# Modulo 1

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Maria Chiara |
|  |  | Cognome | Colla |
|  |  | Data | 26/03/2021 |

Leggete attentamente ogni domanda e argomentare quanto più possibile **fornendo anche degli esempi**.  
ATTENZIONE: Le domande a risposta multipla possono contenere più risposte corrette.

1. *Dare una panoramica degli elementi principali nel Framework .Net. Cosa è successo con .Net 5?  
   Framework .Net è solo per windows, contiene asp.net, wpf e windows form, ma c’era già l’idea di creare qualcosa disponibile per tutti i sistemi operativi.  
   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.*
2. *Definire i principi su cui si basa il linguaggio C#  
   C# è un linguaggio di programmazione object oriented, ovvero si basa sulla creazione di classi con campi, proprietà, metodi e le istanze di queste classi prendono il nome di oggetto.  
   la programmazione orientata agli oggetti si basa su tre pilastri fondamentali:  
   - Incapusulamento: ogni proprietà o metodo deve avere un modificatore di accessibilità per poterne modificare la visibilità nell’intero programma, possono essere: public: visibili da tutti all’interno del programma, private: visibili solo all’interno della classe, protected: visibili solo all’interno della classe o dalle classi derivate, internal: visibili solo all’interno dell’assembly, internal-protected: visibili solo all’interno dell’assembly, della classe e dalle classi derivate.  
   - Ereditarieà: esiste quando due classi hanno una relazione, la classe derivata eredita tutti i campi, le proprietà e i metodi dalla classe base(tranne quelli privati), in C# una classe può ereditare da una sola classe, non esiste l’ereditarietà multipla, in caso di necessità si possono usare le interfacce per cui è l’ereditarietà multipla è concessa.  
   - Polimorfismo: esiste quando c’è una relazione di ereditarietà, se la classe base è definita abstract, la classe derivata deve per forza implementare i metodi abstract(se a sua volta viene derivata da un’altra classe può mantenere i metodi abstract e farli implementare alla classe derivata), se i metodi sono marcati virtual potrà sovvrascriverli, in entrambi i casi i metodi nella classe derivata dovrenno essere marcati override.  
   è un linguaggio fortemente tipizzato(ogni variabile deve avere un tipo) ed è in grado di gestire le eccezzioni con il blocco try catch: try contiene il pezzo di codice che potrebbe causare l’eccezzione, catch cattura queste eccezioni e le gestisce, ci possono essere più blocche catch che gestiscono dall’eccezione più specifica a quella più generale ovvere exception che il padre di tutte le eccezioni e per questo le comprende tutte.*
3. *Dare una spiegazione di Garbage Collector.  
   Serve per liberare quando la memoria heap quandi è piena, seleziona tutte le celle, deseleziona quelle che sono ancora in uso, cancella quelle che sono rimaste selezionate e compatta la memoria.  
   l’heap è diviso in tre sezioni dette Generation, la prima è la più piccola ed è chiamata generation zero, poi ci sono generation 1 e 2 che diventano sempre più grandi.tt*
4. *Descrivere le differenze tra Reference Type e Value Type  
   Value type contiene il dato che rappresenta, se viene copiato, si ha la copia del valore, nel caso in cui venga cambiato non modifica l’originale, viene salvato all’interno dello stack (a meno che non sia contenuto dento una classe, in quel caso viene salvato nell’heap), e non possono essere nulli, un esempio di value type sono i tipi primitivi: int, float, double, char e i tipi definiti da utente: enum e struct.  
   Il reference type contiene l’indirizzo della cella di metmoria in cui si trova il dato che rappresenta, se viene copiato si ha la copia dell’indirizzo, nel caso in cui viene modificato il dato allora si modificherà per tutti i reference type che lo puntano, viene sempre salvato nell’heap, possono essere nulli e per essere istanziati devo usare le porala chiave new(tranne la stringa che è un tipo particolare di reference type), un esempio di reference type: classi, string.*
5. *Cosa è un delegate?  
   i delegate sono dei puntatori a funzioni, il compilatore crea una classe con quel nome che deriva da System.Delegate (solo il compilatore può derivare da quelle classi).  
   il Delegate viene definito prima del Main o della classe, con: Delegate, tipo di ritorno, nome e i parametri, poi viene definito all’interno di una classe come: public static, nuovo nome, stesso tipo e numero di parametri e la sua implementazione, in fine per essere usato viene istanziato un nuovo oggetto del tipo dichiarato con delegate e come parametro il nome della classe con l’implementazione.*
6. *Scrivere una query LINQ : Date le gare disponibili, ricavare il tempo medio, massimo e minimo per disciplina.  
   var tempoMassimo = gare.OrderBy(gara => gara.Disciplina)  
    .Max(gara => gara.TempoImpiegato)*

*var tempoMinimo = gare.OrderBy(gara => gara.Disciplina)  
 .Min(gara => gara.TempoImpiegato)*

*var tempoMedia = gare.OrderBy(gara => gara.Disciplina)  
 .Avg(gara => gara.TempoImpiegato)*

1. *Definire cosa è il design Pattern Factory  
   I design pattern descrivono una soluzione generale a problemi ricorrenti, il desing pattern factory consente di creare oggetti sulla base di determinati criteri.  
   è composta da un’interfaccia implementata da diverse classi concrete, tutte gestiste da una classe concreta, solitamente chiamata factory, che in base all’input richiesto è in grado di creare l’istanza della classe relativa.*

*Mettere la prova pratica e teorica su Github.*