Aula 25/11 ✏️

🎯 Exercício 1 — Adicionar "Saúde" ao Herói

Objetivo: Aplicar encapsulamento, atualizar estado visual, e alterar a UI.

✔️ O que tens de fazer:

1. Na classe Heroi, adicionar o atributo:

   ```
   private int saude;
   ```

   - inicializar com **100**
   - criar **getSaude()**
   - atualizar o construtor
2. Na UI:
   - adicionar um JLabel para mostrar a saúde
   - colocar cor dinâmica:
     - verde → >70
     - amarelo → entre 30 e 70
     - vermelho → <30
3. Quando o herói fizer **missão**, perde **10 de saúde**.
4. Quando fizer **descanso**, recupera **5 de saúde** (máximo 100).

🧪 Exercício 2 — Criar Botão "Comer Poção"

Objetivo: Criar uma nova ação UI + lógica separada.

✔️ O que tens de fazer:

- Criar o método public String pocao() na classe Heroi:
  - Aumenta energia +30
  - Aumenta saúde +20
  - Energia e saúde não podem ultrapassar 100
  - Retornar string com a mensagem
- Criar um botão btnPocao na UI
- Atualizar imagem/cores após o clique
- Mostrar mensagem no label inferior

🧪 Exercício 3 — Mudar a imagem conforme o NIVEL

Objetivo: Mostrar que atributos internos influenciam o visual.

✔️ O que tens de fazer:

Adicionar ao mapa de imagens mais uma imagem:

- heroi_master.png (para nível ≥ 5)

Na função atualizarEstadoVisual(), adicionar:

```
if (heroi.getNivel() >= 5) {
    labelImagem.setIcon(iconMaster);
}
```

🧪 Exercício 4 — Histórico de Ações (JList)

Objetivo: Introduzir um componente novo: **JList**.

✔️ O que tens de fazer:

1. Criar um DefaultListModel<String>
2. Criar um JList<String> com esse modelo
3. Pôr o JList numa barra lateral (por exemplo, BorderLayout.EAST)
4. Cada vez que o herói faz:
   - treino
   - descanso
   - missão
   - poção

   adiciona a frase ao histórico:

```
historico.addElement(mensagem);
```