Programmazione e Algoritmi 1

A.A. 2023/24 — Compito del 20 giugno 2024

prof. Gianluca Amato

Gli esercizi di programmazione saranno valutati sulla base della correttezza, efficienza e comprensibilità della soluzione proposta. In generale, **se volete usare una funzione o un metodo che non è stato presentato a lezione, chiedete prima al docente se è consentito**.

Esercizio 1 (5 punti)

Spiegare la differenza tra un tipo di dato *mutabile* e uno *immutabile*, fornendo un esempio per ciascuno di essi.

Esercizio 2 (8 punti)

Si consideri il seguente codice Python, e se ne scriva la traccia di esecuzione negli appositi moduli.

```
1  def mistero(l1, l2):
2      for i in range(len(l1)):
3          l1[i] = l1[i] + l2[-i-1]
4
5  def mistero2(l):
6      for x in l:
7          print(x)
8
9  x = [1, 2, 3, 4]
10 y = [50, 60, 70, 80]
11 mistero(x, y)
12 mistero2(x)
```

Esercizio 3 (5 punti)

Scrivere una funzione `raddoppia` che prende come parametri una lista di interi `l` e una matrice `m` (implementata al solito come lista di liste). La funzione deve modificare la matrice `m` in maniera tale che uno ed un solo elemento per ogni riga della matrice venga raddoppato. In particolare, nella riga i-esima l'elemento da raddoppiare è quello nella colonna `l[i]`. La lista `l` invece non deve subire modifiche.

Ad esempio, se `l = [1, 2, 0]` e `m = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]`, dopo la chiamata `raddoppia(l, m)` la matrice `m` diventa `[[1, 4, 3], [4, 5, 12], [14, 8, 9]]`.

Si può assumere che la lista `l` contenga dati corretti, ad esempio che abbia lunghezza uguale al numero di righe della matrice `m`, e che i valori presenti all'interno di `l` siano indici validi per le colonne della matrice `m`.

Si determini, infine, **la complessità computazionale** del metodo `raddoppia`.

Esercizio 4 (5 punti)

Scrivere due test nel framework `pytest` per verificare il corretto funzionamento di `raddoppia`. In particolare, si controllino i seguenti casi:

- l'input di esempio dell'Esercizio 3;
- `m` è una matrice generata casualmente con 3 righe e 3 colonne e `l=[0,0,0]`.

Esercizio 5 (5 punti)

Scrivere una funzione `raddoppia2` simile alla funzione `raddoppia` dell'Esercizio 3, ma per la quale la correttezza della lista `l` non è assicurata. Potrebbe quindi accadere che, con questi input, la funzione `raddoppia` generi un errore. La funzione `raddoppia2`, in questi casi, non deve generare errori, ma semplicemente non apportare nessuna modifica ad `m`.

Ad esempio, l'input `l = [1, 6, 0]` e `m = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]` non è valido, perché la riga numero 1 di `m` ha solo 3 elementi, quindi non ha senso raddoppiare l'elemento in colonna 6. In questo caso, la funzione `raddoppia2` **non deve modificare nulla, neanche le altre righe di** `m`.

Suggerimento. Ci sono due modi per risolvere l'esercizio. Uno è controllare preventivamente che la lista `l` sia corretta, e in caso contrario non fare nulla. L'altro è salvare una copia di `m` all'inizio della funzione, e se si verifica un errore ripristinare la matrice `m` alla copia salvata.

Esercizio 6 (5 punti)

Scrivere una programma che visualizza una finestra quadrata, quasi completamente vuota, ma con un riquadro in basso a destra contenente la scritta "Exit". Ogni volta che si clicca all'interno della finestra appare una cerchio pieno di raggio 10 pixel centrato nel punto in cui si è cliccato. Il colore del cerchio si alterna secondo il seguente schema: rosso, verde, blu, ciano, magenta e giallo, per poi ricominciare dal rosso. Se si clicca sul riquadro "Exit", la finestra si chiude.

Qua sotto c'è un esempio di come dovrebbe apparire la finestra appena il programma è stato lanciato e dopo 4 click.