# Esercitazione di Fine Settimana – Week 2

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Maria Chiara |
|  |  | Cognome | Colla |
|  |  | Data | 02/04/2021 |

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 ritornare più di un valore da un metodo in C# è possibile utilizzare:   
   - la keyword out, che ritorna un valore aggiuntivo a quello del return o un valore per le funzioni void,  
   - ritornare un’istanza di una classe, facendo ritornare così tutti i suo campi e le proprietà,  
   - ritornare un’array, un generics o una collections per ritornare una collezione di valori,  
   - oppure ritornare una tupla, una collazione di valori che possono avere tipo diverso, con un massimo di 8 elementi, oppure utilizzare una tupla avente come elementi delle altre tuple per ritornare più valori.
2. Descrivere le due tipologie di casting tra tipi in C#  
   Il cast permette di convertire un tipo in un’altro tipo (in caso di incompatibilità verrà generata un’eccezione), in C# esistono due tipi di cast:  
   - Cast implicito: quando si passa da un tipo specifico a uno più generico, la conversione ha sempre successo e non si avrà una perdita di dati, non è richiesta una sintassi particolare (es. int a = 123456; long b = a;)  
   - Cast esplicito: quando si passa da un tipo generale a uno più specifico, la conversione potrebbe non avvenire o si potrebbe avere una perdita di dati, è richiesta una sintassi specifica: il valore che si vuole convertire deve essere preceduto dal tipo in cui si vuole convertirlo racchiuso tra le parentesi tonde   
   (es. double a = 10.29; int b = (int)a, in questo caso perderò la parte decimale del numero).  
   L’utente può anche definire delle sue operazioni di casting, sia implicito che esplicito, come parametro si mette il tipo da cui si vuole partire, nell’implementazione e come tipo di ritorno della funzione il tipo in cui si vuole convertirlo (devono avere un senso logico), nell’implementazione bisogna andare a compensare le caratteristiche che il tipo di partenza non ha rispetto al tipo di arrivo.
3. Quali sono gli utilizzi della keywork static?  
   La keyword ‘static’ può essere utilizzata per le classi, i metodi e le proprietà:  
   - Una classe viene definita statica quando non deve essere istanziata, ma sarà solo possibile usare i suoi metodi o le sue proprietà utilizzando il nome della classe stessa,  
   - Un metodo viene definito statico quando per chiamarlo bisogna utilizzare il nome della classe.nomeMetodo, senza dover istanziare un oggetto di quella classe,  
   - Una proprietà è statica quando non dipende dall’istanza, per esempio quando si ha un ID/Codice numerico e lo si vuole aumentare per ogni istanza creata della classe, solitamente si utilizza un campo statico per poter tenere aggiornato l’ID/Codice per tutte le istanze.
4. Descrivere le differenze tra le interfacce utilizzabili per le collection  
   Di interfacce di Collection ce ne sono tante e ognuna con i propri metodi:  
   - IEnumerable: è l’interfaccia più generale, consento l’uso del foreach (la possibilità di accedere all’elemento successivo),  
   - IList: da la possibilità di utilizzare i metodi: Insert, RemoveAt, perchè ha le posizione degli elementi ben definiti ed è possibile accedervi tramite indici,  
   - IDictionary: salva gli elementi in forma chiave-valore, ogni chiave deve essere unica e può essere associata a un solo elemento,  
   - ILookUP: salva come idictionary gli elementi in chiave-valore ma permette di avere più elementi per una sola chiave,  
   - ICollection: introduce l’uso di Add, Remove, Clear, Count,  
   -ISet: rappresente un vero e proprio insieme, per questo non accetta duplicati, e può eseguire operazioni insiemistiche tipo: unione e intersezione,  
   -IComparer: permette di comparare i tipi della collezione per poterne stabilire un ordine preciso,  
   - IEqualityComparer: simile a icomparer ma compara le chiavi degli elementi.
5. Cos’è un Generic?  
   Sono delle collezioni infinite di elementi che possono avere tipi diversi, consentono di scrivere classi e metodi indipendenti dal tipo.  
   Permettono di inserire elementi di tipi diversi per evitare l’uso di object che creerebbe delle situazioni non sicure, questo è possibile perchè il compilatore in automatico riconosce il tipo dell’elemento inserito.  
   è possibile creare una nuova classe generica, inserendoci anche delle regole relative al tipo di generics che si vuole utilizzare, con la possibilità di ereditare da un’altra classe e/o delle interfacce.  
   è possibile anche creare dei metodi generici specifici per una classe.  
   i generics non possono assumere dei valori nulli, al massimo si utilizza la keyword default.

**Esercitazione Pratica**

* Realizzare una classe Account per gestire un conto bancario, con le seguenti proprietà:
  + *Numero di Conto*
  + *Nome della Banca*
  + *Saldo*
  + *Data Ultima Operazione*
  + *Lista di Movimenti*
* Realizzare l’overload degli operatori + e – in modo che sia possibile aggiungere movimenti attivi e passivi alla lista dei movimenti (l’overload dovrà anche occuparsi di aggiornare il Saldo e la Data di Ultima Operazione)
* Realizzare un metodo Statement() che stampi i dati del conto, inclusa la lista dei movimenti
* Realizzare una gerarchia di classi per rappresentare movimenti bancari (Movement). Tutte le classi avranno le proprietà
  + *Importo*
  + *Data del Movimento*
  + Realizzare le classi che rappresentano:
    - CashMovement, con la proprietà aggiuntiva *Esecutore*
    - TransfertMovement, con le proprietà aggiuntive *Banca d’Origine* e *Banca Destinazione*
    - CreditCardMovement, con le proprietà aggiuntive *Tipo* (enum con i valori AMEX, VISA, MASTERCARD, OTHER) e *Numero di Carta*
  + 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
* Realizzare una Console app che
  + Crei un nuovo Account
  + Permetta di inserire diversi tipi di Movimenti (input dall'utente)
  + Stampi i dati del conto e i movimenti