Programmazione e Algoritmi 1 - Prova pratica

A.A. 2024/25 — Compito del 6/6/2025

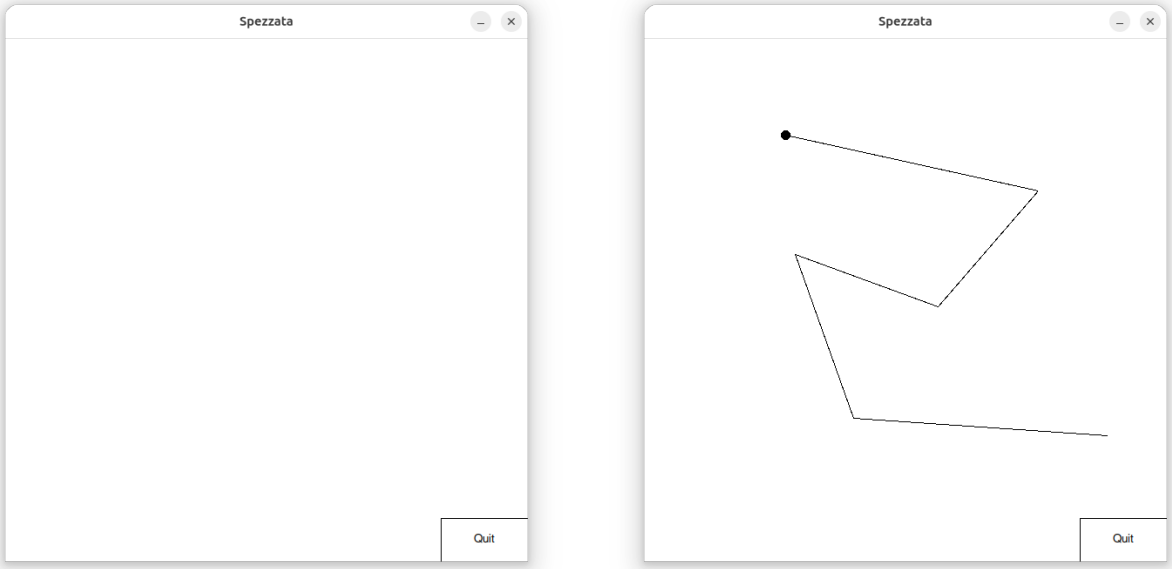
prof. Gianluca Amato

Esercizio 1 (8 punti)

Scrivere un programma che mostra una finestra completamente bianca di dimensione 600x600 pixel, tranne un piccolo rettangolo in basso a destra con la scritta "Quit". Il programma disegna una spezzata, ovvero una sequenza di segmenti, messi in modo tale che un segmento inizia dove finisce il precedente. In pratica, ad ogni click del mouse il programma traccia un nuovo segmento, che va dalla fine del segmento precedente al punto dove si è cliccato. Ci sono solo due eccezioni a questa regola:

1. Il primo click del mouse non disegna alcun segmento, ma un cerchio nero di raggio 10 pixel centrato nel punto del click. Questo sarà anche il punto di partenza per il primo segmento.
2. Quando si clicca sul rettangolo in basso a destra il programma termina.

Trovate qui sotto due screenshot: in quello a sinistra il programma è appena partito, mentre in quello a destra il mouse è stato cliccato in 6 punti diversi, e il programma ha disegnato i segmenti che collegano questi punti.



Quando il programma termina, salva nel file `disegno.txt` le coordinate di tutti i punti che sono stati cliccati, in ordine di apparizione. Ogni riga deve contenere le coordinate di un punto, prima la coordinata x e poi la coordinata y, separate da una virgola. Ad esempio, nel caso dello screenshot a destra, il corrispondente file `disegno.txt` conterrebbe quanto segue:

```
162,110
452,174
337,307
173,247
240,435
532,455
```

Esercizio 2 (6 punti)

Scrivere la funzione `cumsum` (abbreviazione di *cumulative sum*) che prende come parametro una lista l e restituisce una nuova lista della stessa lunghezza in cui ogni elemento è la somma degli elementi di l fino a quell'indice. Ad esempio, il risultato di `cumsum([1, 2, 3, 4])` è `[1, 3, 6, 10]` perché $1 + 2 = 3$, $1 + 2 + 3 = 6$ e $1 + 2 + 3 + 4 = 10$.

Scrivere quindi la funzione `cumsum_inplace` che è simile a `cumsum`, ma opera sul posto: non restituisce nulla, e invece modifica direttamente la lista l passata come argomento. Ad esempio, se `l=[1, 2, 3, 4]` allora dopo l'esecuzione di `cumsum_inplace(l)` la lista l diventa `[1, 3, 6, 10]`.

Esercizio 3 (3 punti)

Utilizzare il framework `pytest` per collaudare le funzioni `cumsum` e `cumsum_inplace` dell'esercizio precedente. Si considerino in particolare, per entrambe le funzioni, i casi corrispondenti ai seguenti parametri in input:

- l'esempio fornito nel testo dell'esercizio;
- una lista con un solo elemento;
- una lista vuota.

Programmazione e Algoritmi 1 - Prova scritta

A.A. 2024/25 — Compito del 6/6/2025

prof. Gianluca Amato

Esercizio 4 (7 punti)

Eseguire passo-passo il seguente codice Python, utilizzando l'apposito modulo:

```python
def myst1(x, y):
    if x == 0:
        return y
    else:
        z = myst1(x - 1, y)
        return z + 1

def myst2(x, y):
    if x == 0:
        return 0
    elif x == 1:
        return y
    else:
        z = myst2(x - 1, y)
        return myst1(y, z)

print(myst2(3, 2))
```

Esercizio 5 (5 punti)

Scrivere una funzione `is_special` che prende come parametro una matrice m non frastagliata (ovviamente rappresentata come lista di liste) e restituisce `True` se la matrice ha entrambe le seguenti proprietà:

- è quadrata, cioè ha lo stesso numero di righe e colonne;
- ogni elemento della matrice è un numero compreso tra 0 e 10, estremi inclusi.

In tutti gli altri casi la funzione deve restituire `False`.

Esercizio 6 (5 punti)

Si descriva la differenza tra i tipi di dato mutabili e quelli immutabili, e si faccia un esempio di ciascuno. Si spieghi inoltre come questa differenza si riflette nella possibilità che ha una funzione di modificare un oggetto passato come parametro.