# Roteiro de Estudo Completo de C# (Básico ao Avançado)

**1. Fundamentos de C# e .NET**  
- O que é C# e para que serve.  
- Estrutura básica de um programa C#.  
- Introdução ao .NET e .NET Core / .NET 5+.  
- Compilação e execução de programas C#.  
- Configuração de ambiente: Visual Studio, VS Code, SDK.

**2. Tipos de Dados e Operadores**  
- Tipos primitivos: int, float, double, bool, char, string.  
- Nullable types.  
- Variáveis e constantes.  
- Conversão de tipos: casting explícito e implícito.  
- Operadores: aritméticos, relacionais, lógicos, ternário.

**3. Controle de Fluxo**- Estruturas condicionais: if, else, switch.  
- Estruturas de repetição: for, while, do-while, foreach.  
- Controle de fluxo: break, continue, return.

**4. Métodos e Funções**  
- Declaração de métodos.  
- Parâmetros e retorno de valores.  
- Modificadores de parâmetros: ref, out, params.  
- Métodos estáticos e de instância.  
- Sobrecarga de métodos.

**5. Programação Orientada a Objetos (POO)**  
- Conceitos de Classe e Objeto.  
- Modificadores de Acesso: public, private, protected, internal.  
- Encapsulamento.  
- Propriedades (get, set) e Auto-Properties.  
- Construtores e Destrutores.  
- Herança e Polimorfismo.  
- Sobrescrita (override) e ocultação (new).  
- Classes e métodos abstratos.  
- Interfaces.  
- Delegação e composição.  
- Sealed e static classes.

**6. Coleções e Arrays**  
- Arrays unidimensionais e multidimensionais.  
- Listas genéricas (List<T>).  
- Dicionários (Dictionary<TKey, TValue>).  
- Conjuntos (HashSet<T>, SortedSet<T>).  
- Pilhas (Stack<T>) e filas (Queue<T>).  
- Métodos de manipulação de coleções: Add, Remove, Contains, etc.

**7. Manipulação de Strings e Data/Hora**  
- Classe String e métodos principais.  
- StringBuilder.  
- Formatação de strings (interpolação, string.Format).  
- Classes de Data e Hora: DateTime, TimeSpan.  
- Fuso horário (DateTimeOffset).

**8. Manipulação de Arquivos e Diretórios**  
- Namespaces: System.IO.  
- Leitura e escrita de arquivos de texto e binários.  
- Manipulação de diretórios.  
- Streams: FileStream, MemoryStream.

**9. Exceções e Tratamento de Erros**  
- Try, Catch, Finally.  
- Criação de exceções personalizadas.  
- Propagação de exceções (throw).  
- Bloco using para gerenciamento de recursos.

**10. Conceitos Avançados de POO**  
- Delegates.  
- Eventos.  
- Lambda Expressions.  
- Funções anônimas.  
- Expression-bodied members.  
- Extension Methods.

**11. LINQ (Language Integrated Query)**  
- Conceito e vantagens.  
- Sintaxe de consulta e sintaxe de método.  
- Operadores principais: Where, Select, OrderBy, GroupBy.  
- Projeções e agregações.  
- LINQ to Objects.  
- LINQ to Entities (para uso com Entity Framework).

**12. Programação Assíncrona e Paralela**  
- Conceitos: Thread, Task, Parallel.  
- async e await.  
- Task<T>, ValueTask.  
- CancellationToken.  
- Parallel.For, Parallel.ForEach.  
- Concorrência e sincronização.

**13. Programação Funcional em C#**  
- Imutabilidade.  
- Funções puras.  
- Delegates e lambdas como funções de primeira classe.  
- Func<> e Action<>.  
- Predicate<>.  
- Pattern Matching.  
- Tuplas e Desconstrução.

**14. Arquitetura e Design Patterns**  
- Principais Padrões de Projeto: Singleton, Factory, Repository, Dependency Injection, Strategy, Observer.  
- SOLID Principles.  
- Clean Code.  
- Domain Driven Design (DDD) – básico.

**15. Desenvolvimento com .NET**  
- Estrutura de projetos em .NET.  
- Gerenciamento de pacotes com NuGet.  
- Criação de bibliotecas e reutilização de código.  
- Criação de APIs com ASP.NET Core.  
- Criação de aplicações desktop com Windows Forms e WPF.  
- Criação de aplicações console.  
- Web APIs com REST.

**16. Entity Framework Core**  
- ORM: o que é e vantagens.  
- Configuração de contexto.  
- Mapeamento de entidades.  
- Migrations.  
- CRUD com EF Core.  
- Consultas com LINQ.

**17. Testes em C#**  
- Testes Unitários: xUnit, MSTest, NUnit.  
- Mocking: Moq.  
- Test Driven Development (TDD).  
- Testes de Integração.

**18. Segurança**  
- Boas práticas de segurança no código C#.  
- Validação de dados.  
- Criptografia: System.Security.Cryptography.  
- Autenticação e autorização com ASP.NET Core Identity.

**19. Deployment e DevOps**  
- Empacotamento de aplicações.  
- Publicação no Azure, AWS ou serviços on-premises.  
- Configuração de CI/CD com GitHub Actions ou Azure DevOps.  
- Docker e C#.

**20. Preparação para Certificação**  
- Revisão de conceitos principais.  
- Simulados de certificação.  
- Projetos práticos completos.  
- Estudo das documentações oficiais.  
- Participação em fóruns: Stack Overflow, GitHub.

**21. Boas Práticas e Recursos Extras**  
- Documentação de código (XML Comments).  
- Logs e monitoramento com Serilog ou NLog.  
- Performance e profiling.  
- Análise estática de código.