Exercício de Fixação 7 — Módulo 1 (POO com Classes Simples)

Objetivos

- Aprofundar os conceitos de Scrum e agilidade com foco em adaptação e valor
- Praticar criação e manipulação de classes com C#
- Resolver problemas com lógica, validação e controle de fluxo
- Desenvolver clareza na modelagem e uso de métodos em POO

> **Nota mínima para aprovação: 8,5 pontos (85%)**

PARTE 1 — Exercícios Teóricos

1.1 — Questões Objetivas: Scrum (0,25 ponto cada)

1. Qual é o principal objetivo da Sprint Review no Scrum?

- [ ] A. Apresentar métricas de desempenho do time
- [ ] B. Redefinir todos os papéis da equipe
- [ ] C. Inspecionar o incremento e adaptar o backlog se necessário
- [ ] D. Fazer uma retrospectiva das falhas da sprint

2. O que diferencia um Sprint Goal de um Product Goal?

- [ ] A. O Sprint Goal orienta uma sprint específica; o Product Goal é de longo prazo
- [ ] B. O Product Goal é redefinido a cada sprint
- [ ] C. Ambos são equivalentes e intercambiáveis
- [ ] D. O Sprint Goal é opcional no Scrum

3. Em metodologias ágeis, o que significa a prática de "timebox"?

- [ ] A. Armazenar entregas em um repositório
- [ ] B. Documentar o tempo gasto em tarefas
- [ ] C. Criar limites de tamanho para histórias de usuário
- [ ] D. Definir duração fixa para eventos ou atividades

4. Quando o Product Owner deve priorizar novamente o backlog?

- [ ] A. Sempre que surgirem novas informações ou mudanças no negócio
- [ ] B. Apenas no início do projeto
- [ ] C. Após aprovação do gerente de projeto
- [ ] D. Nunca, pois o backlog é fixo

5. Como o Scrum Master pode ajudar a aumentar a produtividade do time?

- [ ] A. Delegando tarefas técnicas
- [ ] B. Assumindo atividades no lugar do time

- [ ] C. Substituindo membros com baixo desempenho
- [ ] D. Facilitando a remoção de impedimentos e promovendo melhoria contínua

6. O que é um "incremento" no contexto do Scrum?

- [ ] A. A soma de todos os itens completos do backlog ao final de uma sprint
- [ ] B. Um conjunto de tarefas que precisam ser iniciadas
- [ ] C. Um relatório de progresso da sprint
- [ ] D. A estimativa de esforço para o próximo ciclo

1.2 — Questões Objetivas: C# e Classes (0,25 ponto cada)

7. Em C#, o que define o escopo de acesso de uma propriedade?

- [ ] A. O tipo da variável
- [ ] B. O nome do método que a utiliza
- [ ] C. O modificador como `public`, `private`, `protected`
- [ ] D. O tipo de retorno do construtor

8. Qual o efeito de uma propriedade `auto-implementada` (auto-property) em C#?

- [ ] A. Cria automaticamente um campo de suporte interno
- [ ] B. Gera exceções por padrão
- [ ] C. Não pode ser usada fora da classe
- [ ] D. Requer implementação obrigatória de lógica interna

9. Qual afirmação está correta sobre listas em C#?

- [ ] A. Só podem armazenar tipos primitivos
- [ ] B. São estruturas genéricas que armazenam elementos dinamicamente
- [ ] C. São definidas apenas com `array[]`
- [ ] D. Não podem ser modificadas após a criação

10. Qual o papel de um método `override`?

- [ ] A. Impede uma classe de ser herdada
- [ ] B. Cria uma nova versão de um método com o mesmo nome
- [ ] C. Remove o método herdado
- [ ] D. Substitui a implementação de um método da classe base

11. Qual comando remove todos os elementos de uma lista em C#?

- [ ] A. `lista.Clear()`
- [ ] B. `lista.Remove()`
- [ ] C. `lista.DeleteAll()`
- [ ] D. `lista.Erase()`

12. O que significa um método ser `void` em C#?

- [ ] A. Ele não retorna valor
- [ ] B. Ele retorna null

- ☐ C. Ele é um método abstrato
- ☐ D. Ele só pode ser usado em interfaces

1.3 — Questões Dissertativas (1 ponto cada)

13. Em Programação Orientada a Objetos, qual é a importância de utilizar métodos dentro de uma classe? Explique como eles ajudam a organizar a lógica do sistema e dê um exemplo prático onde o uso de métodos facilita a reutilização do código e a legibilidade.

14. Em uma equipe Scrum, por que é importante ter papéis bem definidos (Scrum Master, PO e Dev Team)? Como isso evita conflitos e contribui para o sucesso do projeto?

🛠️ PARTE 2 — Desafios Práticos em C# (POO com Classes)

> Os desafios abaixo utilizam **propriedades, validações, listas e métodos de classe**.

◇ Exercício 1 — Classe `Aluno` com Cálculo de Média e Situação (1,0 ponto)

Crie uma classe `Aluno` com:

- `Nome` (string), `Notas` (lista de double)

Implemente os métodos:

- `AdicionarNota(double n)`
- `Media()` → média das notas
- `Situacao()` → retorna `"Aprovado"` se média ≥ 7, `"Recuperação"` se ≥ 5, `"Reprovado"` caso contrário

No `Main`:

- Cadastre um aluno e 3 notas
- Exiba nome, média e situação

◇ Exercício 2 — Classe `Produto` com Controle de Preço (1,0 ponto)

Crie a classe `Produto` com:

- `Nome` (string), `PrecoBase` (double), `Categoria` (string)

Implemente:

- Método `AplicarDesconto()` que aplica:
  - 15% se categoria for `"Eletrônico"`
  - 10% se for `"Vestuário"`
  - 5% para outras categorias
- Método `PrecoFinal()` retorna o valor com desconto aplicado

No `Main`:

- Cadastre 2 produtos de categorias diferentes
- Exiba os preços com e sem desconto

◇ Exercício 3 — Classe `ContaInvestimento` com Juros Simples e Movimentações (1,0 ponto)

Crie a classe `ContaInvestimento` com os seguintes atributos:

- `Titular` (string)
- `Saldo` (double)
- `TaxaJuros` (double, percentual anual)

Implemente os métodos:

- `Depositar(double valor)` → adiciona ao saldo (valor deve ser positivo)
- `Sacar(double valor)` → só permite se o valor for menor ou igual ao saldo
- `CalcularRendimento(int meses)` → aplica juros simples: \text{rendimento} = \text{saldo} \times \left(\frac{\text{taxa}}{100}\right) \times \left(\frac{\text{meses}}{12}\right)]
- `ExibirResumo()` → exibe nome, saldo atual e rendimento projetado para 12 meses

---

🎯 **No `Main`:**

- Crie uma conta com saldo inicial e taxa de juros informada pelo usuário
- Realize um **depósito válido** e uma **tentativa de saque inválido**
- Exiba o resumo da conta e o **rendimento estimado para 12 meses**

◇ Exercício 4 — Classe `ReservaHotel` com Validação de Dados (1,0 ponto)

Crie a classe `ReservaHotel` com:

- `NomeHospede` (string), `Dias` (int), `ValorDiaria` (double)

Métodos:

- `ValorTotal()` = dias * valor diária
- `Validar()` → dias e valor diária devem ser positivos
- `ResumoReserva()` → string com nome, total e validade da reserva

No `Main`:

- Cadastre reservas (válidas e inválidas)
- Exiba apenas as válidas com resumo

◇ Exercício 5 — Classe `LojaOnline` com Cálculo de Carrinho (1,0 ponto)

Crie `ItemCompra` com:

- `NomeProduto` (string), `Quantidade` (int), `PrecoUnitario` (double)

Classe `LojaOnline` com:

- Lista de `ItemCompra`
- Método `AdicionarItem(ItemCompra)`
- Método `TotalCarrinho()`
- Método `ListarItens()`

No `Main`:

- Adicione pelo menos 3 itens ao carrinho
- Exiba a lista e o total da compra

☑ Avaliação

Item	Pontuação
Teoria Scrum (6x0,25)	1,5 pts
Teoria C#/Classes (6x0,25)	1,5 pts
Dissertativas (2x1)	2,0 pts
Prática (5x1)	5,0 pts
Nota Máxima	**10,0 pts**
Nota Mínima p/ Aprovação (85%)	**8,5 pts**
