Il comando `lu` e il comando `chol`

`[L,U,P]=lu(A)`

`L*U = P*A`

`L` triangolare inferiore con tutti gli elementi diagonali pari a 1.

`U` triangolare superiore.

`P` matrice di permutazioni.

`[L,U]=lu(A)`

`L*U = A`

`L` non è triangolare inferiore ma una permutazione di righe la rende triangolare inferiore con tutti gli elementi diagonali 1.

`U` triangolare superiore.

`H=chol(A)`

`A` deve essere simmetrica definita positiva

`H'*H = A`

`H` triangolare superiore con tutti gli elementi diagonali positivi.

Esempio

Una funzione di Matlab che implementa il metodo della sostituzione in avanti.

```
function [x]=forward(L,b)
n=length(b);
x=zeros(n,1);
if min(abs(diag(L))) == 0
  x='Matrice tringolare inferiore singolare';
  return
end
x(1)=b(1)/L(1,1);
for i=2:n
  s= L(i,1:i-1)*x(1:i-1);
  x(i)=(b(i)-s)/L(i,i);
end
```

I condizionali in Matlab

```
if condizione 1
    gruppo di comandi 1
elseif condizione 2
    gruppo di comandi 2
elseif condizione 3
    gruppo di comandi 3
else
    gruppo di comandi 4
end
```

- Le condizioni sono espressioni logiche.
- Esempi di operatori logici: and &, or |, not ~.
- Esempi di operatori relazionali: uguale ==, maggiore o uguale >=, minore <.

I cicli in Matlab

- Ciclo `for`

```
a=1;
for i=1:10
  a=a*i;
end
```

Ripete le istruzioni presenti nel ciclo per tutti i valori dell'indice contenuti in un certo vettore riga.

- Ciclo `while`

```
a=1;
cont=1;
while cont <=10
  a=a*cont;
  cont=cont+1;
end
```

Ripete le istruzioni presenti nel ciclo sino a quando una data espressione logica è vera.

Esercizio

- Scrivere una funzione di Matlab che implementi il metodo iterativo di Jacobi per l'approssimazione della soluzione di un sistema lineare.

```
function [x,nit]=jacobi(A,b)
n=length(b);
x0=zeros(n,1);
x=x0;
toll=1.e-8;
nmax=500;
nit=0;
if min(abs(diag(A))) == 0
  x='A ha un elemento diagonale uguale a zero';
  return
end
inc=1; res=1;
while inc > toll | res/norm(b) > toll
  for i=1:n
    x(i)=(b(i)-A(i,1:i-1)*x0(1:i-1)-A(i,i+1:n)*x0(i+1:n))/A(i,i);
  end
  inc=norm(x-x0);
  res=norm(b-A*x);
  x0=x;
  nit=nit+1;
  if nit==nmax, return, end
end
```