Programmazione e Algoritmi 1 - Prova pratica

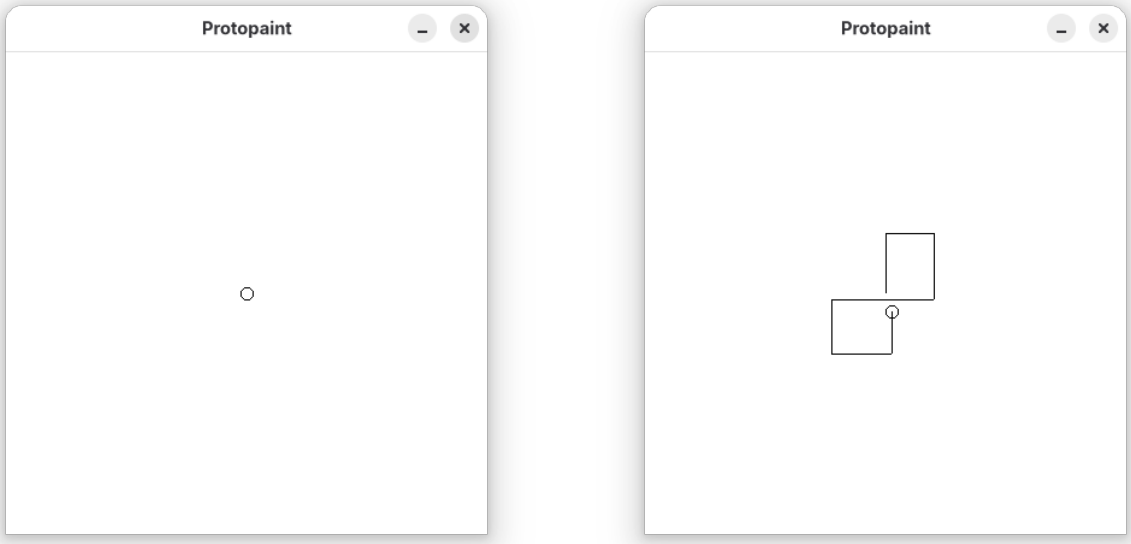
A.A. 2024/25 — Compito del 27/6/2025

prof. Gianluca Amato

Esercizio 1 (8 punti)

Realizzare un programma di disegno rudimentale utilizzando la libreria `ezgraphics`. Il programma deve mostrare inizialmente una finestra completamente vuota, tranne per un "pennello" stilizzato (ad esempio, si può utilizzare un cerchio, ma altre scelte sono possibili). Con i tasti cursore è possibile spostare liberamente il pennello sullo schermo. Quando l'utente preme il tasto *spazio*, il pennello entra in modalità disegno: mentre si sposta lascia una linea nera sullo schermo. Premendo di nuovo spazio il pennello esce dalla modalità disegno e torna a essere un semplice puntatore che non lascia tracce dove passa. Il programma termina quando l'utente preme il tasto *Esc*.

Quello che segue è un esempio di come potrebbe appairire il programma. Lo screenshot di sinistra è stato preso appena il programma è stato avviato, quello di destra dopo aver disegnato una specie di otto inclinato.



Suggerimenti. Per acquisire il tasto premuto dall'utente si può utilizzare il metodo `getKey()` della classe `GraphicsWindow`. Quando bisogna spostare il "pennello", conviene eliminare quello vecchio con il metodo `removeItem()` della classe `GraphicsCanvas` e ridisegnarlo nella nuova posizione.

Esercizio 2 (6 punti)

Scrivere una funzione `add_total` che prende come parametro una matrice di numeri, anche frastagliata, e la modifica sul posto aggiungendo una nuova colonna con i totali delle righe. Ad esempio, se

```
m = [
    [1, 2, 3],
    [4, 5]
]
```

dopo la chiamata a `add_total(m)` si avrà

```
m == [
    [1, 2, 3, 6],
    [4, 5, 9]
]
```

Scrivere quindi una funzione `add_total2` che è simile ad `add_total` ma restituisce una nuova matrice invece di modificare quella passata come parametro.

Esercizio 3 (3 punti)

Utilizzare il framework `pytest` per collaudare le funzioni `add_total` e `add_total2` 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 matrice con una sola riga e una sola colonna;
- una matrice senza nessuna riga.

Programmazione e Algoritmi 1 - Prova scritta

A.A. 2024/25 — Compito del 27/6/2025

prof. Gianluca Amato

Esercizio 4 (7 punti)

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

```
def mystery(m):
    s = 0
    for r in m:
        for v in r:
            if v > 0:
                s += v
    return s

m = [[1, -2, 3], [4, -5]]
print(mystery(m))
```

Esercizio 5 (5 punti)

Determinare la complessità computazionale asintotica in tempo, sia nel caso ottimo che nel caso peggiore, della seguente funzione

```
def mystery2(l):
    res = []
    for i in range(len(l)):
        if l[i] != 0:
            res.append(l[i])
    return res
```

in termine del numero di elementi n della lista passata come parametro. Si può assumere (anche se non è del tutto vero) che il metodo `append` richieda tempo costante.

Esercizio 6 (5 punti)

Scrivere la funzione `alternate_strings` che prende come parametri due stringhe e retituisce la stringa ottenuta alternando i caratteri delle due stringhe. Quando la stringa più corta termina, i restanti caratteri vengono presi dalla stringa più lunga. Ad esempio, `alternate_strings("abc", "12345")` restituisce `"a1b2c345"`.