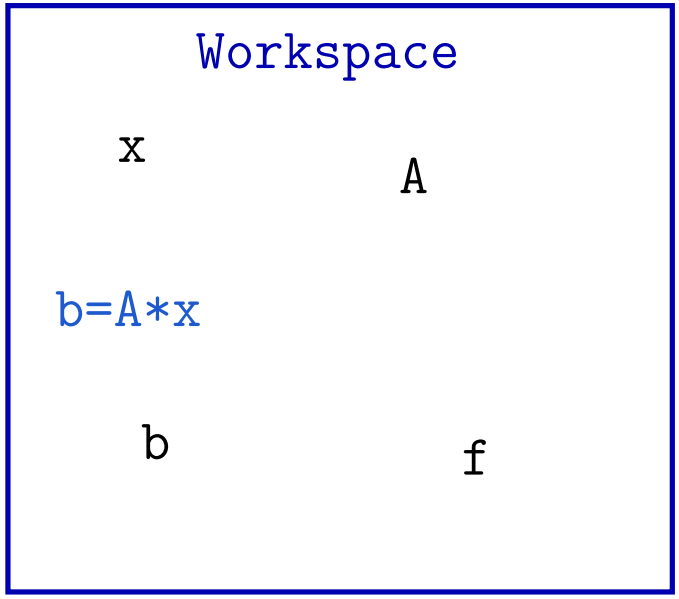
Unità di programma in Octave

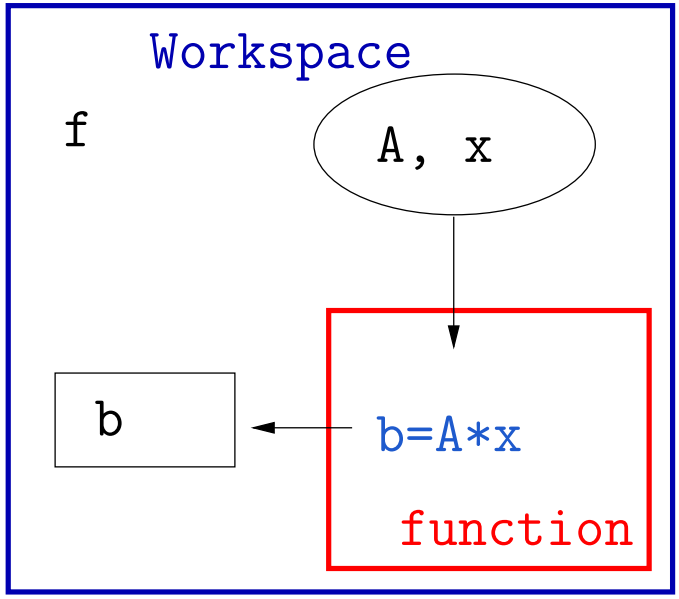
script
è una sequenza
di istruzioni
MATLAB/Octave

function
è una unità di programma
con input e output

GLOBALI ← VARIABILI → LOCALI

Entrambi hanno estensione .m.





Unità di programma in Octave

script
è una sequenza
di istruzioni
MATLAB/Octave

function
è una unità di programma
con input e output

GLOBALI ← VARIABILI → LOCALI

Entrambi hanno estensione **.m**.

Come eseguire uno script:

```
>> nomefile
```


e una function:

```
>> [out]=nomefile(in)
```


dove:
`out`=lista dei parametri di output
`in`=lista dei parametri in input (passaggio per valore).

Per le function si consiglia di salvare il file con lo stesso nome dato internamente alla function.

FOR LOOP

```
for <contatore>
   {espressione}
end
```

Esempio: Calcolare la somma tra due vettori $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{10} (= \mathbb{R}^{10\times 1})$
Nella finestra dell'editor

```
a=[....]; b=[....];
for j=1:10
c(j)=a(j)+b(j);
end
c=c';
```

oppure

```
a=[....]; b=[....];
c=zeros(10,1);
for j=1:10
c(j)=a(j)+b(j);
end
```

N.B. Se il vettore `c` non esiste, il ciclo `for` genera automaticamente un array di tipo riga.
Salvare il file con nome `prova1.m`. Dalla finestra dei comandi MATLAB:

```
>> prova1
```

Esempio: Calcolare la differenza tra due matrici a e b di uguale dimensione (n, m). ($c = a + b$, con $c_{ij} = a_{ij} + b_{ij}$, per $i = 1, ..., n$ e $j = 1, ..., m$.

Nella finestra dell'editor

```
for i=1:n
for j=1:m
c(i,j)=a(i,j)+b(i,j);
end
end
disp('La matrice somma e':'), c
```

CALCOLO DI UNA SOMMATORIA con un for loop.

Es. $s = \sum_{k=1}^{n} \frac{1}{k}$.

$s = 0; s = s + 1; s = s + 1/2; s = s + 1/3; .....$

$s = s + 1/n;$

Nella finestra dell'editor

```
% calcolo di una sommatoria
s=0; n=...
for k=1:n
s=s+1/k;
end
fprintf('La somma e' %8.6f \n',s)
```

`fprintf` è il comando per stampare con formato,
`%8.6f` è il formato con cui voglio scrivere il risultato `s`
`\n` dice di andare a capo dopo la stampa **N.B.** Quando si deve calcolare una somma con addendi tutti con lo stesso segno, ma molto diversi in valore assoluto è preferibile partire dagli addendi di valore assoluto minore per finire con gli addendi di valore assoluto maggiore:

```
s=0; for k=20:-1:1 s=s+1/k; end
```

N.B. Quando si deve calcolare una somma con addendi discordi è bene prima calcolare due somme parziali (tutti i positivi e tutti i negativi) e poi fare la somma fra i due valori ottenuti.

Scrittura di una function per il calcolo della sommatoria

Input: `n`
Output: `s`.
Nella finestra dell'editor:

```
function [s]=sommatoria(n)
s=0;
for k=n:-1:1
s=s+1/k;
fprintf('La somma e' %8.6f \n',s)
end
```

Dalla finestra dei comandi di MATLAB:

```
>> s=sommatoria(20);
```

oppure

```
>> n=20; s=sommatoria(n);
```

IF, IF .. ELSE, IF... ELSEIF ... ELSE

```
if <condizione>
  {espressione}
end

if <condizione>
  {espressione1}
else
  {espressione2}
end

if <condizione1>
  {espressione1}
elseif<condizione2>
  {espressione2}
else
  {espressione3}
end
```

OPERATORI RELAZIONALI

<	minore
>	maggiore
<=	minore o uguale
>=	maggiore o uguale
==	uguale
~=	diverso

N.B. Per il carattere ~: ALT 126

OPERATORI LOGICI

~	not
&	and
|	or
&&	short-circuit and
||	short-circuit or

Esempio: Se $0 < x < 1$ allora calcola $y = 1/x$

```
if x>0 && x<1
y=1/x
end
```

Esempio: Se $x < 0$ o $x \geq 10$ allora calcola $y = sin(x)$ altrimenti calcola $y = cos(x)$.

```
if x<0 || x>=10
y=sin(x);
else
y=cos(x);
end
```

Esempio: Calcolare il fattoriale di un numero intero.

```
function [f]=fattoriale(n)
% calcolo del fattoriale di un numero intero
if n< 0
f=0; disp ('N.B. l'input e' negativo')
elseif n==0
f=1;
else
f=prod(1:n);
end
```

```
>>n=...; f=fattoriale(n);
```

ciclo WHILE

```
while <condizione>
  {espressione}
end
```

Esempio: ciclo while per calcolare la precisione di macchina in base 2

```
epsilon=1;
while epsilon+1 > 1
epsilon=epsilon/2;
end
epsilon=epsilon*2;
```

N.B. All'interno del ciclo while bisogna sempre modificare la variabile da cui dipende la "condizione", altrimenti il ciclo diventa infinito.
In questo caso la condizione è:

```
epsilon+1 > 1
```

e la variabile da modificare è: `epsilon`.

Prodotto tra matrici

Date $a \in \mathbb{R}^{n \times m}$, e $b \in \mathbb{R}^{m \times p}$ calcolare $c = a\ b$. Ricordo che

$$c_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj}, \qquad i = 1, \ldots, n, \quad j = 1, \ldots, p$$

Esercizio Scrivere una function matlab che implementi il prodotto tra due matrici, secondo la regola sopra scritta.
Input: le matrici a e b
Output: la matrice prodotto c

All'interno della function calcolare le dimensioni delle matrici, fare i controlli sulla compatibilità delle dimensioni.
`[n,m]=size(a)` restituisce n. di righe e n. di colonne di a
`n=length(a)` restituisce il massimo tra il n. di righe ed il n. di colonne di a