Laboratorio di Calcolo per Fisici,
Esame del 14/07/2025
A.A. 2024/2025

Nome _____ Cognome _____

Matricola _____ ☐ Ritirato/a

Lo scopo di questa esercitazione è scrivere un programma in C e uno script in python seguendo la traccia riportata di seguito. Si tenga presente che:

1. Per svolgere il compito si hanno a disposizione 3 ore.
2. Si possono usare libri di testo, prontuari e gli appunti ma non è ammesso parlare con nessuno né utilizzare cellulari, tablet o laptop, pena l'annullamento del compito.
3. Il programma va scritto e salvato esclusivamente sul computer del laboratorio, a cui si deve accedere utilizzando come username **studente** e come password **informatica**
4. **Tutti i file vanno salvati in una cartella chiamata `ELCGEN_NOME_COGNOME` nella home directory**, dove `NOME` e `COGNOME` indicano rispettivamente il tuo nome e cognome. Ad esempio lo studente *Marco Rossi* deve creare una cartella chiamata `ELCGEN_MARCO_ROSSI` contenente tutti i file specificati nel testo. **Tutto ciò che non si trova all'interno della cartella suddetta non verrà valutato.** In tutti i programmi e script inserisci all'inizio un commento con il tuo nome, cognome e numero di matricola.
5. **Dovete consegnare il presente testo indicando nome, cognome e numero di matricola** (vedi sopra), barrando la casella "Ritirato/a" se ci si vuole ritirare, ovvero se non si vuole che la presente prova venga valutata.

Si consideri un'equazione di secondo grado del tipo:

$$x^2 + c_1 x + c_0 = 0 \tag{1}$$

Come noto le due radici (o **zeri**) di questa equazione si possono trovare come segue:

$$x_{1,2} = \frac{-c_1 \pm \sqrt{\Delta}}{2} \tag{2}$$

dove $\Delta = c_1^2 - 4c_0$, e si assume $\Delta \geq 0$. Tuttavia questa soluzione può essere molto inaccurata numericamente. Un'alternativa numericamente migliore è scrivere le due soluzioni nel seguente modo:

$$x_1 = \begin{cases} -\frac{c_1+\sqrt{\Delta}}{2} & \text{if } c_1 \geq 0 \\ \frac{-c_1+\sqrt{\Delta}}{2} & \text{if } c_1 < 0 \end{cases} \tag{3}$$

e

$$x_2 = \begin{cases} 0 & \text{if } x_1 = 0 \\ c_0/x_1 & \text{if } x_1 \neq 0 \end{cases} \tag{4}$$

Lo scopo di questo prova d'esame è scrivere un codice che generi casualmente coppie di zeri reali, calcoli la corrispondente equazione quadratica e la risolva con i due metodi suesposti. Dopo di che il programma confronterà le due coppie di zeri con quelli esatti, valutando l'errore. Ripetendo quest'operazione un numero `NZERI` di volte, si può fare un confronto calcolando le frequenze con cui si trova un certo errore ϵ utilizzando le due formule. Definendo x_0^n e x_1^n la coppia di zeri ottenuti numericamente, dove $x_0^n < x_1^n$, e x_0^e e x_1^e la coppia di zeri esatti, con $x_0^e < x_1^e$, si definisce l'errore come

$$\epsilon = \left| \frac{x_0^n - x_0^e}{x_0^e} \right| + \left| \frac{x_1^n - x_1^e}{x_1^e} \right|. \tag{5}$$

Per fare tale confronto si dovranno utilizzare i valori di ϵ delle due soluzioni numeriche per costruire due istogrammi `istoA` e `istoB`, che poi andranno opportunamente graficati.

► **Prima parte:**

Si realizzi un programma in C, chiamato `nome_cognome.c` (tutto minuscolo, senza eventuali spazi, accenti o apostrofi) che calcoli la frequenza degli errori relativi per i due approcci numerici discussi in precedenza e salvi tali frequenze in un file da utilizzare nella seconda parte pere generare un grafico in python. In particolare il programma dovrà:

1. Definire tramite delle macro il numero di coppie di zeri da generare, `NZERI` = 10000, e il numero di bin per l'istogramma, `BINS` = 20.
2. Generare due zeri x_0 e x_1 casualmente usando le seguenti formule:

$$x_0 = 10^{30} + (\xi_0 - 0.5) \cdot 10^{20} \quad (6)$$
$$x_1 = 10^{-30} + \xi_1 \cdot 10^{20} \quad (7)$$

   dove ξ_0 e ξ_1 sono due numeri casuali uniformi estratti nell'intervallo $[0, 1)$.
3. Calcolare i coefficienti c_0 e c_1 della corrispondente equazione quadratica come segue:

$$c_0 = x_0 \cdot x_1 \quad (8)$$
$$c_1 = -(x_0 + x_1) \quad (9)$$

4. Risolvere l'equazione quadratica ottenuta usando prima l'eq. (2) e poi le eq. (3) e (4).
5. Ordinare i due zeri esatti e quelli ottenuti numericamente, in modo che il primo sia minore del secondo.
6. Calcolare l'errore relativo totale ϵ delle due diverse soluzioni numeriche usando l'Eq. (5).
7. Se l'errore relativo totale è diverso da 0, aggiornare gli istogrammi, che chiameremo `istoA` e `istoB`, incrementando di uno il k-esimo bin, definito come:

$$k = \lfloor \log_{10}(\epsilon) + \texttt{BINS} \rfloor \quad (10)$$

   dove $\lfloor x \rfloor$ indica la parte intera di x, $\log_{10}$ è il logaritmo in base 10 (da calcolarsi con la funzione `log10`) e `BINS` è il numero di bin dell'istogramma. *Suggerimento*: prima di aggiornare l'istogramma controllate che $k \geq 0$ e $k <$ `BINS`.
8. Stampare per le prime dieci coppie generate le radici esatte e il valore di ϵ per le soluzioni numeriche, stampando i numeri con 15 cifre significative. *Suggerimento:* per stampare un numero con 15 cifre significative potete usare `%.15G`.
9. Stampare su di un file chiamato `isto.dat` i due istogrammi `istoA` e `istoB` ottenuti dopo aver generato `NZERI` coppie di zeri. Il file deve contenere 3 colonne, la prima con i valori di k e la seconda e la terza con i valori dei due istogrammi divisi per `NZERI`, ottenendo così la frequenza con cui si ottengono gli errori in uno dei `BINS` intervalli considerati. *Suggerimento*: `istoA` dovrebbe avere un picco intorno a 13, mentre `istoB` intorno a 4.

Nello scrivere il programma si richiede che vengano implementate almeno le seguenti funzioni:

- `ordina_zeri()` che ordina i due zeri passati come argomento tramite un opportuno array, mettendo il più piccolo prima.
- `calc_err()` che calcola l'errore relativo ϵ degli zeri ottenuti numericamente tramite l'equazione (5). Tale funzione prende come argomento due array, uno contenente gli zeri esatti e l'altro quelli ottenuti numericamente, e restituisce ϵ.
- `solve_quadratic()` che trova gli zeri con la formula in Eq. (2). Tale funzione prende come argomenti due array: uno con i coefficienti dell'equazione quadratica e un altro dove verrano salvati i valori degli zeri calcolati numericamente.
- `solve_quadratic_best()` che trova gli zeri con le formule in Eq. (3) e (4). Tale funzione prende come argomenti due array: uno con i coefficienti dell'equazione quadratica e un altro dove verrano salvati i valori degli zeri calcolati numericamente.
- `updisto()` che prende come parametri un array contenente l'istogramma e un valore di ϵ, e utilizza l'equazione (10) per incrementare il bin dell'istogramma corrispondente al valore di ϵ.

► **Seconda parte:** Utilizzando il file `isto.dat` generato eseguendo il programma scritto nella prima parte, creare con `python` un grafico che mostri le frequenze dei vari errori per i due approcci usati, aggiungendo un'opportuna legenda, label per gli assi x e y e un titolo.