# Esercitazione di Fine Settimana – Week 1

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Catalina |
|  |  | Cognome | Dorneanu |
|  |  | Data | 09/07/2021 |

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

1. Quali sono i 3 tipi di runtime disponibili nella piattaforma .NET? Cosa è successo col rilascio di .NET 5?

**.Net** si basa su Application Models suddivisi in tre gruppi di tecnologie, .Net Framework, che include WPF, Windows Forms e ASP.NET, per app solo su Window; poi .Net Core, che include UWP e ASP.NET Core, open-source e cross-platform; infine Xamarin, per iOS, Android e OS X, per lo sviluppo di App community e giochi.

Essi si basano sulla libreria comune .Net Standard Library, cioè un insieme di API che le piattaforme .Net devono implementare.

Questa a sua volta si basa su infrastruttura comune standard composta da compilatori, linguaggi e componenti runtime. Infatti il codice sorgente a questo livello viene convertito in Intermediate Language e in fase di esecuzione viene compilato in istruzioni per la CPU che successivamente le esegue.

Nel novembre del 2020 è uscito .**Net 5,** che non divideva più le funzionalità inbase al sistema operativo ma le ha rese disponibili per tutti (cross-platform) creando un’applicazione per ogni scopo. tranne quella mobile al momento.

1. Quale è la funzione del Garbage Collector?

Il **Garbage Collector** è uno strumento che ha lo scopo di liberare la memoria dalle risorse allocate ma non più utilizzate. Dato che la memoria Stack viene liberata automaticamente alla fine dei vari scope, il Garbace Collector si occupa della liberazione della memoria del Managed Heap, cioè dove sono salvati i reference type.

1. Descrivere la differenza tra Value Type e Reference Type

I **Reference Type** contengono solo un riferimento all’oggetto (indirizzo cella), nell’ambito dell’heap, e può essere null. Inoltre le modifiche sui reference modificano l’oggetto a cui puntano, e la copia di un reference type consiste nella copia solo della referenza e non dell’oggetto. Un esempio di Refernce Type sono le classi e string.

Invece, i **Value Type**, a differenza dei Refence Type, sono mantenuti nello stack e contengono direttamente il dato, e non ammettono direttamente il null. Inoltre le modifiche sui Value Type valgono solo sull’istanza corrente, e una copia consiste nella copia dei cadi contenuti nel Value Type. Essi sono i tipi primitivi come int, bool ecc, e gli enum.

1. Cos’è un Delegate?

Un delegate è l’equivalente .Net dei puntatori a funzione del C/C++, ma a differenza di quest’ultimi sono tipizzati. Quando si crea un'istanza di un delegate, è possibile associare l'istanza a qualsiasi metodo con una firma compatibile e un tipo restituito. I delegate vengono utilizzati per passare metodi come parametri ad altri metodi.

Una dichiarazione di un delegate definisce, mediante compilatore, una classe che deriva dalla classe System.Delegate o System.MulticastDelegate.

1. Descrivere l’uso delle keyword async / await

Il modello di programmazione asincrona, fornisce un'astrazione sul codice asincrono. Il codice viene scritto in maniera sequenziale come al solito e può essere letto come se ogni istruzione fosse completata prima che inizi la successiva.

L’obiettivo di questa sintassi è abilitare il codice che si legge come una sequenza di istruzioni, ma che viene eseguito in un ordine molto più complicato in base all'allocazione delle risorse esterne e al completamento delle attività.

La keyword **async** trasforma un metodo in un metodo asincrono, che consente di utilizzare la keyword **await** nel suo corpo. **Await** attende un’operazione asincrona e restituisce il controllo al chiamante fino al completamento dell'attività attesa. **Await** può essere utilizzato solo all'interno di un metodo asincrono.

1. Data una lista di istanze della classe Votazioni,

public class Votazioni

{

public string Materia { get; set; }

public string Studente { get; set; }

public DateTime Data { get; set; }

public int Voto { get; set; }

}

scrivere una query LINQ (Fluent oppure Query Expression) che restituisca un elenco di materie con il voto medio, quello più alto e quello più basso per ciascuna

1. Descrivere il Factory pattern e perché utilizzarlo

Un design pattern descrive una soluzione generale ad una problematica ricorrente. I design pattern si distinguono in tre categorie: creazionali (vengono utilizzati durante la creazione della classi), strutturali (stabiliscono la struttura delle relazioni tra i vari oggetti) e comportamentali(definiscono il comportamento di ogni istanza del programma).

Il Design Pattern Factory è un pattern di tipo creazionale e descrive una soluzione generale per la creazione di oggetti in base a diverse caratteristiche, nascondendo la gerarchia del quale fanno parte. Infatti vi è una classe intermedia Factory, alla quale si interfaccia il client, e che si occupa della creazione concreta dell’oggetto accedendo all’interfaccia pubblica della gerarchia.

**Esercitazione Pratica**

Realizzare una Console app (C#) che:

* Effettui il monitoraggio di una cartella in attesa di un file di testo con l'elenco delle spese (*spese.txt*)
* Apra e legga il file. Ogni riga è nel formato:



* Per ogni riga, determini se la spesa è approvata. Esistono diversi livelli di approvazione, a seconda dell'import della spesa
  + **Manager**: spese fino a 400€
  + **Operational Manager**: da 401€ fino a 1000€
  + **CEO**: sopra i 1000€
  + Nessuna spesa sopra i 2500€ è approvata

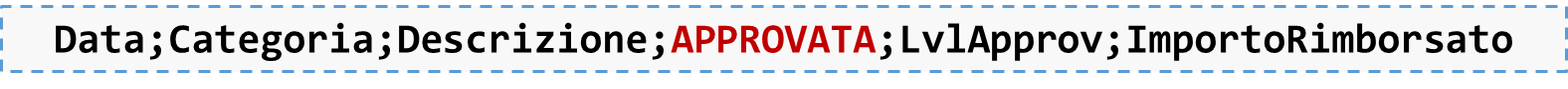
**Usare il Chain of Responsibility pattern**

**(restituire il livello di approvazione)**

* Per ogni spesa approvata, determini l'importo rimborsato sulla base della Categoria
  + **Viaggio**: 100% dell'importo + 50€ fisse
  + **Alloggio**: 100% dell'importo
  + **Vitto**: 70% dell'importo
  + **Altro**: 10% dell'importo

**Usare il Factory pattern**

* Salvi poi le informazioni sulle spese rimborsate e non rimborsate in un file di testo (*spese\_elaborate.txt*)
* Per ogni spesa rimborsata salvare una riga nel formato



* Per ogni spesa non rimborsata salvare una riga nel formato

