

Progetto 1: Progettazione dell'interfaccia

Davide Bresolin
a.a 2026/27



UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Il primo esercizio chiedeva di inserire il codice per disegnare un quadrato in una definizione di funzione:

```
def quadrato():  
    for i in range(4):  
        forward(100)  
        right(90)
```

Inglobare un pezzo di codice in una funzione è chiamato **incapsulamento**. Serve per:

- Riutilizzare il codice senza riscriverlo.
- Migliorare la leggibilità e la documentazione del programma.

- Aggiungere parametri a una funzione la rende più generale.
- Permette di gestire casi diversi senza duplicare codice.

Due esercizi sulla generalizzazione:

- Aggiungere a quadrato un parametro lunghezza
- Disegnare poligoni regolari di un qualunque numero di lati

- L'**interfaccia** definisce come si usa una funzione:
 - Parametri richiesti.
 - Scopo della funzione.
 - Valori restituiti.
- Un'interfaccia “pulita” nasconde i dettagli non necessari.

Esempio: disegnare un cerchio



```
def cerchio(r):  
    circonferenza = 2 * 3.14 * r  
    n = 30  
    lunghezza = circonferenza / n  
    poligono(n, lunghezza)
```

- L'utente controlla solo il parametro significativo: il raggio.
- Lunghezza e numero di lati sono stabiliti dalla funzione.

```
def arco(r, angolo):  
    l_arco = 2 * 3.14 * r * angolo / 360  
    n = 30  
    l_passo = l_arco / n  
    for i in range(n):  
        forward(l_passo)  
        right(angolo/n)
```

- La funzione arco è meno collaborativa: **non possiamo usare poligono!**

- Il refactoring è la riorganizzazione del codice per:
 - Migliorare le interfacce.
 - Eliminare duplicazioni.
 - Favorire il riuso.
- Non modifica il comportamento esterno del programma.

Idea: creiamo una funzione più generale:

```
def polilinea(n, lunghezza, angolo):  
    for i in range(n):  
        forward(lunghezza)  
        right(angolo / n)
```

Ora possiamo riscrivere poligono e arco in modo che usino polilinea:

```
def poligono(n, lunghezza):  
    polilinea(n, lunghezza, 360)  
  
def arco(r, angolo):  
    l_arco = 2 * 3.14 * r * angolo / 360  
    n = 30  
    l_passo = l_arco / n  
    polilinea(n, l_passo, angolo)
```

- 1 Scrivere un piccolo programma funzionante.
- 2 Incapsulare parti autonome in funzioni.
- 3 Generalizzare le funzioni con parametri.
- 4 Ripetere il processo costruendo su funzioni esistenti.
- 5 Applicare refactoring per semplificare e migliorare.

- Permette di progettare man mano che si sviluppa.
- Favorisce codice modulare e facilmente estendibile.
- Aiuta a comprendere meglio il problema durante la scrittura.