# Modulo 1

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Valentyna |
|  |  | Cognome | Bogachova |
|  |  | 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?*
2. *Definire i principi su cui si basa il linguaggio C#*
3. *Dare una spiegazione di Garbage Collector.*
4. *Descrivere le differenze tra Reference Type e Value Type*
5. *Cosa è un delegate?*
6. *Scrivere una query LINQ : Date le gare disponibili, ricavare il tempo medio, massimo e minimo per disciplina.*
7. *Definire cosa è il design Pattern Factory*

1. Net Framework ,modelli di app che funzionano solomento con il sistemo operative Windows e un po vecchio

.Net Core é cross platform(nato nel 2015) (Windows,Linux,Macos)( Net Framework e .Net Core si fanno le app per il desktop)

XAMARIN (2004) per le app mobile e Unity(games)

Sudette technology si basano su .Net Standart Library e implementono e indicono una certa compatibilità, e si basa sulla infrastruttura commune (Common Language Runtime, Language Specification, Garbage Collector)

.NET 5 è appena uscito si sta andando verso Net 5, non ho suddivisione tra sudette tecnologia, ho un unico insieme di tecnologie cross platform

2.C# è linguaggio orientato ad oggetti ma anche fortemente di tipizzato che si dividono Value type e Reference Type, si appoggia sulle classi e interface e implementa concetti fondamentale incapsulamento, ereditarietà e polimorfismo, gestione delle eccezioni, gestione degli eventi e delegates, lettura escrittura dei file, multithreading, LINQ e Lambda.

3. . Garbage Collection è una modalità automatica di gestione della memoria in cui il sistema operativo o il compilatore e il modulo di runtime recuperano porzioni di memoria che non sono più utilizzate dalle applicazioni

.

4. **Value type**: Quando la variabile contiene i dati nella propria allocazione di memoria si riferisce ad un Value type. La variabile infatti memorizza i dati nello stack e quando esce dal campo di applicazione, dopo la procedura per cui è stata definita è terminata, il suo valore viene eliminato dallo stack. Questo tipo è preferibile per q l’ occupazione di memoria. Fra i tipi di valore sono inclusi tutti i tipi di dati numerici ,Boolean, Char e Date

**Reference type**: La variabile contiene un puntatore a un’altra posizione di memoria contenente i dati chiamata heap. La memoria occupata dalla variabile non viene riazzerata come accade con una Value Type, ma resta in memoria fino al momento in cui il sistema decide che non sia più necessaria. Quando la variabile contiene i dati nella propria allocazione di memoria si riferisce ad un Value type. Parole-chiave riservate come New vanno utilizzate per dichiarare una Reference Type.Fra i tipi di riferimento sono inclusi i String,. La differenza sostanziale quindi tra Value type e Reference type è che i tipi di riferimento portano con se informazioni in fase di esecuzione mentre i tipi di valore no.

5. Un delegate è un tipo che rappresenta riferimenti ai metodi con un elenco di parametri e un tipo restituito particolari. Quando si crea un'istanza di un delegato, è possibile associare l'istanza a qualsiasi metodo con una firma compatibile e un tipo restituito. Tramite l'istanza di delegato è possibile richiamare (o chiamare) il metodo. I delegati vengono utilizzati per passare metodi come argomenti ad altri metodi. I gestori di evento non sono altro che metodi richiamati tramite delegati.

6. Var tempi = gare.GroupBy( g => g.disciplina)  
.Select(tempo => new {  
Disciplina = tempo.Key,  
TempoMedio = tempo.Avarage( d => d.tempoImpiegato),

TempoMax = tempo.Max(d => d.tempoImpiegato),  
TempoMin = tempo.Min(d => d.tempoImpiegato)  
});

7. l Factory Method è uno dei design pattern fondamentali per l'implementazione del concetto di factory. Come altri pattern creazionali, indirizza il problema della creazione di oggetti senza specificarne l'esatta classe. Questo pattern raggiunge il suo scopo fornendo un'interfaccia per creare un oggetto, ma lascia che le sottoclassi decidano quale oggetto istanziare.