Projeto 3: Labirinto de Mistérios em Java

Contexto: Parabéns! Você foi escolhido para embarcar em uma emocionante jornada de desenvolvimento de um jogo de labirinto em Java. Este não é apenas um jogo, mas uma aventura onde os jogadores devem navegar por um labirinto misterioso gerado aleatoriamente, coletar tesouros escondidos e evitar perigos iminentes.

Requisitos do Projeto:

1. **Classes (10%)**:
    - **Aventureiro**: Esta classe representa o bravo aventureiro no jogo.
        - Atributos: nome, localização atual no labirinto, tesouros coletados (ArrayList).
        - Métodos: getters e setters para os atributos, mover-se pelo labirinto, coletar tesouro.
    - **Tesouro**: Esta classe representa um tesouro que pode ser descoberto no jogo.
        - Atributos: nome, localização no labirinto, valor em pontos.
        - Métodos: getters e setters para os atributos.
    - **Perigo**: Esta classe representa um perigo/armadilha que o aventureiro deve evitar no jogo.
        - Atributos: localização no labirinto, dano potencial.
        - Métodos: getters e setters para os atributos.
    - **Labirinto**: Esta classe representa o labirinto e contém a lógica principal do jogo.
        - Atributos: estrutura do labirinto (ArrayList de ArrayLists), lista de tesouros (ArrayList), lista de perigos (ArrayList).
        - Métodos: gerar labirinto, adicionar tesouro, remover tesouro, adicionar perigo, remover perigo.
2. **Coleções (10%)**: Utilizar coleções (como ArrayList) para armazenar a lista de tesouros coletados pelo aventureiro (na classe Aventureiro), a estrutura do labirinto, a lista de tesouros e a lista de perigos no labirinto (na classe Labirinto).
3. **Tratamento de Exceções (10%)**: Implementar tratamento de exceções para garantir uma experiência de jogo suave. Por exemplo, tratar situações como o aventureiro tentando se mover para uma posição fora do labirinto ou tentando coletar um tesouro que não está na mesma posição que o aventureiro.
4. **Polimorfismo (5%)**: Implementar polimorfismo na classe Tesouro. Por exemplo, a classe Tesouro pode ter um método efeito() que é sobrescrito nas classes que herdam de Tesouro para implementar diferentes efeitos quando o tesouro é coletado pelo aventureiro.
5. **Abstração (5%)**: A abstração é implementada através das classes e métodos descritos acima. Cada classe representa uma entidade distinta (Aventureiro, Tesouro, Perigo, Labirinto) e cada método representa uma ação que pode ser realizada por essa entidade.
6. **Encapsulamento (5%)**: O encapsulamento é implementado através do uso de modificadores de acesso (private, public) para os atributos das classes. Os atributos são definidos como privados e são acessados através de métodos públicos (getters e setters).
7. **Entrega (5%)**: Link do GitHub via Teams. No GitHub deve conter o código, além de screenshots da interface do usuário. Um readme com uma breve descrição do que foi

feito deve estar preenchido. A equipe toda deve ter acesso ao GitHub compartilhado (é um projeto em equipe).

8. **Requisitos adicionais (20%)**: Será considerado o que for feito além do requerimento mínimo esperado.
9. **Easter Egg (10%)**: Faça algo que torne o seu projeto inesquecível.
10. **Apresentação (20%)**: Pitch de no máximo 10 minutos vendendo a ideia. Será avaliado pela turma (cliente em potencial) por votação (kahoot) qual o melhor projeto (comprou a ideia) entre dois projetos apresentados.

Observação: Você tem liberdade para criar um ou vários contextos para o jogo. Seja criativo! Talvez o labirinto seja uma antiga pirâmide egípcia ou um castelo assombrado. Deixe sua imaginação correr solta!

Prazo: 17 de junho de 2024

Boa sorte com o seu projeto!