# Esercitazione di Fine Settimana – Week 2

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Michela |
|  |  | Cognome | Murtas |
|  |  | Data | 17 settembre ’21 |

Leggete attentamente ogni domanda e argomentare quanto più possibile fornendo anche degli esempi.

1. **Descrivere le modalità per ritornare più valori da un metodo in C#**

Per far restituire più valori in C# bisogna inserire in una funzione uno dei seguenti parametri:

* Array
* Collections
* Tupla
* Oggetto di tipo classe

1. **Descrivere le due tipologie di casting tra tipi in C#**

La prima tipologia di casting è il “Casting esplicito”, e consiste nell’effettuare una conversione manuale del dato, in modo da scongiurare una possibile perdita di informazione o un fallimento di conversione.

La seconda tipologia viene detta “Casting implicito” e con questa viene fatta una conversione senza perdite di informazioni.

1. **Quali sono gli utilizzi della keyword static?**

I metodi statici vengono utilizzati per definire dei metodi che agiscono nel contesto della classe (ma non necessariamente sugli oggetti della classe stessa). Si possono anche definire delle classi statiche e in questi casi non è necessario istanziare degli oggetti di quel tipo e viene utilizzata quando, ad esempio, si vogliono utilizzare dei metodi di I/O che non operano sugli oggetti.

1. **Descrivere le modalità di implementazione di un evento in C#**

Un evento serve ad attivare la comunicazione tra l’interno e l’esterno di una classe.

Nell’evento è possibile invidurare un publisher e una lista di subscriber che ricevono delle notifiche dal primo. Ogni subscriber può essere aggiunto alla lista del publisher (subscribe) oppure essere rimosso (unsubscribe).

Sia il subscripe che l’unsubscribe si avvalgono di un overload degli operatori, nel primo caso viene utilizzato il += e nel secondo il -=.

Per gestire questi due eventi ci sono due metodi: creare un delegate che ‘solleva l’evento’ e lo istanzia nel program, oppure sfruttare un altro delegate già implementato in c# che sgrutta l’EventHandler.

1. **A cosa serve l’interfaccia IEnumerable<T>? Come si implementa in una nostra classe?**

IEnumerable è l'interfaccia di base per tutte le raccolte non generiche che possono essere enumerate.

Con il suo utilizzo viene implementato il metodo GetEnumerator.

**Esercitazione Pratica**

* Realizzare una classe Warehouse per gestire un Magazzino Merci, con le seguenti proprietà:
  + *Id Magazzino (GUID)*
  + *Indirizzo*
  + *Importo Totale Merci in giacenza*
  + *Data Ultima Operazione*
  + *Lista delle Merci in giacenza*
* Realizzare **l’overload degli operatori + e –** in modo che sia possibile aggiungere e rimuovere Merci dalla lista (l’overload dovrà anche occuparsi di aggiornare l’Importo Totale e la Data di Ultima Operazione)
* Realizzare un metodo StockList() che stampi i dati del Magazzino, inclusa la lista delle Merci in giacenza
* Realizzare una gerarchia di classi per rappresentare le Merci (Good). Tutte le classi avranno le proprietà
  + *Codice Merce*
  + *Descrizione*
  + *Prezzo*
  + *Data di Ricevimento*
  + *Quantità in Giacenza*
  + Realizzare le classi che rappresentano:
    - ElectronicGood, con la proprietà aggiuntiva *Produttore*
    - PerishableGood, con le proprietà aggiuntive *Data di Scadenza* e *Modalità di Conservazione* (enum con i valori FREEZER, FRIDGE e SHELF)
    - SpiritDrinkGood, con le proprietà aggiuntive *Tipo* (enum con i valori WHISKY, WODKA, GRAPPA, GIN e OTHER) e *Gradazione Alcoolica*
  + Tutte le classi saranno dotate di costruttore che accetti tutti i parametri necessari per popolare le proprietà
  + Tutte le classi dovranno implementare la propria versione del metodo ToString() e visualizzare tutti i dati
  + ***OPZIONALE***: realizzare una procedura di caricamento dati da un file della lista delle Merci in giacenza (l’implementazione di eventi per notificare le fasi di caricamento dati costituisce un bonus)
  + ***OPZIONALE 2***: utilizzare una (o più) eccezione custom per gestire tutte le tipologie di errori che dovessero verificarsi durante l’utilizzo della classe Warehouse e delle classi Good
* Realizzare una Console app che
  + Crei un nuovo Magazzino
  + Permetta di ricevere diverse tipologie di Merci (gestire l’input dall'utente)
  + Stampi i dati del Magazzino e le Merci in giacenza