Class assignment of .NET Framework using C# (MCAE0402)

1. Imagine you are explaining the .NET framework architecture to a colleague unfamiliar with the framework. How would you break down the architecture and its components, such as the CLR, FCL, and the application domains? Provide a structured explanation.
2. In a team meeting, you are asked to explain key .NET framework runtime concepts like the Common Language Runtime (CLR), Common Type System (CTS), and Common Language Specification (CLS). How would you present these to ensure clarity and relevance to the team's work?
3. You are developing a large-scale application and need to explain to a junior developer how assemblies are used in .NET framework to organize and deploy the application. Provide an explanation of assemblies and include an example scenario where multiple assemblies are used.
4. In your project, you notice a developer struggling to organize classes and methods properly. How would you explain the concept of namespaces in .NET framework and demonstrate how they are used to avoid naming conflicts in large projects?
5. During a code review, a developer confuses primitive types with reference types in their application. How would you explain the difference between primitive types and reference types?
6. While refactoring a piece of C# code, you notice both value types and reference types are being used incorrectly. Explain the difference between value types and reference types in C#, and provide examples to clarify their behaviour in memory.
7. You are tasked with creating a method that demonstrates both implicit and explicit type conversions. Write a program in C# that converts an int to a double implicitly and a double to an int explicitly, explaining each step in your code.
8. A junior developer asks for help writing a program to determine whether a number is positive, negative, or zero. Use if-else statements to write this program in C#, and explain the logic behind the code.
9. You are explaining control flow constructs to a new hire. Use a switch-case construct to explain how it works in C#. Illustrate the use of this construct by writing a program that takes a number (1-5) and prints the corresponding weekday.
10. You are mentoring a developer on decision constructs in C#. Demonstrate how to use nested if-else and switch-case statements together by writing a program that checks a number and prints whether it is even/odd and whether it falls into specific ranges (e.g., 0-10, 11-20).
11. During a live coding session, you are asked to write a program that prints the Fibonacci series using a for loop in C#. Provide a detailed explanation of your approach, and explain how the loop is used to generate the series.
12. You are leading a training session on loops in C#. Explain the key differences between while and do-while loops, and provide examples of each where one might be more appropriate than the other.
13. You are developing a pattern generation tool for a project. Write a program in C# that uses nested loops to generate a pyramid pattern of stars (*). Explain how the loops work together to produce the pattern.

14. You are giving a presentation on object-oriented programming. Define Encapsulation, Inheritance, Polymorphism, and Abstraction, and provide real-world examples of each in the context of C# development.
15. In a team discussion, you are asked to demonstrate the use of constructors and destructors in C#. Write a C# program that includes both, explaining the lifecycle of an object from creation to destruction.
16. A team member is confused about access modifiers in C#. How would you explain public, private, protected, and internal modifiers, and demonstrate their use by writing a small C# class with methods using different access levels?
17. You are tasked with illustrating the concept of inheritance in C#. Write a program where a Vehicle class is inherited by a Car class and a Bike class, each with their own unique methods. Demonstrate how inheritance allows code reuse.
18. In a bug-fixing scenario, your team needs to handle unexpected runtime errors. Explain how the try-catch-finally blocks work in C# with an example of catching and handling an arithmetic exception, and how finally is always executed.
19. You are implementing a custom exception for a specific error scenario in your application. Write a C# program that demonstrates exception handling by throwing and catching a custom exception, explaining why custom exceptions are beneficial.
20. During a code quality meeting, you are asked to highlight the advantages of using exception handling in C#. Explain how proper exception handling improves application's robustness.