

Nota: 10.0

(E) Tempo: **Terminou**



Laboratório 4: Coleções Genéricas I

Objetivo

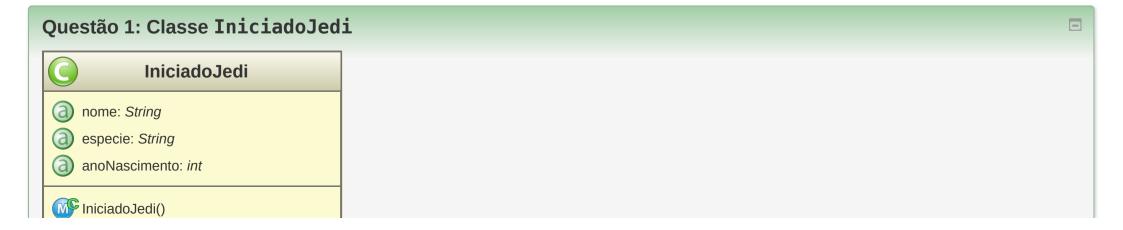
• Modelagem e implementação de classes em Java que usam listas implementadas pela *Java Collections Framework*.

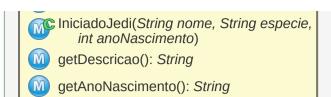
Descrição

- Neste trabalho, você implementará as classes necessárias para representar *Iniciados Jedi* em uma *Sessão Jedi*.
- Irá descrever atributos e implementar métodos e construtores com base em um diagrama de classes da UML.
- Em seguida, objetos serão criados a partir das classes implementadas.
- Nesta abstração, uma Sessão Jedi possui um Mestre Jedi e uma lista de Iniciados Jedi.

Passos Iniciais

- Inicie o Eclipse. Vá em "File" → "New" → "New Java Project". Nome do projeto: "TAP-ColecoesGenericasI".
 - Na parte *Module* (parte de baixo da janela), desselecione "Create module-info.java file".
- Nas questões a seguir, será pedido para criar diversas classes. Para isso, vá em "File" → "New" → "Class".
 - Deixe o campo "package" em branco.
- O Eclipse compila automaticamente as classes sempre que você salva. Para executar e testar, basta ir em "Run" → "Run" (Ctrl+F11).
- A saída e entrada do programa será feita através da aba "Console" do Eclipse.





Crie uma classe para representar o IniciadoJedi. Os atributos de um objeto da classe IniciadoJedi poderão ser setados diretamente (e.g., iniciadol.nome="Katooni") ou pelo *método construtor*. Use a técnica de *encadeamento de construtores* para criar os construtores da classe, semelhante ao feito em sala e mostrado nos slides.

Implemente os métodos:

- getAnoNascimento: retorna uma String contendo o ano de nascimento do Iniciado Jedi seguido da sigla ABY ("Antes da Batalha de Yavin". caso o anoNacimento seja negativo) ou DBY ("Depois da Batalha de Yavin", caso o anoNascimento seja positivo). Exemplo: "23 ABY", para um anoNascimento=-23.
- getDescricao: retorna uma String contendo a descrição completa do Iniciado Jedi de acordo com o exemplo abaixo:

Katooni (especie=Tholothian, nascimento=23 ABY)

Para testar a classe, crie uma nova classe chamada SessaoJediMain. Nesta classe, crie o método main, que será o ponto de partida do seu programa. No método main, crie um ou mais objetos da classe IniciadoJedi e, em seguida, imprima o resultado da execução do método getDescricao dos objetos criados.

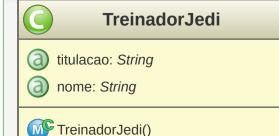
Para essa guestão, submeta apenas a classe IniciadoJedi. Não precisa submeter a classe SessaoJediMain.

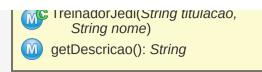
O prazo de entrega do trabalho terminou. Portanto, o botão abaixo está desabilitado.

Enviar "IniciadoJedi.java"

Solução correta!

Questão 2: Classe TreinadorJedi





De forma semelhante às questões anteriores, crie uma classe para representar o Treinador Jedi.

Implemente o método:

• getDescricao: retorna uma String contendo a descrição completa do TreinadorJedi de acordo com o exemplo abaixo (onde a titulação é "Grão-Mestre"):

Grão-Mestre Fae Coven

Para testar a classe, modifique a classe SessaoJediMain, criada na questão anterior, para criar um ou mais objetos da classe TreinadorJedi e, em seguida, imprima o resultado da execução do método getDescricao dos objetos criados.

Para essa questão, submeta apenas a classe TreinadorJedi. Não precisa submeter a classe SessaoJediMain.

O prazo de entrega do trabalho terminou. Portanto, o botão abaixo está desabilitado.

Enviar "TreinadorJedi.java"

Solução correta!

Ouestão 3: Classe SessaoJedi



SessaoJedi



nome: String



treinador: TreinadorJedi



iniciados: ArrayList<IniciadoJedi>



SessaoJedi(String nome, TreinadorJedi treinador)



addIniciado(IniciadoJedi iniciado): void



getIniciado(String nome): IniciadoJedi



getMediaAnoNascimento(): double



getDescricao(): String

De forma semelhante às questões anteriores, crie uma classe para representar uma SessaoJedi.

Implemente os métodos:

• addIniciado: adiciona um IniciadoJedi na lista de iniciados que farão parte da sessão. Caso já exista um iniciado com o mesmo nome, o iniciado não deve ser adicionado.