Resumo capítulo 4 - Thinking Object-Oriented

1. Introdução à Programação Orientada a Objetos (OOP)

- OOP surgiu como resposta à crescente complexidade do desenvolvimento de software
- Estruturar programas de forma mais intuitiva
- Programação estruturada e funcional

2. Porque OOP é popular?

- escalabilidade → adequado para sistemas simples e complexos
- facilidade de abstração → modelagem próxima à realidade
- bibliotecas extensas → grande suporte para ≠ domínios
- solução para a crise do software → auxílio no gerenciamento de projetos
- NÃO é uma solução mágica → exige criatividade, lógica e experiência

3. Conceitos Fundamentais do OOP

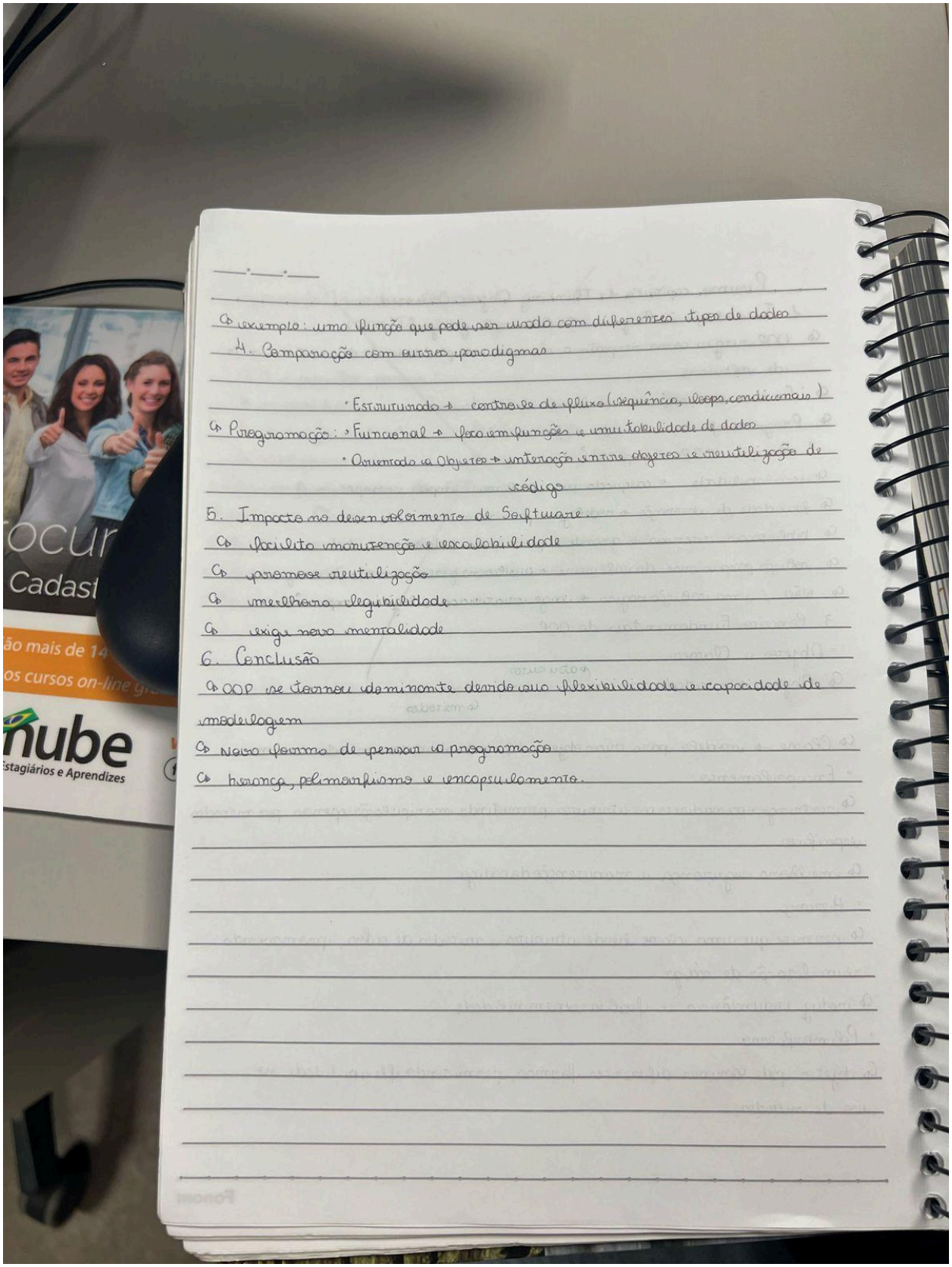
- Objetos e Classes
- Objeto → entidade com estado (atributos) e comportamento (métodos)
- Classe → modelo para criar objetos, definindo atributos e métodos comuns

- Encapsulamento
- restringe o acesso direto aos atributos, permitindo manipulação apenas por métodos específicos.
- melhora segurança e manutenção do código

- Herança
- permite que uma classe herde atributos e métodos de outra, promovendo reutilização de código
- reduz redundância e facilita extensibilidade

- Polimorfismo
- objetos pode assumir diferentes formas, permitindo flexibilidade no uso de métodos.

FORONI



- exemplo: uma função que pode ser usada com diferentes tipos de dados

4. Comparação com outros paradigmas

- Estruturado → controle de fluxo (sequência, loops, condicionais)
- Programação: • Funcional → foco em funções e imutabilidade de dados
  • Orientado a Objetos → interação entre objetos e reutilização de código

5. Impacto no desenvolvimento de Software.

- facilita manutenção e escalabilidade
- promove reutilização
- melhora legibilidade
- exige nova mentalidade

6. Conclusão

- OOP se tornou dominante devido sua flexibilidade e capacidade de modelagem
- Nova forma de pensar a programação
- herança, polimorfismo e encapsulamento.