**Part 1 - Stack, Heap, Boxing, Unboxing, Array, ArrayList, Generics, Threading**  
Question 1 :- Explain difference between .NET and C# ?  
Question 2 :- .NET Framework vs .NET Core vs .NET 5.0  
Question 3 :- What is IL ( Intermediate Language) Code ?  
Question 4 :- What is the use of JIT ( Just in time compiler) ?  
Question 5 :- Is it possible to view IL code ?  
Question 6 :- What is the benefit of compiling in to IL code ?  
Question 7 :- Does .NET support multiple programming languages ?  
Question 8 :- What is CLR ( Common Language Runtime) ?  
Question 9 :- What is managed and unmanaged code ?  
Question 10 :- Explain the importance of Garbage collector ?  
Question 11 :- Can garbage collector claim unmanaged objects ?  
Question 12 :- What is the importance of CTS ?  
Question 13 :- Explain CLS ?  
Question 14 :- Difference between Stack vs Heap ?  
Question 15 :- What are Value types & Reference types?  
Question 16 :- Explain boxing and unboxing ?  
Question 17 :- What is consequence of boxing and unboxing ?  
Question 18 :- Explain casting, implicit casting and explicit casting ?  
Question 19 :- What can happen during explicit casting ?  
Question 20 :- Differentiate between Array and ArrayList ?  
Question 21 :- Whose performance is better array or arraylist ?  
Question 22 :- What are generic collections ?  
Question 23 :- What are threads (Multithreading)?  
Question 24 :- How are threads different from TPL ?  
Question 25 :- How do we handle exceptions in C#(try/catch)?  
Question 26 :- What is the need of finally?  
Question 27 :- Why do we need the out keyword ?  
Question 28 :- What is the need of Delegates ?  
Question 29 :- What are events ?  
Question 30 :- What's the difference between Abstract class and interface ?

**Part 2 - Questions on Delegates, Event and Delegates vs Events.**  
Question 31  :- What is a delegate and How to create a delegate ?  
Question 32  :- Where have you used delegates ?  
Question 33  :- What is a Multicast delegates ?  
Question 34  :- What is a Event ?  
Question 35  :- How to create a event ?  
Question 36  :- Delegate vs Events.

**Part 3 - OOP, Abstraction, Encapsulation, Inheritance, Overriding & overloading.**  
Question 37 :- Why do we need OOP ?  
Question 38 :- What are the important pillars of OOPs ?  
Question 39 :- What is a class and object ?  
Question 40 :- Abstraction vs Encapsulation?  
Question 41 :- Explain Inheritance ?  
Question 42 :- Explain virtual keyword ?  
Question 43 :- What is overriding ?  
Question 44 :- Explain overloading ?  
Question 45 :- Overloading vs Overriding ?

**Part 4 - Polymorphism, Static vs Dynamic polymorphism and operator overloading.**  
Question 46 :- What is polymorphism ?  
Question 47 :- Can polymorphism work with out inheritance ?  
Question 48 :- Explain static vs dynamic polymorphism ?  
Question 49 :- Explain operator overloading ?

**Part 5 - Tricky Questions around Abstract classes and Interfaces.**  
Question 50  :- Why do we need Abstract classes ?  
Question 51  :- Are Abstract methods virtual ?  
Question 52  :- Can we create a instance of Abstract classes ?  
Question 53  :- Is it compulsory to implement Abstract methods ?  
Question 54  :- Why simple base class replace Abstract class ?  
Question 55  :- Explain interfaces and why do we need it ?  
Question 56  :- Can we write logic in interface ?  
Question 57  :- Can we define methods as private in interface ?  
Question 58  :- If i want to change interface what's the best practice ?  
Question 59  :- Explain Multiple inheritance in Interface ?  
Question 60  :- Explain Interface Segregation principle ?  
Question 61  :- Can we create instance of interface ?  
Question 62  :- Can we do Multiple inheritance with Abstract classes ?

**Part 6 - Answering the most asked Question "Abstract classes vs Interface".**  
Question 63 :- Difference between Abstract Class & Interfaces?

**Part 7 - Questions around constructors & parent child constructor.**  
Question 64  :- Why do we need constructors ?  
Question 65  :- In parent child which constructor fires first ?  
Question 66  :- How are initializers executed ?  
Question 67  :- How are static constructors executed in Parent child ?  
Question 68  :- When does static constructor fires ?

**Part 8 - Questions around Shadowing, Sealed, Nested classes and partial classes.**  
Question 69 :- What is Shadowing?  
Question 70 :- Explain method hiding?  
Question 71 :- Shadowing vs Overriding ?  
Question 72 :- When do we need Shadowing ?  
Question 73 :- Explain Sealed Classes ?  
Question 74 :- Can we create instance of sealed classes ?  
Question 75 :- What are nested classes and when to use them ?  
Question 76 :- Can Nested class access outer class variables ?  
Question 77 :- Can we have public, protected access modifiers in nested class ?  
Question 78 :- Explain Partial classes ?  
Question 79 :- In What scenarios do we use partial classes ?

**Part 9 - Questions Around SOLID principles , Dependency injection (DI) and IOC**  
Question 80 :- What is SOLID ?  
Question 81 :- What is the full form of SOLID ?  
Question 82 :- What is the goal of SOLID ?  
Question 83 :- Explain SRP with A example ?  
Question 84 :- What is the benefit of SRP ?  
Question 85 :- Explain OCP with a example ?  
Question 86 :- What is the  benefit of OCP ?  
Question 87 :- Can you explain LISKOV Principle and it's violation?  
Question 88 :- How can we fix LISKOV Problem ?  
Question 89 :- Explain Interface Segregation Principle ?  
Question 90:- Is there a connection between LISKOV and ISP ?  
Question 91 :- Define dependency inversion ?  
Question 92 :- What is higher level module and lower level module ?  
Question 93 :- How does dependency inversion benefit, show with an example ?  
Question 94 :- Will only Dependency inversion solve decoupling problem ?  
Question 95 :- Why do developers  move object creation outside high lever module ?  
Question 96 :- Explain IOC ( Inversion of Control) ?  
Question 97 :- Explain Dependency Injection with an example ?  
Question 98 :- Is SOLID, IOC and DI design pattern or Principle?  
Question 99 :- Is only SOLID Enough for good code/ architecture ?

**Part 10- Explain & Differentiate Composition, Aggregation and Association in C#.**  
Question 100 :- What are the different types of "USING/HAS A" relationship ?  
Question 101 :- What is a composition relationship ?  
Question 102 :- Explain Aggregation ?  
Question 103 :- Explain Association ?  
Question 104 :- Differentiate between Composition vs Aggregation vs Association ?  
Question 105 :- UML Symbols for Composition, Aggregation and Association

**Part 11 - Crack questions on Stack, Heap, Boxing, Unboxing, Value &  reference types**  
Question 106 :- Explain stack and Heap ?  
Question 107 :- Where are stack and heap stored ?  
Question 108 :- What goes on stack and what goes on heap ?  
Question 109:- How is the stack memory address arranged ?  
Question 110 :- How is stack memory deallocated LIFO or FIFO ?  
Question 111 :- How are primitive and objects stored in memory?  
Question 112 :- Can primitive data types be stored in heap ?  
Question 113 :- Explain value types and reference types ?  
Question 114 :- Explain byval and byref ?  
Question 115 :- Differentiate between copy byvalue and copy byref ?  
Question 116 :- What is boxing and unboxing ?  
Question 117 :- Is boxing unboxing good or bad ?  
Question 118 :- Can we avoid boxing and unboxing ?  
Question 119 :- What effect does boxing and unboxing have on performance ?  
Question 120 :- Are string allocated on stack or heap ?  
Question 121 :- How many stack and heaps are created for an application ?  
Question 122 :- How are stack and heap memory deallocated ?  
Question 123 :- Who clears the heap memory ?  
Question 124 :- Where is structure allocated Stack or Heap ?  
Question 125 :- Are structures copy byval or copy byref ?  
Question 126 :- Can structures get created on Heap ?

**Part 12 - What is Garbage collector, Managed vs UnManaged code, Dispose Pattern, Memory Leaks, weak VS strong references ?**

Question 127: - Explain Garbage collector (GC)?  
Question 128:- How does Garbage collector know when to clean the objects ?  
Question 129 :- Is there a way we can see this Heap memory ?  
Question 130 :- Does Garbage collector clean primitive types ?  
Question 131: - Managed vs UnManaged code/objects/resources?  
Question 132:- Can garbage collector clean unmanaged code ?  
Question 133:- Explain Generations  ?  
Question 134:- What is GC0,GC1, and  GC2 ?  
Question 135:- Why do we need Generations ?  
Question 136:- Which is the best place to clean unmanaged objects ?  
Question 137:- How does GC behave when we have a destructor ?  
Question 138:- What do you think about empty destructor ?  
Question 139:- Explain the Dispose Pattern?  
Question 140 :- Finalize vs Destructor ?  
Question 141:- What is the use of using keyword ?  
Question 142:- Can you force Garbage collector ?  
Question 143:- Is it a good practice to force GC ?  
Question 144:- How can we detect a memory issues ?  
Question 145:- How can we know the exact source of memory issues ?  
Question 146 :- What is a memory leak ?  
Question 147 :- Can .NET Application have memory leak as we have GC?  
Question 148:- How to detect memory leaks in .NET applications ?  
Question 149:- Explain weak and strong references ?  
Question 150 :- When will you use weak references ?

**Lesson 13 :- Questions around Design Pattern Basics, Types, Singleton Pattern, Prototype, Template and Adapter.**

Question 151:- What are design patterns?  
Question 152 :- Which are the different types of design patterns?  
Question 153 :- Explain structural , Behavioral and Creational design pattern ?  
Question 154:- Explain Singleton Pattern and the use of the same?  
Question 155:- How did you implement singleton pattern?  
Question 156:- Can we use Static class rather than using a private constructor?  
Question 157:- Static vs Singleton pattern?  
Question 158:- How did you implement thread safety in Singleton?  
Question 159:- What is double null check in Singleton?  
Question 160:- Can Singleton pattern code be made easy with Lazy keyword?  
Question 161:- Can we rid of this double null check code?

**Lesson 14 :- Repository Pattern and Unit of Work Design Pattern Interview Questions.**

Question 162:- What is the use of repository pattern?  
Question 163:- Is Dal (Data access Layer) and Repository same?   
Question 164:- What is Generic repository pattern ?  
Question 165:- Is abstraction the only benefit of Repository?  
Question 166:- How to implement transaction in repository?  
Question 167:- What is Unit of work design pattern?  
Question 168:- Do we need repository pattern as EF does almost the same work?  
Question 169:- Did you do unit testing with Repository ?  
Question 170:- How does repository pattern make unit testing easy?  
Question 171:- How can we do mock testing with Repository?

**Lesson 15:- Most asked Factory Pattern, DI and IOC Interview Questions.**

Question 172 :- What is Factory pattern and how does it benefit?  
Question 173 :- How does centralizing object creation helps in loose coupling ?  
Question 174 :- What is IOC and DI ?  
Question 175 :- DI vs IOC ?  
Question 176 :- What is a service locator ?  
Question 177:- Service Locator vs DI ?  
Question 178 :- Which is good to use Service Locator or DI ?  
Question 179 :- Can not we use a simple class rather than interface for DI ?  
Question 180 :- Is DI a Factory Pattern?  
Question 181 :- So If you just centralize object creation is it Factory pattern?  
Question 182 :- Static DI and Dynamic DI ?  
Question 183 :- In which scenarios to use Static DI vs Dynamic DI ?

**Lesson 16 :- The Real Factory and Abstract Factory Patterns.**

Question 184 :- The real Factory pattern ?  
Question 185 :- Factory Method vs Factory pattern ?  
Question 186 :- How are new behaviors created in FP ?  
Question 187 :- What is Abstract Factory Pattern ?  
Question 188 :- Does Abstract Factory Pattern use FP inside ?  
Question 189 :- Explain Simple Factory Pattern ?  
Question 190 :- Simple Factory vs Factory (Factory Method) vs Abstract Factory ?  
Question 191 :- How to remove IF conditions from Simple Factory?