C# Interview Questions with Programming Examples (Advanced Topics)

31. How to create a thread in C#?
Answer:

```
using System.Threading;

void PrintNumbers() {
    for (int i = 1; i <= 5; i++)
        Console.WriteLine($"Thread: {i}");
}

Thread t = new Thread(PrintNumbers);
t.Start();
```

32. What is Task vs Thread in C#?
Answer:
- Task: Higher-level abstraction (uses thread pool).
- Thread: Lower-level, manually controlled.

```
await Task.Run(() => {
    Console.WriteLine("Running async task");
});
```

33. Explain Dependency Injection (DI) with an example.
Answer:

```
interface IService {
    void Serve();
}

class EmailService : IService {
    public void Serve() => Console.WriteLine("Email sent");
}

class Controller {
    private readonly IService _service;
    public Controller(IService service) {
        _service = service;
    }

    public void Process() => _service.Serve();
}

var service = new EmailService();
var controller = new Controller(service);
controller.Process();
```

34. How do you read/write files in C#?
Answer:

```
// Write
File.WriteAllText("output.txt", "Hello, file!");

// Read
string content = File.ReadAllText("output.txt");
```

```
Console.WriteLine(content);
```

35. Implement the Factory Pattern in C#.
Answer:

```
interface IShape {
    void Draw();
}

class Circle : IShape {
    public void Draw() => Console.WriteLine("Drawing Circle");
}

class ShapeFactory {
    public static IShape GetShape(string type) {
        return type switch {
            "circle" => new Circle(),
            _ => throw new Exception("Unknown shape")
        };
    }
}

var shape = ShapeFactory.GetShape("circle");
shape.Draw();
```

36. LINQ: Group by and count.
Answer:

```
var words = new List<string> { "cat", "dog", "cat", "bird", "dog" };
var grouped = words.GroupBy(w => w).Select(g => new { Word = g.Key, Count = g.Count()
});

foreach (var item in grouped)
    Console.WriteLine($"{item.Word}: {item.Count}");
```

37. Implement the Singleton Pattern.
Answer:

```
public class Singleton {
    private static readonly Singleton _instance = new Singleton();
    private Singleton() { }
    public static Singleton Instance => _instance;
}

var s1 = Singleton.Instance;
```

38. How do you use lock in multithreading?
Answer:

```
class Counter {
    private int _count;
    private readonly object _lock = new();

    public void Increment() {
        lock (_lock) {
            _count++;
        }
    }
```

```
}
```

39. LINQ: Top 2 highest numbers.
Answer:

```
var numbers = new List<int> { 1, 5, 7, 2, 9, 3 };
var topTwo = numbers.OrderByDescending(x => x).Take(2);

foreach (var num in topTwo)
    Console.WriteLine(num);
```

40. Repository Pattern.
Answer:

```
interface IProductRepository {
    IEnumerable<string> GetAll();
}

class ProductRepository : IProductRepository {
    public IEnumerable<string> GetAll() => new List<string> { "Pen", "Pencil" };
}

class ProductService {
    private readonly IProductRepository _repo;
    public ProductService(IProductRepository repo) {
        _repo = repo;
    }

    public void ShowProducts() {
        foreach (var item in _repo.GetAll())
            Console.WriteLine(item);
    }
}
```