo* in un vettore non ordinato avente *n* elementi di un certo tipo. In particolare la funzione deve produrre una indicazione sull'esistenza dell'informazione nel vettore e, nel caso esista, la posizione.
- Si fa l'ipotesi che nel caso in cui l'elemento non sia presente nel vettore la sua posizione è 0.
- Esempio:

  input: v= [9 5 6 8 7], info=8

  output: msg=elemento presente, posiz=4

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- **Descrizione dell'algoritmo**
- prendere un oggetto alla volta e confrontarlo con quello dato fino a quando non sene trova uno uguale ad esso o è stato visionato l'intero insieme.
- La ricerca di un'informazione in un vettore che contiene valori tra loro non ordinati procede in maniera sequenziale. Si comincia a confrontare il primo elemento con quello ricercato. Poi il secondo, poi il terzo e così via fin quando il confronto non risulta verificato. In questo modo si verifica l'assenza dell'elemento dopo averlo confrontato con tutti gli elementi del vettore. Nel caso contrario, quando lo s incontra, si ferma l'indagine facendo l'ipotesi che nel vettore non esistano valori tra di loro uguali.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Esempio. Sia assegnato il vettore: v=[9 5 6 8 7] (1 2 3 4 5) (tra le parentesi tonde sono indicate le posizioni degli elementi nel vettore) e sia 4 il valore da ricercare. Allora l'algoritmo procede iterativamente nel seguente modo:

```
trovato=0;
i=0;
posiz=0;
while (!trovato & i<n)
      if (v[i]==info)
          trovato=1;
          posiz=i;
          msg='elemento presente';
      end
      i++;
end
if trovato==0
   msg='elemento non presente';
end
```

passo (posizione nel vettore)	confronto corrente	trovato	fine vettore
1	4==9	no	no
2	4==5	no	no
3	4==6	no	no
4	4==8	no	no
5	4==9	no	si

e la ricerca termina con l'indicazione che 4 non è presente nel vettore.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Esempio. Sia assegnato il vettore: v=[9 5 6 8 7] (1 2 3 4 5) (tra le parentesi tonde sono indicate le posizioni degli elementi nel vettore) Cerchiamo quindi il valore 8:

```
trovato=0;
i=0;
posiz=0;
while (!trovato & i<n)
      if (v[i]==info)
         trovato=1;
         posiz=i;
        msg='elemento presente';
      end
      i++;
end
if trovato==0
   msg='elemento non presente';
end
```

passo (posizione nel vettore)	confronto corrente	trovato	fine vettore
1	8==9	no	no
2	8==5	no	no
3	8==6	no	no
4	8==8	si	no

e la ricerca termina con l'indicazione che il valore 8 è presente all'interno del vettore nella posizione 4.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Descrizione delle funzioni

Per la realizzazione dell'algoritmo si utilizza una sola funzione **ricercaseq** con le seguenti caratteristiche:

- Parametri di Input: [v, info] vettore d'ingresso ed informazione da ricercare

- Parametri di Output: [msg, posiz] variabile contenete il messaggio sull'esito della ricerca e posizione dell'elemento all'interno del vettore

- Variabili locali: [i, trovato, n] contatore di ciclo, variabile binaria che indica se il confronto ha avuto successo o meno e riempimento del vettore

- Funzioni richiamate: **length** per il calcolo del riempimento del vettore.

Implementazione

```
% FUNZIONE PER LA RICERCA SEQUENZIALE DI UN ELEMENTO IN UN VETTORE
% La funzione [msg,posiz]=ricercaseq(v,info)
% permette di cercare l'elemento info nel vettore v.
% Se la ricerca ha successo viene restituita la posizione posiz
% dell'elemento nel vettore.
% msg è una variabile che segnala l'esito della ricerca.
% Esempio:
%              v=[1 2 8 9 5 6], info=8
%              [msg,posiz]=ricercaseq(v,info)
%              msg= 'elemento presente', posiz=3

function [msg,posiz]=ricercaseq(v,info)
% inizializzazione variabili locali
% variabile binaria che indica il successo del confronto
% (0 indica che l'elemento non è stato ancora trovato,
% 1 il successo del confronto)
trovato=0;
% riempimento del vettore
n=length(v);
% cerca iterativamente l'elemento nel vettore confrontandolo con i suoi elementi
% dal primo all'ultimo
i=0;
posiz=0;
while (~trovato & i<n)
    if (v[i]==info)
        trovato=1;
        posiz=i;
       msg='elemento presente';
    end
    i++;
end
if trovato==0
  msg='elemento non presente';
end
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB ed il confronto con la funzione **find** predefinita nel linguaggio.

```
» help ricercaseq
  FUNZIONE PER LA RICERCA SEQUENZIALE DI UN ELEMENTO IN UN VETTORE
  La funzione [msg,posiz]=ricercaseq(v,info)
  permette di cercare l'elemento info nel vettore v.
  Se la ricerca ha successo viene restituita la posizione posiz
  dell'elemento nel vettore.
  msg è una variabile che segnala l'esito della ricerca.
  Esempio:
                v=[1 2 8 9 5 6], info=8
                [msg,posiz]=ricercaseq(v,info)
                msg= 'elemento presente', posiz=3
» v=[9 5 6 8 7];
» info=8;
» [msg,posiz]=ricercaseq(v,info)

msg = elemento presente
posiz = 4

» posiz=find(v==8)
posiz =4
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***La ricerca binaria***
- **Descrizione del problema**
- Scrivere una funzione per la ricerca dell'informazione i*nfo* in un vettore ordinato in senso crescente avente *n* elementi di un certo tipo. In particolare la funzione deve produrre una indicazione sull'esistenza dell'informazione nel vettore e, nel caso esista, la posizione. Si fa l'ipotesi che nel caso in cui l'elemento non sia presente nel vettore la sua posizione è 0. Esempio:

  input: v=[10 12 15 33 40 55], info=33

  output: msg=elemento presente, posiz=4

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

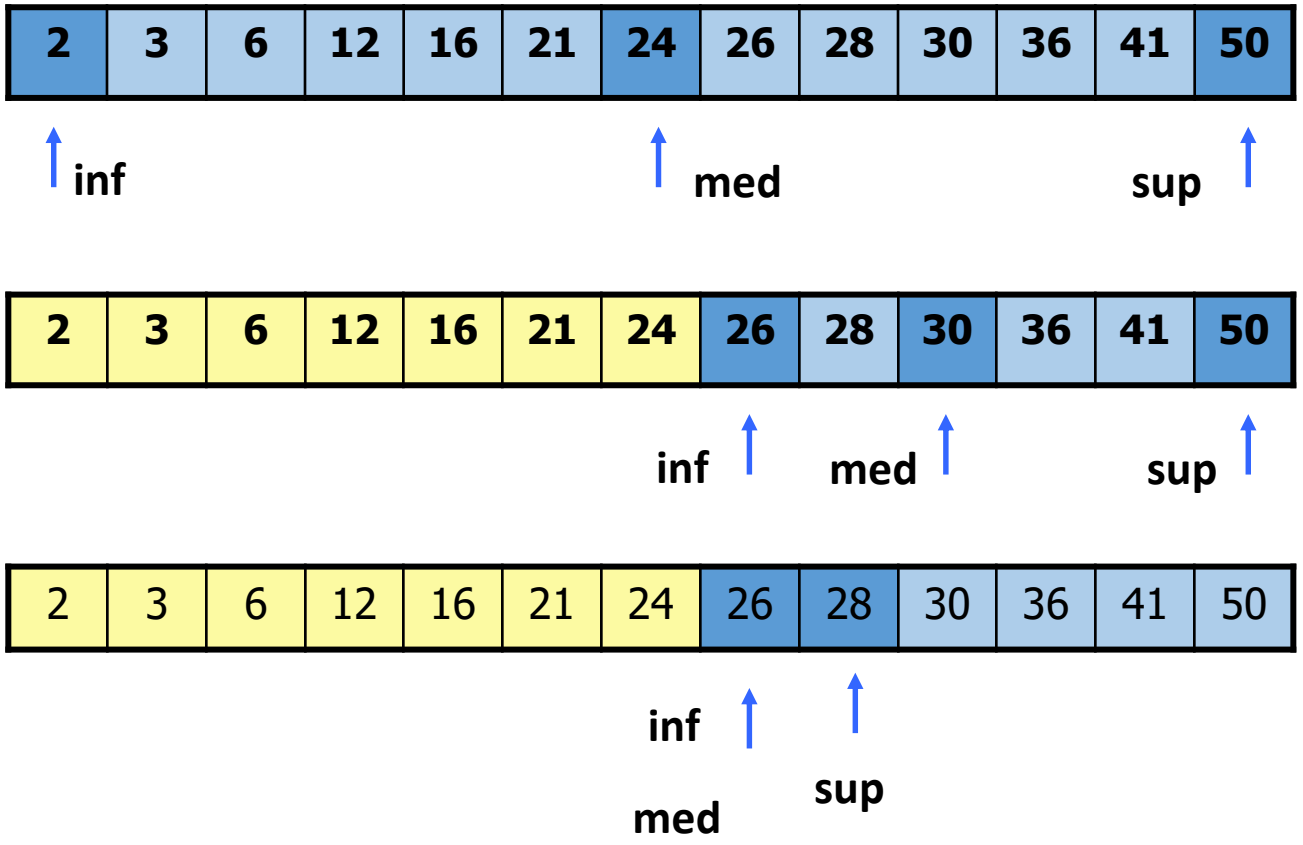
- **Descrizione dell'algoritmo**
- La ricerca di una informazione in un elenco si può effettuare come visto nel caso precedente confrontando uno dopo l'altro i valori dell'elenco con quello cercato fino a quando non si trova un elemento uguale o non si è analizzato l'intero elenco.
- Il metodo sequenziale richiede nel caso peggiore n confronti, dove n è il riempimento del vettore. Se l'elenco è ordinato, si ricorre allora ad una ricerca che ad ogni passo riduce l'insieme in cui cercare mediante un'opportuna tecnica di dimezzamento. In questo modo il caso peggiore richiede al più log2(n) confronti. Il metodo noto col nome di ricerca binaria si basa sui seguenti passi:
  1. determinazione del punto medio dell'insieme in cui cercare;
  2. confronto dell'elemento in questa posizione con quello da cercare;
  3. individuazione dell'insieme in cui continuare la ricerca se il passo 2 non ha successo; esso risulta per vettore ordinato in senso crescente: il sottoinsieme degli elementi in posizioni successive al punto medio se il valore da cercare è maggiore di quello del punto medio, altrimenti quello caratterizzato da posizioni inferiori;
  4. ripetizioni dei passi precedenti finché il passo 2 non risulti verificato o non sia possibile fissare un sottoinsieme in cui continuare la ricerca.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

1. determinazione del punto medio dell'insieme in cui cercare;
2. confronto dell'elemento in questa posizione con quello da cercare;
3. individuazione dell'insieme in cui continuare la ricerca se il passo 2 non ha successo; esso risulta per vettore ordinato in senso crescente: il sottoinsieme degli elementi in posizioni successive al punto medio se il valore da cercare è maggiore di quello del punto medio, altrimenti quello caratterizzato da posizioni inferiori;
4. ripetizioni dei passi precedenti finché il passo 2 non risulti verificato o non sia possibile fissare un sottoinsieme in cui continuare la ricerca.

Esempio numerico : Valore cercato 26



UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Sia assegnato il vettore: v=[15 22 29 36 50 55] e sia 21 il valore da ricercare. Allora l'algoritmo procede iterativamente nel seguente modo:

Medio=floor(1+6)/2=3
Medio=floor(1+2)/2=1
Medio=floor(2+2)/2=2

passo	punto medio	confronto corrente	prossimo sottoinsieme di ricerca	fine	trovato
1	29	21<29	[15, 22]	no	no
2	15	21>15	[22]	no	no
3	22	21<22	[22, 15]	si	no

- A questo punto la ricerca termina perché non esiste un sottoinsieme (l'estremo inferiore è maggiore del superiore) in cui continuare la ricerca, con l'indicazione che 21 non è presente nel vettore.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Sia assegnato il vettore: v=[15 22 29 36 50 55] (1 2 3 4 5 6) (tra le parentesi tonde sono indicate le posizioni degli elementi nel vettore) e sia 50 il valore da ricercare. Allora l'algoritmo procede iterativamente nel seguente modo:

Medio=floor(1+6)/2=3
Medio=floor(4+6)/2=5

passo	punto medio	confronto corrente	prossimo sottoinsieme di ricerca	fine	trovato
1	29	50>29	[36, 50, 55]	no	no
2	50	50==50	Vuoto	si	si

- la ricerca termina con l'indicazione che il valore 50 è presente all'interno del vettore nella posizione 5.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Utilizzando la sintassi di Matlab, l'algoritmo assume la seguente forma:

```
posiz=0;
trovato=0;
ei=1;
es=n;
while (!trovato & ei<=es+1)
  medio=floor((ei+es)/2);
  if (info==v(medio))
    trovato=1;
  end;
  if (info<v(medio))
    es=medio-1;
  else ei=medio+1;
  end;
end;
```

floor(A) arrotonda A all'intero più vicino MINORE O UGUALE ad A

ei=estremo inferiore

es=estremo superiore

Gli algoritmi di base in MATLAB

Implementazione

```
% FUNZIONE PER LA RICERCA BINARIA DI UN ELEMENTO IN UN VETTORE
% La funzione [msg,posiz]=ricercabin(v,info)
% permette di cercare l'elemento info nel vettore v.
% Se la ricerca ha successo viene restituita la posizione posiz
% dell'elemento nel vettore.
% msg è una variabile che segnala l'esito della ricerca.
% Esempio:
%               v=[1 2 8 9 5 6], info=8
%               [msg,posiz]=ricercabin(v,info)
%               msg= 'elemento presente', posiz=3
% Nota: la ricerca è applicabile solo se il vettore d'ingresso è ordinato
function [msg,posiz]=ricercabin(v,info)
% inizializza le varibili locali
% posizione dell'elemento all'interno del vettore
% (se l'elemento non viene trovato la sua posizione è nulla)
posiz=0;
% variabile binaria che indica il successo del confronto
% (0 indica che l'elemento non è stato ancora trovato,
% 1 il successo del confronto)
trovato=0;
% estremi superiore e inferiore dei sottoinsieme di ricerca
% (inizialmente il sottoinsime di ricerca coincide con l'intero vettore)
ei=1;
es=length(v);
% inizio ricerca
% la condizione di continuazione della ricerca è che l'estremo inferiore del
% sottoinsieme corrente di ricerca sia minore di quello superiore e che l'elemento
% non sia stato ancora trovato
while (!trovato & ei<es+1)
% calcolo del punto medio
  medio=floor((ei+es)/2);
% se l'elmento viene trovato pone ad 1 la variabile trovato
% e arresta la ricerca
  if (info==v(medio))
    trovato=1;
  end;
% se l'elmento non viene trovato aggiorna gli estremi del sottoinsieme di ricerca
% a seconda che l'info da ricercare sia minore o maggiore del punto medio
  if (info<v(medio))
      es=medio-1;
  else ei=medio+1;
  end;
end;
% a secona dell'esito della ricerca aggiorna la variabile msg
if (trovato==1)
  msg='elemento presente';
  posiz=medio;
else
  msg='elemento non presente';
end;
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help ricercabin
  FUNZIONE PER LA RICERCA BINARIA DI UN ELEMENTO IN UN VETTORE
  La funzione [msg,posiz]=ricercabin(v,info)
  permette di cercare l'elemento info nel vettore v.
  Se la ricerca ha successo viene restituita la posizione posiz
  dell'elemento nel vettore.
  msg è una variabile che segnala l'esito della ricerca.
  Esempio:
                v=[1 2 8 9 5 6], info=8
                [msg,posiz]=ricercabin(v,info)
                msg= 'elemento presente', posiz=3
  Nota: la ricerca è applicabile solo se il vettore d'ingresso è ordinato
» v=[15 22 29 36 50 55];
» info=21;
» [msg,posiz]=ricercabin(v,info)
msg = elemento non presente
posiz = 0
» info=50;
» [msg,posiz]=ricercabin(v,info)
msg = elemento presente
posiz = 5
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***Il valore massimo in un vettore***
- **Descrizione del problema**

  Scrivere una funzione che determini il massimo di un vettore formato da *n* elementi di un certo tipo. Esempio:

  input: v=[10 55 20 11 30]

  output: max=55

  Esempio. Sia assegnato il vettore:

  v=[3 5 6 8 7]

  Allora l'algoritmo procede iterativamente nel seguente modo:

```
max=v(1);
for i=2:n
   if (v(i)>max)
      max=v(i);
   end
end
```

massimo	posizione vettore	confronto (max<elemento corrente del vettore)	esito confronto	aggiornamento massimo
max=3	2	3<5	Vero	si
max=5	3	5<6	Vero	si
max=6	4	6<8	vero	si
max=8	5	8<7	falso	no

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

Per la realizzazione dell'algoritmo si utilizza una sola funzione **massimo** con le seguenti caratteristiche:

- Parametri di Input: [v] vettore d'ingresso

- Parametri di Output: [max] variabile contenete il massimo del vettore

- Variabili locali: [i, n] contatore di ciclo e riempimento del vettore

- Funzioni richiamate: **length** per il calcolo del riempimento del vettore

Implementazione

```
% FUNZIONE PER IL CALCOLO DEL MASSIMO DI UN VETTORE
% La funzione [max]=massimo(v)
% permette di calcolare l'elemento massimo max del vettore v.
% Esempio:
%              v=[2 4 8 7 5]
%              [max]=massimo(v)
%              max=8
function [max]=massimo(v)
% calcolo riempimento vettore
n=length(v);
% inizializza il massimo al primo elemento del vettore
max=v(1);
% iterativamente scorre il vettore e verifica se ci sia qualche
% elemento maggiore del massimo
for i=2:n
  % se l'elmento corrente è maggiore del massimo aggiorna il massimo
  if (v(i)>max)
    max=v(i);
  end
end
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help massimo
  FUNZIONE PER IL CALCOLO DELA MASSIMO DI UN VETTORE
  La funzione [max]=massimo(v)
  permette di calcolare l'elemento massimo max del vettore v.
  Esempio:
                v=[2 4 8 7 5]
                [max]=massimo(v)
                max=8
» v=[3 5 6 8 7];
» [max]=massimo(v)
max = 8
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***La posizione del valore minimo di un vettore***
- **Descrizione del problema**
- Scrivere una funzione che determini la posizione del valore minimo di un vettore formato da *n* elementi di un certo tipo. Esempio:

  input: v=[10 55 20 11 30]

  output: posiz=1
- **Descrizione dell'algoritmo**
- L'individuazione della posizione del minimo in un insieme si effettua osservando uno dopo l'altro gli elementi che lo compongono e memorizzando di volta in volta la posizione dell'elemento avente il valore più piccolo. In particolare, si inizia facendo l'ipotesi che la posizione del minimo sia quella del primo elemento del vettore. Successivamente si confronta l'elemento nella presunta posizione di minimo con i restanti valori del vettore. Ogni volta che si incontra un valore più piccolo, si effettua l'aggiornamento della posizione in modo che alla fine dei confronti si abbia la posizione del minimo. Si fa l'ipotesi che nel vettore non esistano elementi uguali.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Sia assegnato il vettore:
- v=[9 5 6 3 7]
- Allora l'algoritmo procede iterativamente nel seguente modo:

```
posix=1;
for i=2:n
   if (v(posix)>v(i))
      posix=i;
   end
end
```

posiz. minimo	posizione vettore	confronto (min>elemento corrente del vettore)	esito confronto	aggiornamento posizione minimo
posiz=1	2	9>5	vero	si
posiz=2	3	5>6	falso	no
posiz=2	4	5>3	vero	si
posiz=4	5	3>7	falso	no

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

Per la realizzazione dell'algoritmo si utilizza una sola funzione **posmin** con le seguenti caratteristiche:

- Parametri di Input: [v] vettore d'ingresso- Parametri di Output: [posiz] variabile contenete il massimo del vettore

- Variabili locali: [i, n] contatore di ciclo e riempimento del vettore

- Funzioni richiamate: **length** per il calcolo del riempimento del vettore

Implementazione

```
% FUNZIONE PER IL CALCOLO DELLA POSIZIONE DEL MINIMO DI UN VETTORE
% La funzione [posiz]=posmin(v)
% permette di calcolare la posizione dell'elemento minimo posiz del vettore v.
% Esempio:
%               v=[2 4 8 7 5]
%               [posiz]=posmin(v)
%               posiz=1
function [posiz]=posmin(v)
% calcolo riempimento vettore
n=length(v);
% inizializza la posizione del minimo alla prima nel vettore
posiz=1;
% iterativamente scorre il vettore e verifica se ci sia qualche elemento
% minore di quello nella presunta posizione di minimo
for i=2:n
  % se l'elmento corrente è minore aggiorna la posizione
  if (v(posiz)>v(i))
    posiz=i;
  end
end
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help posmin
  FUNZIONE PER IL CALCOLO DELLA POSIZIONE DEL MINIMO DI UN VETTORE
  La funzione [posiz]=posmin(v)
  permette di calcolare la posizione dell'elemento minimo posiz del vettore v.
  Esempio:
                v=[2 4 8 7 5]
                [posiz]=posmin(v)
                posiz=1
» v=[9 5 6 3 7];
» [posiz]=posmin(v)
posiz = 4
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***Minimo e massimo in una matrice***
- **Descrizione del problema**
- Scrivere una funzione che determini il valore minimo e massimo di una matrice formata da *n x m* elementi di un certo tipo. Esempio:

  input: A=[1 2 3 4; 8 7 6 5]

  output: max=8, min=1
- L'individuazione del minimo e del massimo in un insieme si effettua osservando uno dopo l'altro gli elementi che lo compongono e memorizzando di volta in volta la posizione i valori dell'elemento avente il valore più piccolo e più grande.
- In particolare, si inizia facendo l'ipotesi che il primo elemento della matrice sia contemporaneamente il massimo ed il minimo e successivamente lo si confronta con i restanti valori della matrice. Ogni volta che si incontra un valore più piccolo o più grande, si effettua l'aggiornamento del minimo e del massimo.

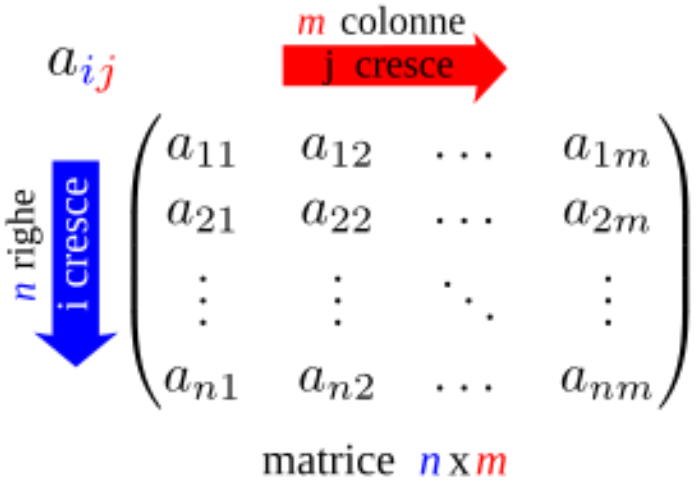
UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

-Funzioni richiamate: size per il calcolo del numero di righe di colonne della matrice

-[n,m]=size(A)

RIGHE COLONNE



A

a[1][1]	a[1][2]	a[1][3]	a[1][4]	a[1][5]
a[2][1]	a[2][2]	a[2][3]	a[2][4]	a[2][5]
a[3][1]	a[3][2]	a[3][3]	a[3][4]	a[3][5]

```
[n,m]=size(A);
% iterativamente scorre la matrice e verifica se ci sia qualche
% elemento minore o maggiore
% di quello minimo o massimo presunto
for i=1:n
for j=1:m
% se l'elemento corrente è maggiore del massimo aggiorna il massimo
% se l'elemento corrente è minore del minimo aggiorna il minimo
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Per la realizzazione dell'algoritmo si utilizza una sola funzione **minmax** con le seguenti caratteristiche:

- Parametri di Input: [A] matrice d'ingresso
- Parametri di Output: [min, max] variabili contenete il minimo e massimo della matrice
- Variabili locali: [i,j,n,m] indice di riga, indice di colonna, numero di righe e numero di colonne della matrice
- Funzioni richiamate: **size** per il calcolo del numero di righe di colonne della matrice
- [n,m]=size(A)

Implementazione

```
% FUNZIONE PER IL CALCOLO DEL MINIMO e MASSIMO DI UNA MATRICE
% La funzione [min,max]=minmax(A)
% permette di calcolare il minimo e massimo (min e max) di una matrice A.
% Esempio:  A=[1 2;3 4]
%                                  [min,max]=minmax(A)
%                                  min=1, max=4

function [min,max]=minmax(A)
% calcolo dimensioni matrice
[n,m]=size(A);
% inizializza il massimo e il minimo al primo elmento della matrice
min=A(1,1);
max=min;
% iterativamente scorre la matrice e verifica se ci sia qualche
% elemento minore o maggiore
% di quello minimo o massimo presunto
for i=1:n
  for j=1:m
% se l'elmento corrente è maggiore del massimo aggiorna il massimo
    if (A(i,j)>max)
            max=A(i,j);
    else
% se l'elmento corrente è minore del minimo aggiorna il minimo
      if (A(i,j)<min)
            min=A(i,j);
      end
    end
  end
end
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help minmax
  FUNZIONE PER IL CALCOLO DEL MINIMO e MASSIMO DI UNA MATRICE
  La funzione [min,max]=minmax(A)
  permette di calcolare il minimo e massimo (min e max) di una matrice A.
  Esempio:
                A=[1 2;3 4]
                [min,max]=minmax(A)
                min=1, max=4

» A=[1 2 3 4;8 7 6 5];
» [min,max]=minmax(A)
min =  1, max =  8
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***Ordinamento di un vettore col metodo della selezione***
- **Descrizione del problema**
- Scrivere una funzione per l'ordinamento in senso crescente di un vettore di *n* elementi di un certo tipo. Esempio:

  input: v=[30 10 50 12 22]

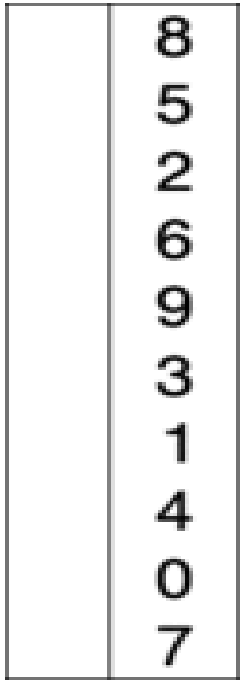
  output: v=[10 12 22 30 50]
- **Descrizione dell'algoritmo**
- Ordinare un vettore in senso crescente significa imporre che scelti due qualsiasi indici i e j, tali che i<j risulti v(i)<v(j). Un meccanismo di ordinamento consiste nel dividere l'insieme da ordinare in due parti: una ordinata ed una disordinata. Si procede allora estraendo un elemento alla volta dall'insieme disordinato e accodandolo a quello ordinato in modo che l'ordinamento non venga alterato. All'inizio l'insieme ordinato è vuoto e l'algoritmo termina quando contiene tutti gli elementi del vettore. Vi sono vari modi di estrarre l'elemento dall'insieme disordinato. Quello che di seguito presentiamo seleziona ad ogni passo l'elemento da accodare. In particolare seleziona il minimo dall'insieme disordinato e lo accoda all'insieme ordinato. In tal modo si viene ad ogni passo a scegliere il valore più grandi di quelli che lo precedono e contemporaneamente più piccolo di quelli che lo seguono.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

IDEA BASE

cerca il minimo dell'array di partenza, posizionalo nella prima posizione dell'array ordinato. Quindi scegli il minimo tra elementi rimanenti, posizionalo nella seconda posizione dell'array ordinato, … e così fino ad esaurimento del vettore



Animazione visibile su: https://it.wikipedia.org/wiki/Selection_sort

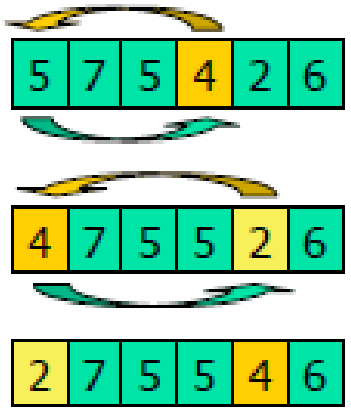
UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si confronta il primo elemento del vettore con tutti gli altri, man mano che si trova un valore minore del primo, lo si scambia con il primo. Dopo questa operazione il valore minimo è nella prima posizione (pos. 0) del vettore

è il successivo a 0

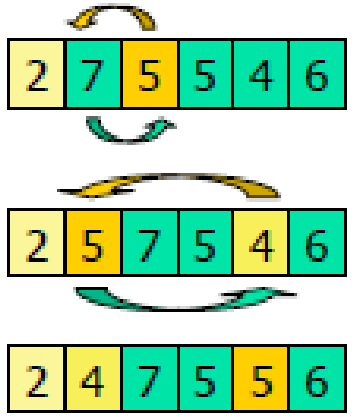
```
for (j=  1; j<N; j++)
  if (v[j] < v[0])
  {
    tmp = v[j];
    v[j] = v[0];
    v[0] = tmp;
  }
```



- Se si ripete lo stesso identico procedimento per tutti gli elementi a partire dal secondo, si determina il secondo valore più piccolo e lo si colloca al secondo posto

è il successivo a 1

```
for (j=  2; j<N; j++)
  if (v[j] < v[1])
  {
    tmp = v[j];
    v[j] = v[1];
    v[1] = tmp;
  }
```



SEMPLIFICATO

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Se si ripete lo stesso identico procedimento per tutti gli elementi a partire dall'i-esimo, si determina l'i-esimo valore più piccolo e lo si colloca all'i-esimo posto

è il successivo a **i**

```
for (j=i+1; j<N; j++)
  if (v[j] < v[i])
  {
     tmp = v[j];
     v[j] = v[i];
     v[i] = tmp;
  }
```

- Se si ripete questo procedimento per tutti i valori di **i** da 0 fino al penultimo (l'ultimo va a posto da sé) si ottiene l'ordinamento in senso crescente di tutto il vettore

```
for (i=0; i<N-1; i++)
  for (j=i+1; j<N; j++)
    if (v[j] < v[i])
    {
       tmp = v[j];
       v[j] = v[i];
       v[i] = tmp;
    }
```

SEMPLIFICATO

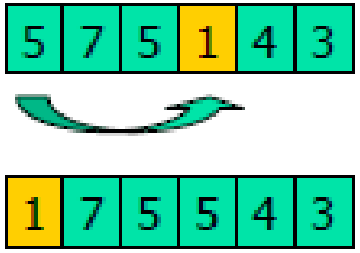
UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Il primo passo è quello di individuare il minimo tra gli elementi del vettore e scambiarlo con quello nella prima posizione, il primo valore è ora al posto giusto

```
jmin = 0;
for (j=  1; j<N; j++)
  if (v[j] < v[jmin])
    jmin = j;
tmp = v[jmin];
v[jmin] = v[0];
v[0] = tmp;
```

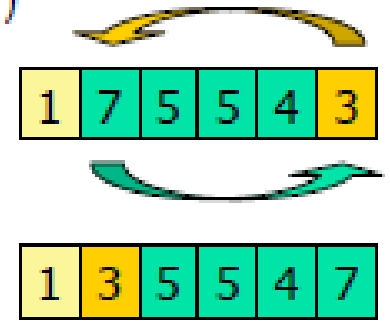
è il successivo a 0



- Se si ripete lo stesso identico procedimento per tutti gli elementi a partire dal secondo, si determina il secondo valore più piccolo e lo si colloca al secondo posto

```
jmin = 1;
for (j=  2; j<N; j++)
  if (v[j] < v[jmin])
    jmin = j;
tmp = v[jmin];
v[jmin] = v[1];
v[1] = tmp;
```

è il successivo a 1



UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Se si ripete lo stesso identico procedimento per tutti gli elementi a partire dall'i-esimo, si determina l'i-esimo valore più piccolo e lo si colloca all'i-esimo posto

è il successivo a **i**

```
jmin = i;
for (j=i+1; j<N; j++)
  if (v[j] < v[jmin])
    jmin = j;
tmp = v[jmin];
v[jmin] = v[i];
v[i] = tmp;
```

- Se si ripete questo procedimento per tutti i valori di **i** da 0 fino al penultimo (l'ultimo va a posto da sé) si ottiene l'ordinamento in senso crescente di tutto il vettore

```
for (i=0; i<N-1; i++)
{ jmin = i;
  for (j=i+1; j<N; j++)
    if (v[j] < v[jmin])
      jmin = j;
  tmp = v[jmin];
  v[jmin] = v[i];
  v[i] = tmp;
}
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- esempio. Sia assegnato il vettore: v=[9 2 1 4 7]
- Allora l'algoritmo procede iterativamente nel seguente modo:

Parte Ordinata	Parte Disordinata	posizione minimo (valore minimo)	accodamento	nuovo intervallo di ricerca del minimo
Vuota	(1 2 3 4 5)	3(1)	[1]	[9 2 4 7]
(1)	(2 3 4 5)	2(2)	[1 2]	[9 4 7]
(1 2)	(3 4 5)	4(4)	[1 2 4]	[9 7]
(1 2 3)	(4 5)	5(7)	[1 2 4 7]	[9]
(1 2 3 4)	(5)	5(9)	[1 2 4 7 9]	vuoto

- Si noti che l'algoritmo termina quando l'insieme disordinato si riduce ad un unico elemento. Inoltre poiché si usa lo stesso vettore per la parte ordinata e disordinata, l'accodamento viene effettuato scambiando di posto il minimo ed il valore che occupa la posizione di accodamento.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

```
for i=1:n-1
   imin=i;
   for j=i+1:n
      if (v(j)<v(imin))
         imin=j;
      end
   end
   temp=v(i);
   v(i)=v(imin);
   v(imin)=temp;
end
```

o il vettore: v=[9 2 1 4 7]

ede iterativamente nel seguente modo:

Parte Ordinata	Parte Disordinata	posizione minimo (valore minimo)	accodamento	nuovo intervallo di ricerca del minimo
Vuota	(1 2 3 4 5)	3(1)	[1]	[9 2 4 7]
(1)	(2 3 4 5)	2(2)	[1 2]	[9 4 7]
(1 2)	(3 4 5)	4(4)	[1 2 4]	[9 7]
(1 2 3)	(4 5)	5(7)	[1 2 4 7]	[9]
(1 2 3 4)	(5)	5(9)	[1 2 4 7 9]	vuoto

termina quando l'insieme disordinato si riduce ad un
e poiché si usa lo stesso vettore per la parte ordinata
mento viene effettuato scambiando di posto il
e occupa la posizione di accodamento.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

Per la realizzazione dell'algoritmo si utilizza una sola funzione **ordina** con le seguenti caratteristiche:

- Parametri di Input: [v] vettore d'ingresso

- Parametri di Output: [v] vettore ordinato

- Variabili locali: [i,n,imin,temp] contatore di ciclo, riempimento del vettore, indice del minimo corrente dell'insieme disordinato e variabile d'appoggio per effettuare lo scambio ai fini dell' accodamento

- Funzioni richiamate: **length** per il calcolo del riempimento del vettore

Implementazione

```
% FUNZIONE PER L'ORDINAMENTO DI UN VETTORE
% La funzione [v]=selectsort(v)
% permette di ordinare in senso crescente il vettore v.
% Esempio:          v=[4 1 3 2 5]
%                                [v]=selectsort(v)
%                                v=[1 2 3 4 5]

function [v]=selectsort(v)

% calcolo riempimento vettore
n=length(v);
% doppio ciclo per l'ordinamento
for i=1:n-1
  imin=i;
  % determina il minimo per la parte disordinata
  for j=i+1:n
    if (v(j)<v(imin))
      imin=j;
    end
  end
  % effettua l'accodamento scambiando di posto l'elemento minimo e il valore
  % che occupa la posizione di accodamento
  temp=v(i);
  v(i)=v(imin);
  v(imin)=temp;
end
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help selectsort
  FUNZIONE PER L'ORDINAMENTO DI UN VETTORE
  La funzione [v]=selectsort(v)
  permette di ordinare in senso crescente il vettore v.
  Esempio:
                  v=[4 1 3 2 5]
                  [v]=selectsort(v)
                  v=[1 2 3 4 5]
» v=[9 2 1 4 7];
» [v]=selectsort(v)
v = 1   2   4   7   9
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- ***Ordinamento di un vettore col metodo del gorgogliamento***
- **Descrizione del problema**
- Scrivere una funzione per l'ordinamento in senso crescente di un vettore di n elementi di un certo tipo.

Esempio: **input: v=[30 10 50 12 22]**
output: v=[10 12 22 30 50]

- **Descrizione dell'algoritmo**
- Ordinare un vettore in senso crescente significa imporre che scelti due qualsiasi indici i e j, tali che i<j risulti v(i)<v(j). Un meccanismo di ordinamento consiste nel dividere l'insieme da ordinare in due parti: una ordinata ed una disordinata. Si procede allora estraendo un elemento alla volta dall'insieme disordinato e accodandolo a quello ordinato in modo che l'ordinamento non venga alterato. All'inizio l'insieme ordinato è vuoto e l'algoritmo termina quando contiene tutti gli elementi del vettore. Vi sono vari modi di estrarre l'elemento dall'insieme disordinato. Quello che di seguito presentiamo fa in modo che il minimo dell'insieme disordinato gorgogli in prima posizione così come una bolla d'aria sale dal fondo alla superficie. In questo modo si viene ad ogni passo a scegliere il valore più grandi di quelli che lo precedono e contemporaneamente più piccolo di quelli che lo seguono. Il gorgogliamento viene eseguito confrontando a due a due gli elementi dell'insieme disordinato ed effettuando lo scambio di posizione quando le condizioni di ordinamento non sono rispettate. Se dopo uno scorrimento non si eseguono scambi si deduce che il vettore è ordinato e l'algoritmo può terminare.

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

IDEA BASE

Se due elementi vicini non sono in ordine allora scambiali.

Analizza tutte le coppie e scambiale se necessario.

Se esplori tutto il vettore senza fare scambi allora è in ordine.

6 5 3 1 8 7 2 4

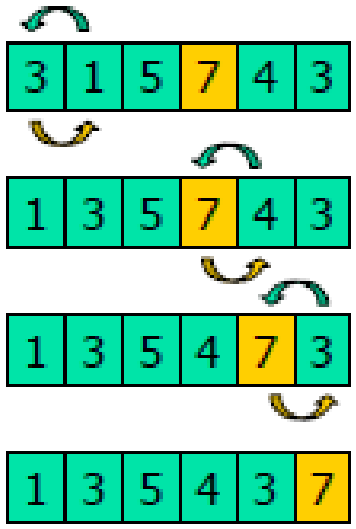
Animazione visibile su: https://it.wikipedia.org/wiki/Bubble_sort

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Se si scorrono tutti gli elementi di un vettore e ogni volta che si trovano due valori ADIACENTI non in ordine (il più piccolo dei 2 a destra del più grande) li si scambia: il più grande di tutti risale a destra

```
for (j=0; j<N-1; j++)
  if (v[j] > v[j+1])
  {
    tmp = v[j];
    v[j] = v[j+1];
    v[j+1] = tmp;
  }
```



- Ripetendo N-1 volte questa operazione, tutti i valori risalgono verso destra fino ad occupare la posizione corretta e quindi vengono ordinati in senso crescente

```
for (i=0  ; i<N-1; i++)
  for (j=0; j<N-1; j++)
    if (v[j] > v[j+1])
    {
      tmp = v[j];
      v[j] = v[j+1];
      v[j+1] = tmp;
    }
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Inefficienza: gli ultimi valori vengono in ogni caso confrontati, anche quando sono già stati collocati; per evitare perdita di tempo in questi controlli inutili si ferma prima della fine il ciclo interno, sfruttando il ciclo esterno

```
for (i=N-1; i>0  ; i--)
  for (j=0; j<i  ; j++)
    if (v[j] > v[j+1])
    {
      tmp = v[j];
      v[j] = v[j+1];
      v[j+1] = tmp;
    }
```

- Inefficienza: se pochi valori sono fuori posto e l'ordinamento si ottiene prima delle N-1 passate, i cicli continuano ad essere eseguiti. Lo si fa terminare se non ci sono stati scambi

```
scambi = SI;
for (i=N-1; i>0 && scambi ;i--)
{ scambi = NO;
  for (j=0; j<i  ; j++)
    if (v[j] > v[j+1])
    { scambi = SI;
      tmp = v[j];
      v[j] = v[j+1];
      v[j+1] = tmp; }
}
```

continua solo se ci sono stati scambi

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- esempio. Sia assegnato il vettore: v=[9 2 1 4 7]. Allora l'algoritmo procede iterativamente nel seguente modo:

l'algoritmo termina quando l'insieme disordinato si riduce ad un unico elemento. Inoltre se si applica l'algoritmo ad un vettore già ordinato si ha che non vengono effettuati scambi e quindi questa condizione può essere utile per arrestare l'algoritmo stesso.

Parte Ordinata	Parte Disordinata	Confronto	gorgogliamento
Vuota	(1 2 3 4 5)	4<7(4,5) 1<4 (3,4) 2>1(2,3) -> scambio 9>1(1,2) -> scambio	[9 2 1 4 7] [9 2 1 4 7] [9 1 2 4 7] [1] [9 2 4 7]
(1)	(2 3 4 5)	4<7(4,5) 2<4 (3,4) 9>2 (2,3) -> scambio	[1] [9 2 4 7] [1] [9 2 4 7] [1 2] [9 4 7]
(1 2)	(3 4 5)	4<7(4,5) 9>4 (3,4) -> scambio	[1 2] [9 4 7] [1 2 4] [9 7]
(1 2 3)	(4 5)	9>7 (4,5) -> scambio	[1 2 4 7] [9]
(1 2 3 4)	(5)	Nessuno	[1 2 4 7 9]

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- esempio. Sia assegnato il vettore: v=[9 2 1 4 7]. Allora l'algoritmo procede iterativamente nel seguente modo:

```
scambio=1;
i=1;
while (i<n & scambio==1)
   scambio=0;
   for j=i+1:n
      if (v(j)<v(i))
       temp=v(i);
       v(i)=v(j);
         v(j)=temp;
         scambio=1;
      end
   end
   i=i+1;
end
```

Parte Ordinata	Parte Disordinata	Confronto	gorgogliamento
Vuota	(1 2 3 4 5)	4<7(4,5) 1<4 (3,4) 2>1(2,3) -> scambio 9>1(1,2) -> scambio	[9 2 1 4 7] [9 2 1 4 7] [9 1 2 4 7] [1] [9 2 4 7]
(1)	(2 3 4 5)	4<7(4,5) 2<4 (3,4) 9>2 (2,3) -> scambio	[1] [9 2 4 7] [1] [9 2 4 7] [1 2] [9 4 7]
(1 2)	(3 4 5)	4<7(4,5) 9>4 (3,4) -> scambio	[1 2] [9 4 7] [1 2 4] [9 7]
(1 2 3)	(4 5)	9>7 (4,5) -> scambio	[1 2 4 7] [9]
(1 2 3 4)	(5)	Nessuno	[1 2 4 7 9]

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Per la realizzazione dell'algoritmo si utilizza una sola funzione **ordina** con le seguenti caratteristiche:

- Parametri di Input: [v] vettore d'ingresso

- Parametri di Output: [v] vettore ordinato

- Variabili locali: [i,n,scambio,temp] contatore di ciclo, riempimento del vettore, variabile binaria che indica se è avvenuto qualche scambio nell'iterazione corrente e variabile d'appoggio per effettuare lo scambio ai fini dell'accodamento nell'insieme ordinato

- Funzioni richiamate: **length** per il calcolo del riempimento del vettore

Implementazione

```
% FUNZIONE PER L'ORDINAMENTO DI UN VETTORE
% La funzione [v]=bubblesort(v)
% permette di ordinare in senso crescente il vettore v.
% Esempio:
%              v=[4 1 3 2 5]
%              [v]=bubblesort(v)
%              v=[1 2 3 4 5]
function [v]=bubblesort(v)
% variabili locali
% calcolo del riempimento
n=length(v);
% variabile locale che segnala se nell'iterazione corrente ci sono stati scambi
scambio=1;
% contatore di ciclo
i=1;
% ciclo di ordinamento
while (i<n & scambio==1)
  scambio=0;
  for j=i+1:n
    % spostamento del valore minimo verso la parte ordinata
    if (v(j)<v(i))
    temp=v(i);
    v(i)=v(j);
      v(j)=temp;
      scambio=1;
    end
  end
  i=i+1;
end
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- Si riporta un esempio d'uso del programma mediante la shell (interprete dei comandi) dell'ambiente MATLAB.

```
» help bubblesort
  FUNZIONE PER L'ORDINAMENTO DI UN VETTORE
  La funzione [v]=bubblesort(v)
  permette di ordinare in senso crescente il vettore v.
  Esempio:
                v=[4 1 3 2 5]
                [v]=bubblesort(v)
                v=[1 2 3 4 5]
» v=[9 2 1 4 7];
» [v]=bubblesort(v)
v = 1     2     4     7     9
```

```
function [v]=bubblesort(v)
n=length(v);
scambio=1;
i=1;
while (i<n & scambio==1)
   scambio=0;
      for j=i+1:n
         if (v(j)<v(i))
         temp=v(i)
         v(i)=v(j)
         v(j)=temp
         scambio=1
         end
      end
   i=i+1
end
```

Gli algoritmi di base in MATLAB

```
function [v]=bubblesort2(v)
n=length(v);
for j=1:1:n-1
      for i=1:1:n-1
            if (v(i)>v(i+1))
            v(i)=v(i+1)
            v(i+1)=temp
            end
    end
end
```

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA
LUIGI VANVITELLI
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA CIVILE
DESIGN EDILIZIA E AMBIENTE

Gli algoritmi di base in MATLAB

- v=[7 5 9 4 1]
- [v]=bubblesort(v)
- v= [7] [5] 9 4 1
- v = [5] 7 9 [4] 1
- v = [4] 7 9 5 [1]
- v = 1 [7] 9 [5] 4
- v = 1 [5] 9 7 [4]
- v = 1 4 [9] [7] 5
- v = 1 4 [7] 9 [5]
- v = 1 4 5 [9] [7]
- v = 1 4 5 7 9

- v=[7 5 9 4 1]
- [v]=bubblesort2(v)
- v= [7] [5] 9 4 1
- v = 5 7 [9] [4] 1
- v = 5 7 4 [9] [1]
- v = 5 [7] [4] 1 9
- v = 5 4 [7] [1] 9
- v = [5] [4] 1 7 9
- v = 4 [5] [1] 7 9
- v = [4] [1] 5 7 9
- v = 1 4 5 7 9