Laboratório Classes Abstratas e Interfaces (Segunda Parte)

(Deve ser feito Individual ou em dupla)

Prof. Dirson S. de Campos

dirson_campos@ufg.br

Material elaborado em parceria com os professores **Nádia F. F. da Silva, Juliana P. Félix, Guilherme S. Marques** e **Reinaldo de S. Júnior**.

09/11/2023

INF
INSTITUTO DE INFORMÁTICA

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

Instruções

Resolva as questões que não envolvem código em um único arquivo chamado **exercicio.txt**. Lembre-se de identificá-las apropriadamente para facilitar a correção, colocando números e quebra de linha para cada uma. Também no conteúdo deste arquivo, coloque seu nome e número de matrícula. Em caso de fazer em dupla coloque a matrícula e nome de ambos os estudantes.

Para as questões envolvendo UML, deve ser enviado:

- Enviar o arquivo do UMLet no formato .uxf dentro do pacote de classes;

Para as questões que envolvem códigos, deve ser enviado:

- O pacote completo criado pelo Eclipse ou no Visual Studio Code OU;
- Uma pasta com todas as classes usadas.

Se houver mais de uma questão de código, lembre-se de separar em pastas ou identificar no nome dos arquivos a qual se refere. Crie o hábito de identificar o código, bem como seu nome e número de matrícula nos comentários antes das classes.

Junte todos os arquivos finais em um único **.zip,** de preferência com seu nome para submeter no SIGAA.

Exercício de números 1 e 2 (Parte teórica)

Exercício 1 - Teórico

Preencha as lacunas em cada seguimento, de acordo com o que aprendemos:

1. Uma classe abstrata não permite que sejam criados _____ de si mesma.
2. Uma classe concreta não pode ter métodos _____.
3. Uma interface não pode ter _____ nem métodos _____ dentro de sua definição.
4. Em uma definição de classe, usa-se a palavra chave _____ antes de uma lista de interfaces.
5. O modificador do Javadoc _____ também conhecido como a forma de informar ao compilador da sobrescrita é quando reescrevemos a lógica de um método que foi herdado da classe mãe.

Exercício 2 - Teórico

2.1 - O que é um Diagrama de Classe em UML? Dê um pequeno exemplo que não seja o mesmo deste lab. Comente as partes do diagrama de Classe do seu exemplo.

Exercício de números 3 (Parte prática)

Exercício 3 - Prático

Nos slides seguintes, está o código de como deverá ficar sua classe `Principal`, demonstrando o uso de alguns dos métodos implementados e do Polimorfismo viabilizado com a `Interface Estudante`.

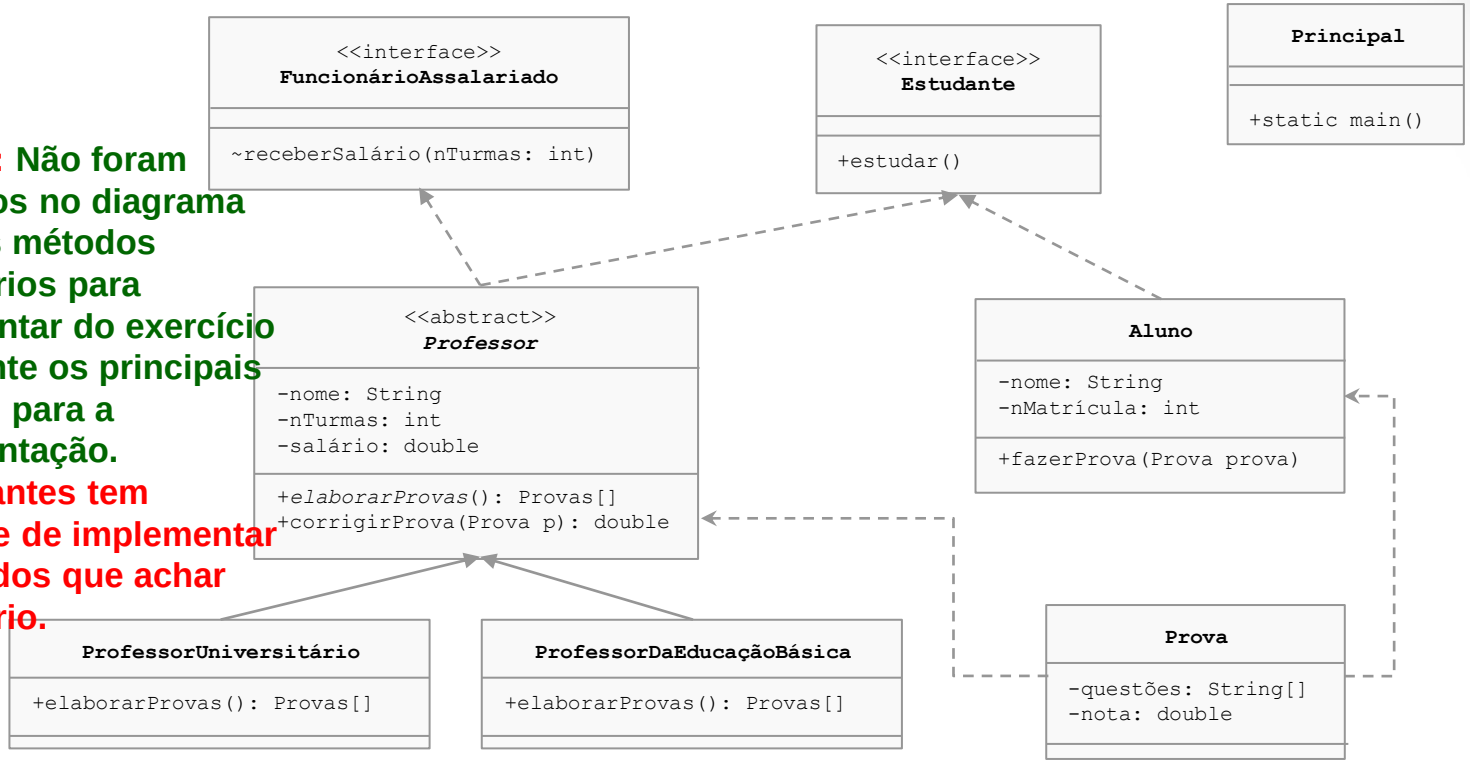
Mais adiante, está o Diagrama UML que mostra a modelagem proposta para este problema. Fique atento para o uso de Classes Abstratas e Interfaces.

A classe `Prova` tem valor de nota inicial `0`. Seu construtor precisa apenas das `questões`, já que as `respostas` serão configuradas depois, mas o vetor das `respostas` deve ser inicializado com o tamanho do vetor de `questões` passado.

Além disso, no último slide, estão as descrições de algumas regras para implementação dos métodos pedidos.

Exercício 3 - Implemente o Diagrama de Classe abaixo na ferramenta UMLet

Atenção: Não foram colocados no diagrama todos os métodos necessários para implementar do exercício 3, somente os principais métodos para a implementação.
O estudantes tem liberdade de implementar os métodos que achar necessário.



Exercícios 3

Regras para codificação lógica dos métodos:

- `corrigirProva`: todo professor conferirá provas do mesmo modo: percorrendo o vetor de respostas da prova e conferindo se, para questão "P1Q1", a resposta é "R1", se para "P1Q2" é "R2", e assim vai. A nota final calculada deve ser então definida com peso igual entre as questões.
- `receberSalário`: todo professor terá seu salário definido baseando-se na regra: `salário + (nTurmas * salário * 0.05)`.
- `elaborarProvas`: o professor universitário deverá elaborar 3 provas com 2 questões cadas, já o professor da educação básica fará 4 provas com 5 questões. O conteúdo das questões é simplesmente "P1Q1" para a questão 1 da prova 1, "P1Q2" para a questão 2 da prova 1 e assim vai...
- `fazerProva`: o aluno faz a prova preenchendo o vetor de respostas com valores com "R1", "R2"... sempre o mesmo valor para este exercício.
- `estudar`: o método estudar é apenas um `System.out.println` personalizado para cada caso, indicando que aquele indivíduo está estudando.

Exercícios 3 – Gerar o Diagrama de Classe da UML com setas

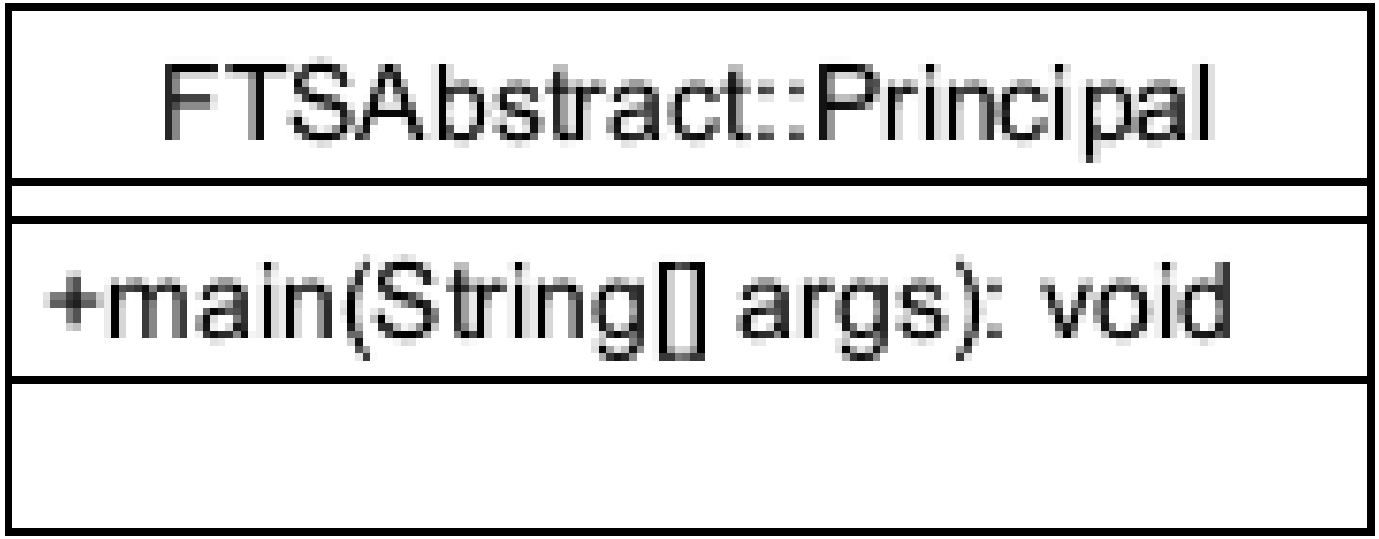
Gere o diagrama de Classe da UML no Umlet (**similar ao desenhado no PowerPoint no slide 5,** porém a ferramenta UMLet tem suas própria notação UML (forma no design do Diagrama de Classe).

Observações: O tipo do arquivo gerado pelo UMLet é **.uxf (Uml eXchange Format)**, o arquivo deverá ter o seu primeiro + as iniciais do seu sobrenome. No caso de ser feito em dupla, escolha um dos nomes dos estudantes livremente para o pacote.

- **Exemplo: Se o estudante chamar Fulano de Tal da Silva o nome do arquivo .unf deverá ser FulanoTS.uxf**

Soluções parciais da parte prática do Exercício 3 desenvolvido pelo professor
Código-fonte da Classe principal

Exercício 3 Diagrama de Classe da Classe Principal na ferramenta UMLet



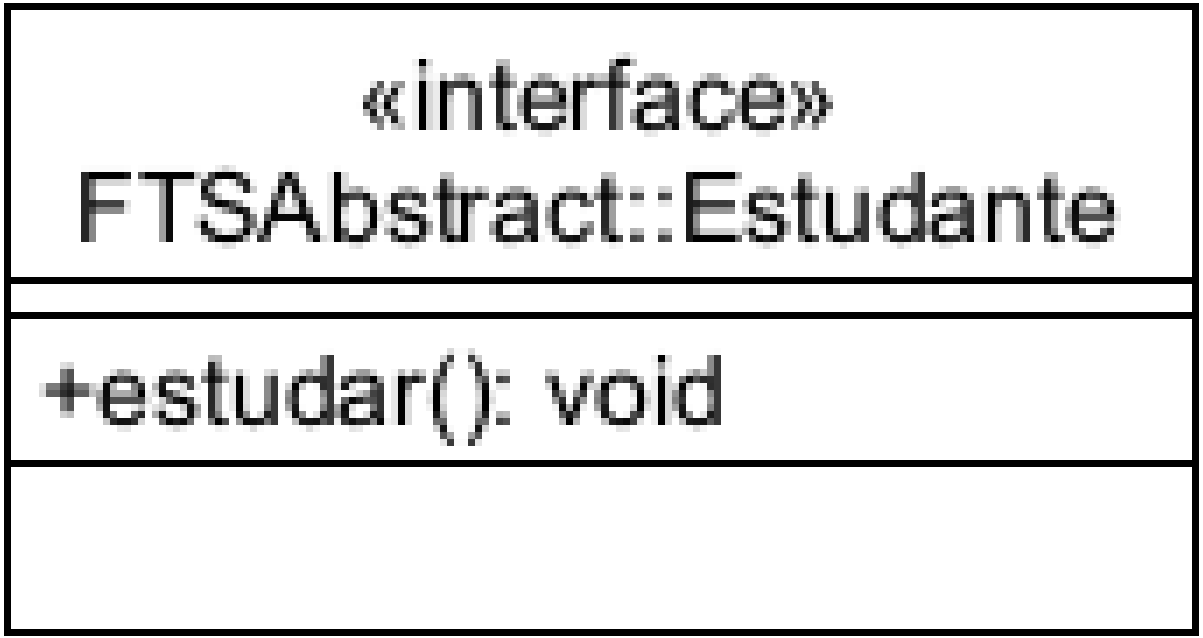
Exercício 3 – Código-fonte da Classe Principal

```
public class Principal {

    public static void main(String[] args) {
    Professor maria = new ProfessorUniversitário("Maria", 2, 3000);
    Professor jose = new ProfessorDaEducaçãoBásica("Jose", 3, 2000);
    Aluno PrimeiroNomedoEstudante = new Aluno("PrimeiroNomedoEstudante",
12345);
  //Caso seja feita em dupla, colocar criar o objeto SegundoNomedoEstudante
    Estudante[] pessoasQueEstudam = new Estudante[3];
    pessoasQueEstudam[0] = maria;
    pessoasQueEstudam[1] = jose;
    pessoasQueEstudam[2] = PrimeiroNomedoEstudante;
    //Caso seja feita em dupla, colocar no vetor o SegundoNomedoEstudante
    for (Estudante estudante: pessoasQueEstudam)
    estudante.estudar();

    Prova[] provasDaFaculdade = maria.elaborarProvas();
```

Exercício 3 - Código-fonte da Classe Principal (continuação)


```
    PrimeiroNomedoEstudante.fazerProva(provasDaFaculdade[0]);
     //Caso seja feita em dupla,
    //colocar a prova para o SegundoNomedoEstudante
    maria.corrigirProva(provasDaFaculdade[0]);

     System.out.println("PrimeiroNomedoEstudante tirou "+
provasDaFaculdade[0].getNota() + " na prova");
      //Caso seja feita em dupla,
     // imprimir o segundo estudante
     }
}
```

Exercício 3 Diagrama de Classe da Interface Estudante na ferramenta UMLet

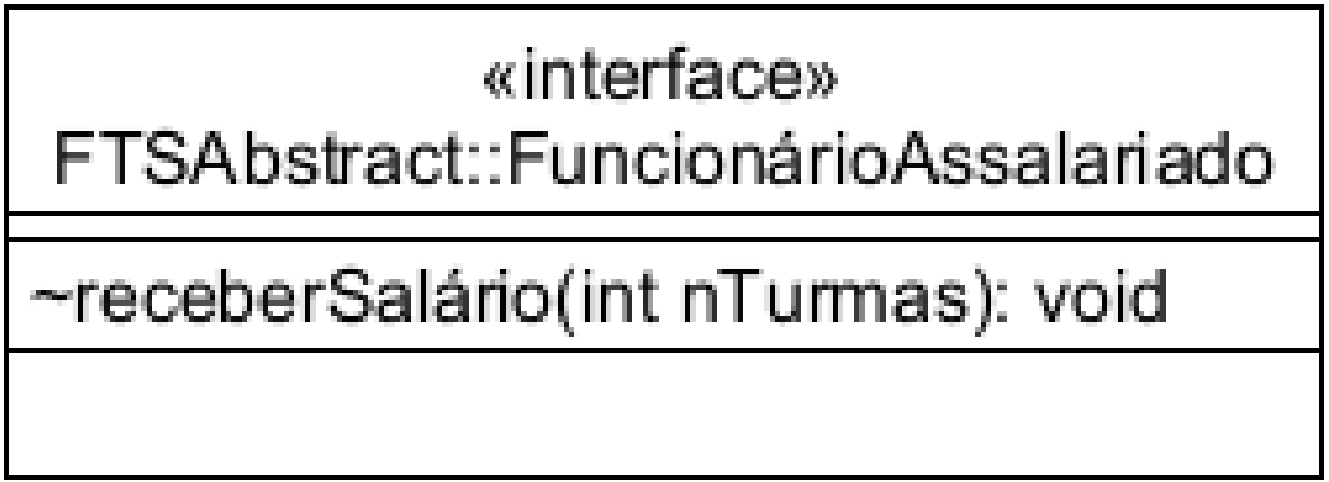


Exercício 3 – Código-fonte da Interface Estudante

INF
INSTITUTO DE INFORMÁTICA

```
package FTSAbstract;

public interface Estudante {

public void estudar();

}
```

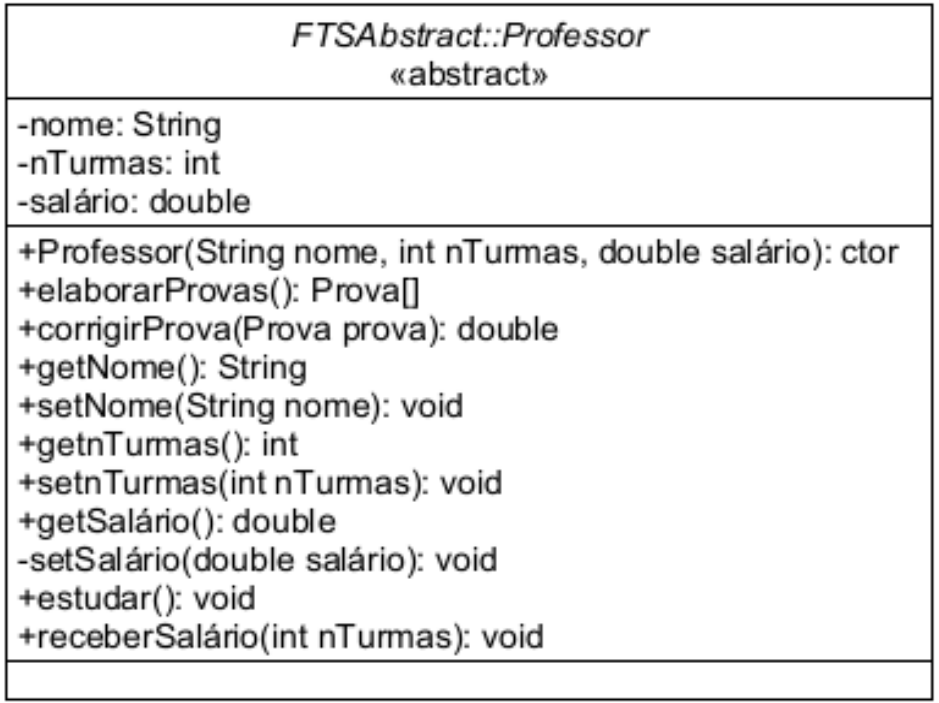
Exercício 3 Diagrama de Classe da Interface FuncionárioAssalariado na ferramenta UMLet



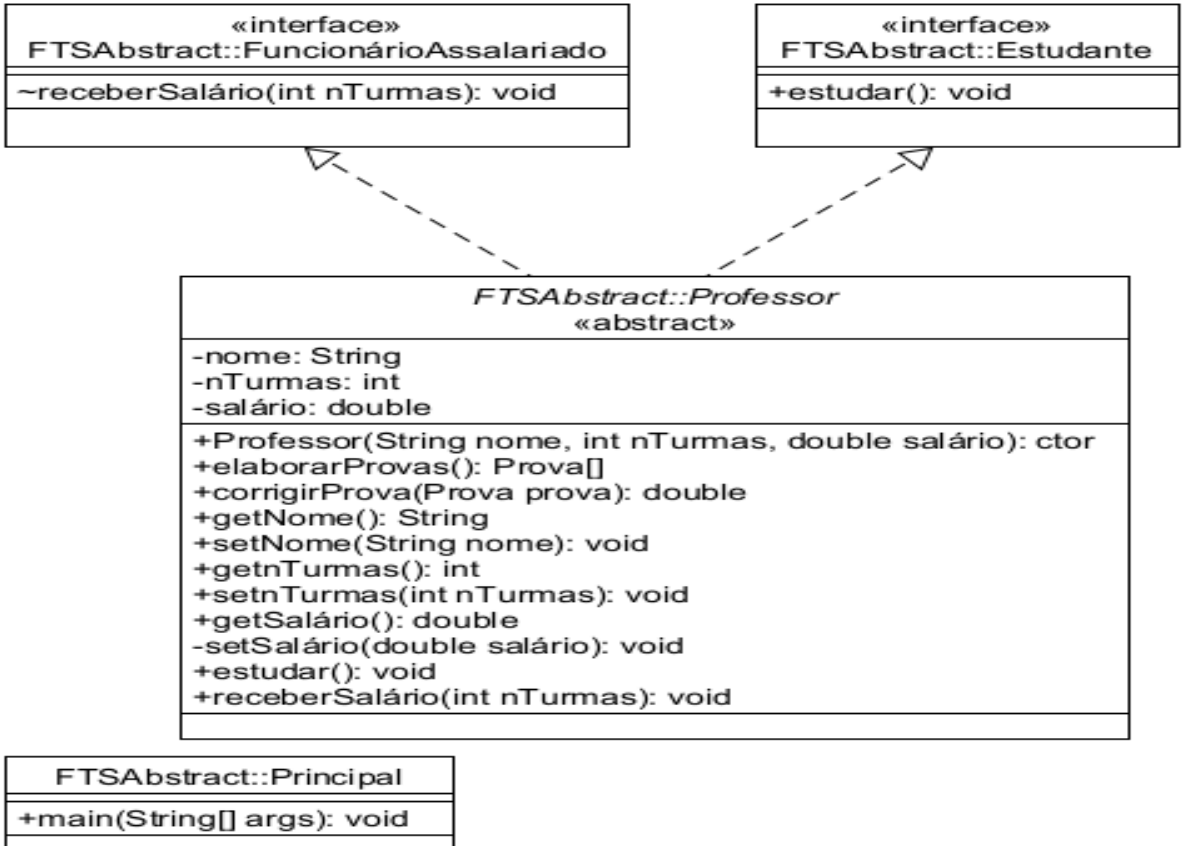
Exercício 3 – Código-fonte da Interface FuncionárioAssalariado

```
package FTSAbstract;

public interface FuncionárioAssalariado
{
 void receberSalário(int nTurmas);
}
```

Exercício 3 - Diagrama de Classe da Classe Abstrata Professor com os métodos implementados na solução docente usando a ferramenta UMLet



Exercício 3 - Diagrama de Classe parcial disponibilizado no arquivo FulanoTS_parcial.uxf



Execução da classe Principal.java para o estudante com o nome fictício de PrimeiroNomedoEstudante

```
Professor também estuda!
Professor também estuda!
Oh vida, quanto estudo!
PrimeiroNomedoEstudante tirou 10.0 na prova
```