**Sistema de Gestão de Aventuras (RPG em Python com POO e Combate Detalhado)**

**1. Objetivo Geral**

Desenvolver um projeto prático em Python utilizando os conceitos fundamentais de **Programação Orientada a Objetos (POO)**, aplicando estruturas de decisão, laços, listas, dicionários, funções e persistência de dados.
O projeto será desenvolvido em grupos de até **10 alunos**, com duração de **3 semanas**.

**2. Descrição do Projeto**

O grupo deverá construir um **RPG textual (jogo de aventura)** no qual o jogador cria um personagem, enfrenta missões e inimigos, ganha experiência, coleta itens e evolui.
O projeto deve ser totalmente **orientado a objetos**, explorando herança, encapsulamento e polimorfismo.

O jogo deverá exibir informações detalhadas de combate, como **HP do inimigo**, **dano causado**, **defesa aplicada** e **resultado de cada turno**, reforçando a compreensão de fluxo de execução e interação entre objetos.

**3. Estrutura do Projeto**

**3.1. Classes Principais**

- Personagem — representa o jogador com nome, atributos e inventário.
- Guerreiro e Mago — subclasses com habilidades específicas.
- Inimigo — inimigos genéricos e subclasses (Goblin, Lobo, etc.).
- Missao — define um desafio, inimigo e recompensa, com combate detalhado.
- Jogo — orquestrador do fluxo (menu, combate, salvar/carregar).
- Repositorio — responsável por salvar e carregar dados em JSON.
- Logger — registra eventos do jogo em arquivo .log.

**4. Organização do Repositório**

rpg_oo/

├─ README.md

├─ main.py

├─ jogo.py

├─ models/

│ ├─ base.py

```
│   ├── personagem.py
│   ├── classes.py
│   ├── inimigo.py
│   └── missao.py
└── utils/
    ├── repositorio.py
    └── logger.py
```

O arquivo principal main.py deve iniciar o jogo e exibir o menu de interação.

**5. Cronograma de Desenvolvimento**

**Semana 1 — Fundamentos de POO**

**Objetivos:**

- Implementar as classes básicas (Personagem, Inimigo, Missao, Jogo).
- Utilizar construtores (__init__), atributos e métodos.
- Criar um menu inicial com opções:
    - Criar personagem
    - Encarar missão
    - Salvar / Carregar jogo
    - Sair

**Entregável:**
Versão funcional do jogo com personagens e missões simples.

**Semana 2 — Regras, Coleções e Funções**

**Objetivos:**

- Usar listas e dicionários para organizar inimigos, itens e missões.
- Adicionar funções auxiliares e controle de fluxo.
- Trabalhar com módulo random (sorteio de inimigos, itens, missões).
- Implementar um inventário de itens e sistema de cura/mana.
- Validar ações do jogador com estruturas condicionais.

- Exibir dados de combate (HP, dano, turnos, vitória e derrota).

**Entregável:**
Versão aprimorada com múltiplos inimigos, missões e inventário funcional, incluindo logs de turnos de batalha.

**Semana 3 — Polimorfismo, Herança e Persistência**

**Objetivos:**

- Criar subclasses (Arqueiro, Chefao, etc.) com habilidades próprias.
- Implementar métodos sobrescritos (atacar, habilidade_especial).
- Utilizar o módulo json para salvar e carregar o progresso.
- Implementar logs de eventos (vitórias, derrotas, itens coletados).
- Aplicar princípios de polimorfismo entre classes.

**Entregável:**
Versão final do jogo com hierarquia OO, logs e persistência de dados.

**6. Critérios de Avaliação (0–10 pontos)**

| Critério | Descrição | Peso |
|---|---|---|
| **Modelagem OO** | Estrutura de classes, herança, encapsulamento e polimorfismo. | 3 |
| **Fluxo do jogo** | Menu, combate, decisões e progressão. | 2 |
| **Lógica e validações** | Estruturas condicionais e cálculos de combate. | 2 |
| **Coleções e funções** | Uso de listas, dicionários e funções auxiliares. | 1.5 |
| **Organização e documentação** | Estrutura de pastas, docstrings e comentários. | 1 |
| **Criatividade** | História, nomes, classes e narrativa. | 0.5 |

**7. Regras Gerais**

- O trabalho deve ser **desenvolvido em grupo** (máximo 10 alunos).
- Cada aluno deve contribuir com **pelo menos uma classe ou funcionalidade**.
- O código deve conter **docstrings e comentários em português**, explicando os principais trechos.
- É obrigatório o uso de **POO desde o início** do projeto.

- O jogo deve **executar sem erros** em ambiente Python 3.11+.

**8. Entrega Final**

**Formato de Entrega:**

- Repositório no GitHub.
- Incluir um arquivo README.md explicando:
    - Integrantes do grupo
    - Como rodar o jogo
    - Resumo das principais classes e funções

**Data de Entrega:** ao final da **3ª semana** de desenvolvimento.

**9. Combate Detalhado (requisito obrigatório)**

Cada missão deve exibir **detalhes do combate** no terminal, mostrando:

- Nome do inimigo e HP inicial.
- Dano causado e defesa aplicada.
- HP restante do inimigo e do jogador a cada turno.
- Resultado final com XP e loot.

**Exemplo de saída esperada:**

=== Missão: Encontro na Floresta ===

Você encontrou um Goblin!

HP do inimigo: 14

--- Turno 1 ---

Léo usa habilidade especial!

Léo causa 8 de dano em Goblin!

Goblin agora tem 6 HP.

--- Turno 2 ---

Goblin causa 2 de dano em Léo!

Léo agora tem 20 HP.

--- Turno 3 ---

Léo causa 5 de dano em Goblin!

Goblin agora tem 1 HP.

--- Turno 4 ---

Goblin causa 3 de dano em Léo!

Léo agora tem 17 HP.

--- Turno 5 ---

Léo causa 6 de dano em Goblin!

Goblin agora tem 0 HP.

=== Resultado da Missão ===

🎉 Léo venceu o combate!

XP ganho: 50

Itens obtidos: poção

==========

**10. Desafios Extras (opcionais – valem pontos bônus)**

- Implementar um **modo de múltiplas missões em sequência**.
- Criar uma **classe Chefão** com ataques especiais.
- Adicionar **chance de crítico** e **status aleatórios** (veneno, fogo, cura).
- Exibir um **ranking** de pontuação (XP total).
- Integrar **cores e emojis** para deixar o jogo mais visual.
- Exibir **barras de vida** com # representando HP restante.

**11. Exemplo de Execução Completa**

=== RPG OO ===

[1] Criar personagem

[2] Encarar missão

[3] Salvar

[4] Carregar

[0] Sair

> 1

Nome do personagem: Léo

Escolha a classe: [1] Guerreiro  [2] Mago

> 2

Personagem criado: Léo (Mago)

> 2

=== Missão: Encontro na Floresta ===

Você encontrou um Goblin!

HP do inimigo: 14

--- Turno 1 ---

Léo usa habilidade especial!

Léo causa 8 de dano em Goblin!

Goblin agora tem 6 HP.

--- Turno 2 ---

Goblin causa 2 de dano em Léo!

Léo agora tem 20 HP.

=== Resultado da Missão ===

🎉 Léo venceu o combate!

XP ganho: 50

Você recebeu: poção

**12. Referências Didáticas**

- **Material de Aula:** slides de “Paradigmas de Python – POO”.
- **Documentação Oficial:** https://docs.python.org/pt-br/3/tutorial/classes.html
- **Livro Base:** *Use a Cabeça! Python*, O'Reilly.
- **Exemplo Complementar:** projetos de RPG com classes no GitHub (educacionais).