

Esercizio 1

Si vuole progettare un semplice sistema per la gestione dei prodotti alimentari presenti all'interno di un magazzino digitale.

In generale, un **Prodotto Alimentare** è caratterizzato da:

- nome commerciale
- prezzo di vendita

All'interno del magazzino si distinguono almeno due categorie di prodotti, ciascuna con caratteristiche specifiche:

- **ProdottoFresco**: rappresenta prodotti con scadenza ravvicinata e mantiene le caratteristiche base ereditandole pubblicamente da Prodotto.
- **ProdottoConservato**: rappresenta prodotti a lunga conservazione, anch'esso derivato pubblicamente da Prodotto.

Per la gestione dei prodotti si realizza una classe:

- **GestoreProdottiAlimentari**, che mantiene al proprio interno un contenitore (ad esempio `vector<Prodotto*>`) con l'elenco dei prodotti attualmente presenti in magazzino.

Le classi devono essere implementate prevedendo:

1. **Costruttori** che inizializzino correttamente sia i dati ereditati dalla classe base sia gli eventuali campi specifici delle classi derivate.
2. Una funzione `stampaDettagli()` (o equivalente) che mostri a schermo le informazioni relative al prodotto, sfruttando il polimorfismo per garantire che la funzione corretta venga chiamata in base al tipo reale dell'oggetto.
3. La classe `GestoreProdottiAlimentari` deve offrire alcune funzionalità gestionali, tra cui:
 - inserimento di nuovi prodotti nel vettore
 - stampa di tutti i prodotti presenti
 - ricerca di un prodotto dato il nome
 - calcolo del totale del valore di magazzino
4. Il programma principale (`main()`) deve mostrare l'utilizzo del sistema

Esercizio 2

Si vuole progettare un semplice sistema software per la gestione dei veicoli utilizzati da un servizio di autonoleggio.

In generale, un **Veicolo** è caratterizzato da:

- una targa univoca
- la marca
- il costo di noleggio giornaliero

All'interno del sistema si distinguono due tipologie principali di veicolo:

- **Auto:** eredita pubblicamente da Veicolo e aggiunge il numero di posti disponibili.
- **Furgone:** eredita pubblicamente da Veicolo e aggiunge il volume di carico espresso in litri.

La classe base Veicolo deve dichiarare una funzione stampaDettagli, che permetta di mostrare i dati del veicolo sfruttando il polimorfismo.

Per la gestione della flotta si introduce una classe:

- **GestoreNoleggio**, che mantiene al suo interno un contenitore (ad esempio `vector<Veicolo*>`) contenente i veicoli disponibili nel sistema.

Le classi devono essere realizzate rispettando i seguenti vincoli:

1. Devono esistere costruttori in grado di inizializzare correttamente sia i campi della classe base sia quelli specifici delle classi derivate.
2. Ogni classe derivata deve ridefinire la funzione stampaDettagli() per mostrare, oltre alle informazioni generiche, anche quelle aggiuntive della tipologia di veicolo.
3. Il GestoreNoleggio deve fornire almeno le seguenti funzionalità:
 - aggiunta di un veicolo alla flotta
 - stampa completa dell'elenco dei veicoli
 - ricerca di un veicolo dato il numero di targa
 - rimozione di un veicolo indicato tramite targa
4. Il programma principale (main()) deve dimostrare l'utilizzo del sistema