Metodi numerici per l'ingegneria navale - 15 giugno 2010 - Testo formato da **due** pagine.

Nota: Creare sul desktop una cartella in cui mettere tutti i file.
Come nome della cartella usare il proprio cognome.
Al termine della prova cliccare sul tasto start, scegliere il menu "Computer" e aprire il disco (W:Consegna).
Trascinare l'icona della cartella contenente i file dentro la finestra che si è aperta nel momento in cui si è fatto doppio click sul disco W:

Nel primo file creato, scrivere, come commento, anche Nome, Cognome e indirizzo e-mail
È possibile (anzi consigliabile) aggiungere, ove occorra, righe di commento agli m-file

$\boxed{\mathcal{A}}$ Costruire un m-file **funzione** col nome **`alfa(x,n)`**. Le variabili di input saranno x numero reale e n numero intero maggiore di 2. La funzione calcolerà la matrice m di formato $n \times n$ così fatta:

$$m = \left(\begin{array}{c|cccc|c} 1 & 1 & 1 & \cdots & 1 & 1 \\ 1.5 & n & 0 & \cdots & 0 & x \\ 2 & 0 & n-1 & ... & 0 & 0 \\ ... & ... & ... & ... & ... & ... \\ ... & 0 & 0 & ... & 3 & 0 \\ ... & x & 0 & ... & 0 & 2 \end{array}\right)$$

Consigli per un file di buona qualità:

1. Nel listato evitare il più possibile i comandi del tipo **`for...end`**.
2. Controllare bene le variabili di input e output.
3. Porre (dopo aver verificato che il file funzioni) dei simboli **`;`** alla fine di ogni istruzione.
4. Sarebbe bene che la funzione fosse *a prova di errore*:
   Se n non è intero o è inferiore a 3, la funzione potrebbe sostituire 3 a n e dare un avvertimento (comando **`warning`**)

$\boxed{\mathcal{B}}$ Consideriamo per ogni x la matrice **`alfa(x,5)`** e i suoi autovalori $\lambda_1(x) \leq \cdots \leq \lambda_5(x)$ ordinati.

Definiamo le 5 funzioni $\lambda_k(x)$:

Disegnare (sovrapposti) i grafici delle 5 funzioni $\lambda_k(x)$ nell'intervallo $[-2, 7]$ usando il passo 0.01.

Per ognuna di esse determinare (a meno di 0.02) massimo, minimo, punto di massimo e punto di minimo. Su schermo potrebbe apparire qualcosa del tipo:

```
lambda_1:  max= ...  , in x=...      min=..., in x=...
lambda_2:...
...
lambda_5:...
```

Salvare i comandi relativi in un m-file di tipo script col nome "beta.m".

Consiglio:

Usare la funzione **`sort`**

$\boxed{\mathcal{C}}$ Consideriamo la funzione $z(x, y)$ definita nel seguente modo:

$$z(x, y) = x^3 - y^2 - 3x + ky$$

Costruire un m-file funzione denominata **`gamma`** avente la variabile **`k`** input e un vettore **`(x,y,s)`** in output.

La funzione dovrà

1. Disegnare il grafico della funzione z nel rettangolo $[-2, 2] \times [-4, 2]$ usando un passo $p = 0.1$

2. Esaminare il grafico della funzione nel rettangolo dato, indi trovarne (con approssimazione circa 0.2) il punto di sella, che sarà un punto (x, y) dove
   $z(x, y) < z(x + p, y)$ e $z(x, y) < z(x - p, y)$, ma $z(x, y) > z(x, y + p)$ e $z(x, y) > z(x, y - p)$.
3. Dare come outupt il vettore **`(x,y,s)`** contenente il punto **`(x,y)`** di sella e il valore **`s`** di z nel punto di sella.

Consigli:

1. Il nocciolo del problema sta nell'usare correttamente le array **`xx,yy`** costruite mediante il comando **`meshgrid`**.
2. I **`k`** interessanti sono k piccoli (anche **`k=0`**)

$\mathcal{D}$ Implementare l'algoritmo di DeCastelajau per disegnare una porzione di parabola, dato il poligono di controllo e il numero di punti.
Si costruirà una funzione **`decast(p1,p2,p3,n)`** che avrà come input tre array di due elementi che forniranno i punti del poligono di controllo e il numero n dei punti.
La funzione non avrà variabile di output, ma disegnerà la porzione di parabola e il poligono di controllo.
Consiglio:
Considerare la possibilità di usare la funzione **`linspace`**.