UNICAMP

Universidade Estadual de Campinas
Instituto de Computação

Laboratório 3

Lógica de Batalha com Monstro

MC322 - Programação Orientada a Objetos

Professor: Marcos Raimundo
PEDs: Augusto Cesar / Heigon Soldera
PAD: Matheus Seiji Luna Noda

INSTITUTO DE COMPUTAÇÃO

1. Descrição Geral

No laboratório três, avançaremos na construção do nosso jogo de cartas, incorporando conceitos e aprendizados adquiridos anteriormente. Após a apresentação de conceitos e as suas limitações encontradas no laboratório dois, agora será avaliado a necessidade de refatorar o projeto para uma abordagem mais organizada.

Uma das melhorias propostas é a utilização de uma Classe Enum para gerenciar as diferentes Classes, Raças e Tipo de Itens. Além disso, esse laboratório irá aplicar os princípios de Polimorfismo e Sobrecarga para criar novas classes que representarão diferentes tipos de cartas e suas funcionalidades específicas.

Depois desse laboratório já teremos como base uma estrutura jogável, onde será capaz batalhar com monstros. Lembrando que as regras se inspiram em Munchkin, porém possui mudanças para facilitar a implementação e estudos dos conceitos necessários em POO. Segue abaixo uma estrutura das classes que compõem este laboratório. Para a conclusão do laboratório pode ser implementadas mais classes e métodos, porém importante manter a utilização das classes e conceitos presentes nesse diagrama.

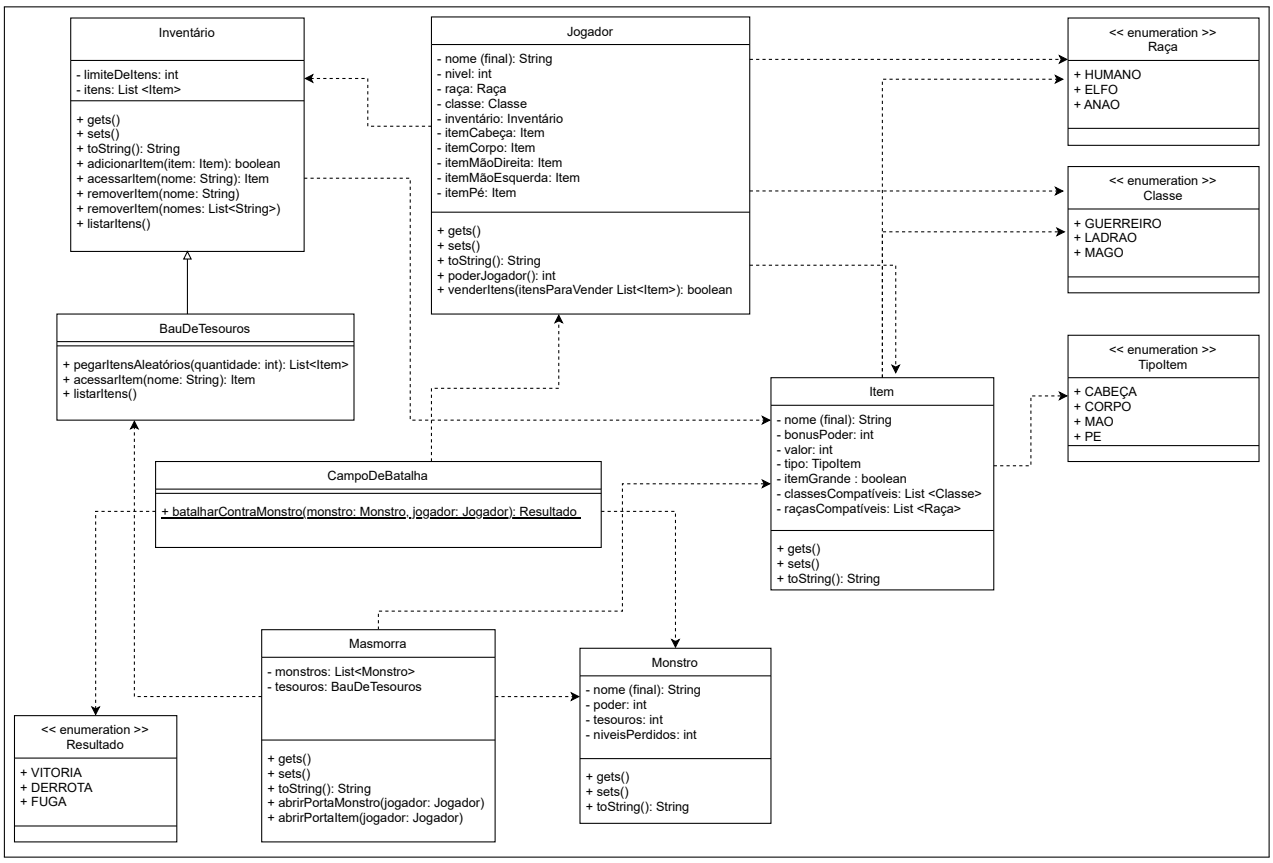

Figura 1: Diagrama de Classe

2. Objetivos

Os principais objetivos deste laboratório consistem em:

- Consolidação dos conteúdos vistos nos labs anteriores;
- Utilização de Enum;
- Utilização de método estático;
- Aplicação do conceito de polimorfismo e sobrecarga de métodos;
- Capacidade de abstração de como aplicar os conceitos de orientação objetos em face das funcionalidades requeridas.
- Capacidade de refatorar um projeto;
- Entrada de dados via teclado (System.in);

3. Atividades

A principal atividade desse laboratório é a implementação de um loop de jogo onde é possível um jogador escolher atividades a se realizar e batalhar com monstros, ganhando itens para poder equipar ou perdendo níveis, caso não consiga derrotar.

4. Descrição das Classes

Algumas classes foram criadas e outras foram alteradas, a seguir segue alguns destaques importantes:

Classe - Jogador

A classe Jogador representará cada participante do jogo. Além de nome e nível, a classe será composta da raça a qual o jogador pertence e a classe escolhida. Aqui foi inserido os itens a se equipar e o controle do inventário. Lembrando que o jogador pode possuir apenas 1 item grande equipado, mas o inventário não tem limite de itens grandes.

Método - poderJogador

O método *poderJogador* calcula o poder de ataque do jogador com base no valor da soma dos itens equipados mais o nível do jogador.

Método - venderItens

O método *venderItens* recebe uma lista de itens, as quais o jogador tem no inventário. Ao somar 1000 de valor, o jogador sobe 1 nível. Dessa forma, esse método irá vender os itens, removendo eles do inventário do Jogador e acrescentando um nível ao jogador a cada 1000 de valor.

Classe - Monstro

A classe Monstro será responsável por modelar os inimigos encontrados pelos jogadores durante a exploração das masmorras. Cada monstro terá atributos como nome, nível de poder. Aqui muda a lista de itens do Monstro por uma quantidade inteira de tesouros que irá dar ao jogador caso seja derrotado.

Classe - Item

A classe Item representará os equipamentos e objetos que os jogadores podem encontrar e utilizar durante o jogo. Cada item terá atributos como nome, tipo que agora é um Enum, bônus de poder e valor. Além disso, os objetos possuem classes e raças específicas que podem utilizá-los representado em uma lista.

Classe - Inventário

No inventário deverão ser armazenados todos os itens que o jogador carrega. Ele é responsável pela adição e remoção.

Método - adicionarItem

Método para adicionar item ao inventário. Retorna verdadeiro em caso de sucesso e falso em caso de erro, por exemplo, inventário cheio.

Método - acessarItem

Método para retornar um objeto Item do inventário a partir do atributo nome de Item.

Método - removerItem

Método para remover um objeto Item do inventário a partir do atributo nome de Item.

Método - listarItens

Método para imprimir todos os itens no inventário.

Classe - BauDeTesouros

Essa classe será herdada de Inventário, para aproveitarmos boa parte dos métodos já implementados e trabalharmos Herança com Polimorfismo e Sobrecarga. Ela será utilizada como baú para a Masmorra do jogo e conterá os itens de recompensa.

Método - acessarItem

Diferente do método da classe Invnetário, ao acessar um item específico do baú ele também será removido do mesmo.

Método - listarItens

Agora esse método irá listar apenas 2 itens, escolhido de forma aleatória.

Método - pegarItensAleatórios

Método que irá retornar uma lista de itens aleatórios do Baú, a quantidade de itens é definida como parâmetro. Esses itens serão removidos do baú

Classe - CampoDeBatalha

Classe que, até o momento, terá um método estático para implementação da batalha entre um Jogador e um Monstro. A batalha segue três pontos principais

- Se o jogador possuir maior poder que o monstro ele ganha a batalha, será adicionado ao Inventário do Jogador a quantidade de itens aleatórios, definidos pelo atributo tesouros do Monstros, do Baú de Tesouros da Masmorra;
- Se o jogador possuir nível menor, deve tentar fugir. Para fugir deve tirar 5 ou 6 em um dado, que pode ser simulado pela classe Random;
- Caso fuja, nada acontece. Caso não consiga fugir, o jogador perde o número de níveis definido pelo monstro.

Classe - Masmorra

Essa classe é essencial para o desenvolvimento do jogo. Ela possuirá o Baú De Tesouros para serem retirados os prêmios dos jogadores e uma lista de Monstros que podem serem enfrentados ao entrar na masmorra.

Método - abrirPortaMonstro

Irá abrir a porta do Monstro e o jogador deve batalhar com ele. Utilizando o método estático batalharContraMonstro. E aplicar as mudanças necessárias ao Jogador a partir do resultado da batalha.

Método - abrirPortaItem

Irá abrir a porta de Item, nesse caso deve ser utilizado o método listarItens para imprimir dois itens aleatórios que o jogador deve escolher um deles. Depois de escolhido deve utilizar o método acessarItem para pegar esse item do Baú e atribuir ao Inventário do Jogador.

5. Execução

Além da correta implemetação da UML, esse laboratório deve possuir na uma lógica para simulação do jogo a partir de entradas padrões do usuário pelo teclado:

- Instanciação dos objetos na classe Main e inicialização da Masmorra com alguns Monstros e Itens no Baú. Além do Jogador:
  - A Main deve possuir um loop de rodadas que possuem a seguinte ordem: Ação do jogador, abrir porta da masmorra e outra ação do jogador.
  - Ações possíveis do Jogador que ele pode escolher via teclado, podem se repetir até o jogador escolher abrir uma porta da Masmorra:
    - Listar os itens do inventário.
    - Equipe itens do inventário ao Jogador.
    - Vender itens do inventário para subir um nível.
    - Ver o poder total que possui, seu nível e os itens equipados.
    - Passar para abrir a porta
  - Abrir uma porta da Masmorra:
    - Definir aleatoriamente se irá abrir uma porta de Monstro ou Item.
    - Se abrir uma porta de Item o jogador deve poder escolher entre dois itens aleatórios do baú.
    - Se for uma porta de monstro o jogador irá lutar com o monstro, deve informar o jogador do resultado e mostrar os itens que ganhou ou os níveis que perdeu.
- Outra oportunidade de ação do jogador para vender ou equipar itens até passar a rodada para outro jogador.

5.1. Exemplo de interação:

```
Jogo mostra:
Jogador 1: Magic Mike
O que você deseja fazer?
1 - Listar itens do inventário
2 - Equipar itens do inventário
3 - Vender itens do inventário
4 - Ver informações do jogador
5 - Passar para abrir a porta

0 - Sair do jogo

--------------

Jogador digita: 1

--------------

Jogo mostra:
Listando itens do inventário
- Espada de fogo do pantal - nível 9
- Escudo de carvalho - nível: 4
...
[Pressione 1 para voltar]
```

```
--------------

Jogador pressiona: 1

--------------

Jogo Mostra:
Jogador 1: Magic Mike
O que você deseja fazer?
1 - Listar itens do inventário
2 - Equipar itens do inventário
3 - Vender itens do inventário
4 - Ver informações do jogador
5 - Passar para abrir a porta

0 - Sair do jogo
```

6. Avaliação

- Entrega realizada dentro do prazo estipulado;
- Qualidade do código desenvolvido (tabulação, comentários);
- Implementação da lógica do jogo.
- Desenvolvimento correto dos métodos de acesso.

7. Entrega

- **A entrega do Laboratório é realizada exclusivamente via Github**.
- Utilize os horários de laboratório e atendimentos para tirar eventuais dúvidas de submissão e também relacionadas ao desenvolvimento do laboratório.
- **Prazo de Entrega:** 16/04 - 19h