|  |  |
| --- | --- |
|  | **AARE**: **Paradigmas de Linguagens de Programação**  **Professor**: Dr. Ausberto S. Castro V.  **Orientação a Objetos** |

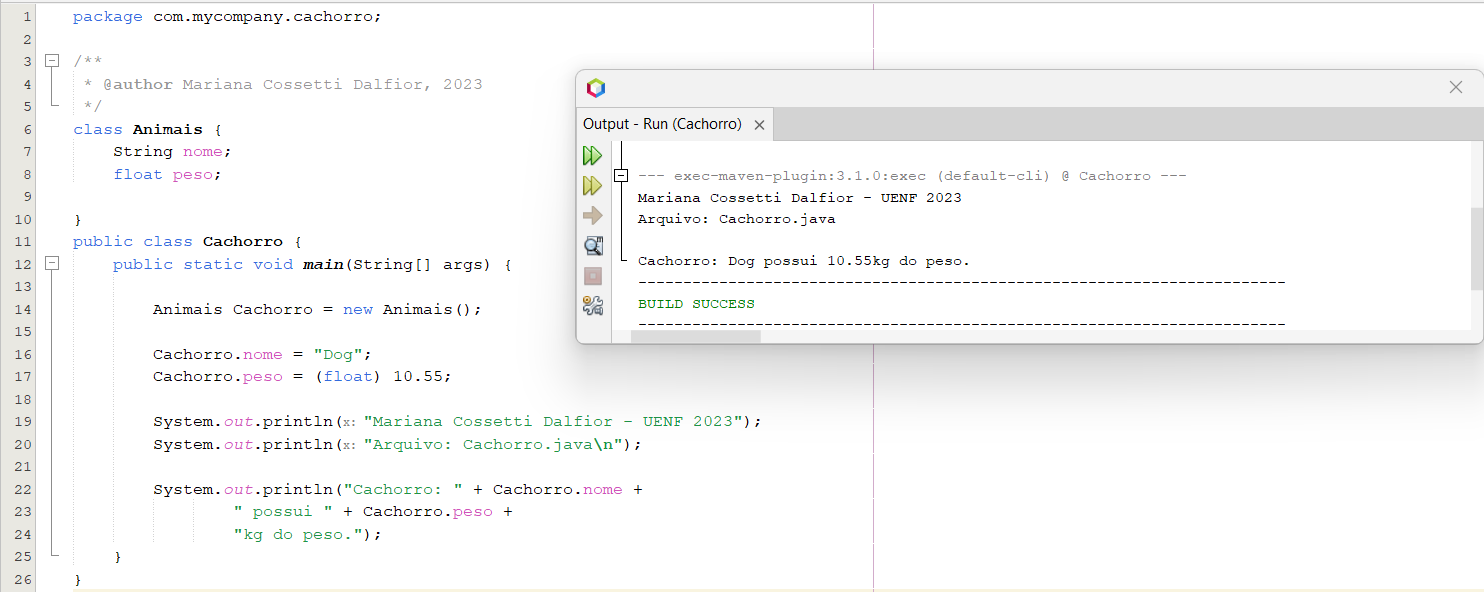
**Prova JAVA**

Nome Completo: Mariana Cossetti Dalfior

Data: 12 de junho de 2023

1. **(1 pontos)** Responda detalhadamente as seguintes questões (**exemplifique**):
   1. Qual é a diferença entre uma classe e um objeto?

R: Classe é um conceito que encapsula abstrações de atributos e métodos que descrevem o conteúdo e/ou comportamento de uma entidade do mundo real, já o objeto é a instanciação de uma classe. Isso é, a classe é um molde para a criação de variáveis com suas propriedades e ações, que são chamadas de objetos. Um exemplo de classe seria animais e um objeto dessa classe seria cachorro:



* 1. Explique como se define uma classe em Java.

R: Para a definição de uma classe em Java é utilizado a “class” seguido pelo nome que essa classe vai receber, podendo receber public, protected ou private como modificadores de acesso a essas classes para controlar a visibilidade dessas para outras classes. Uma classe geralmente define o estado e o comportamento de um objeto implementando seus atributos e seus métodos. Sintaxe de uma classe:

public class Classe {

tipo atributo1;

tipo atributo2;

public Classe(tipo atributo1, tipo atributo2) {

this.atributo1 = atributo1;

this.atributo2 = atributo2;

}

public void método1(){

}

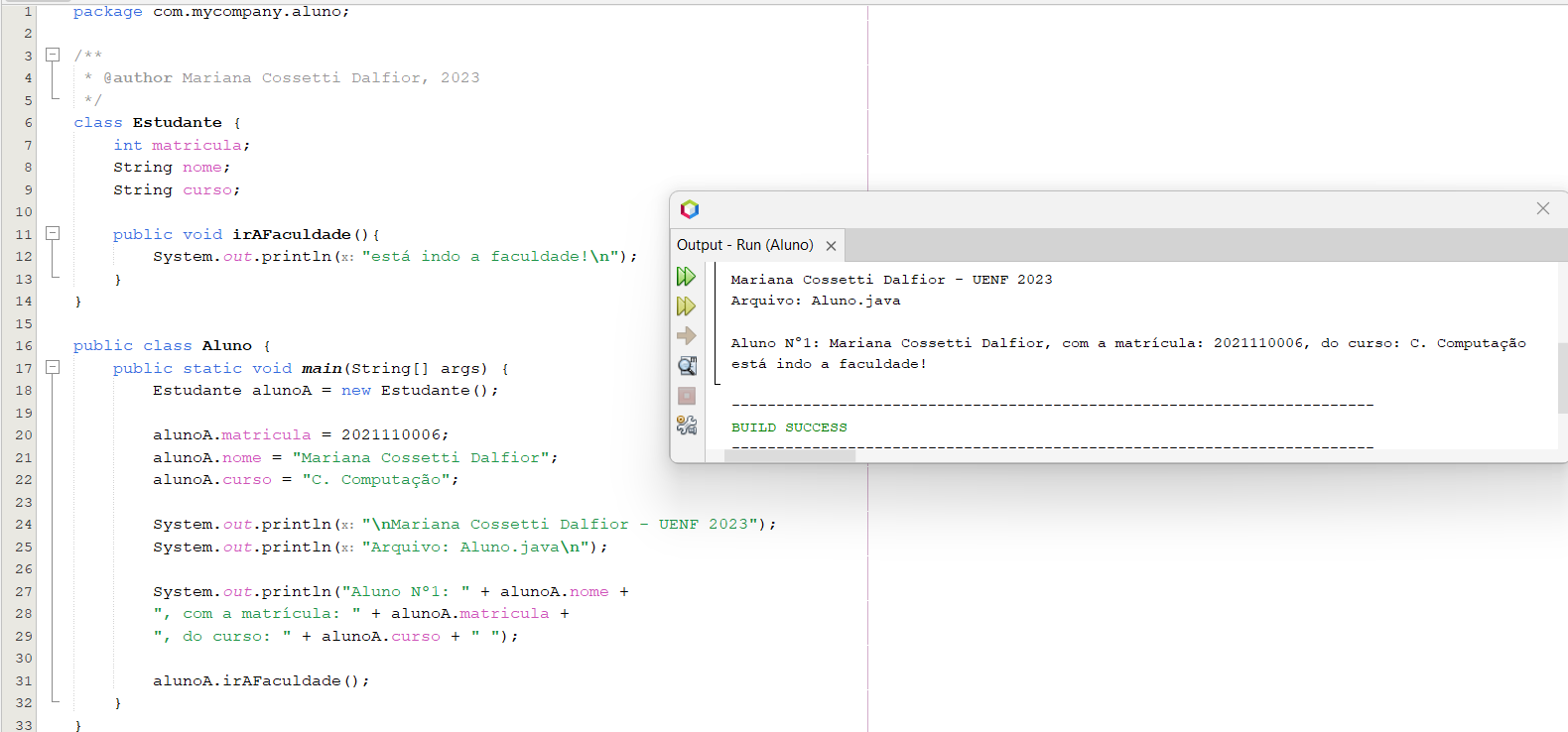
public tipo método2(){

return valor;

}

}

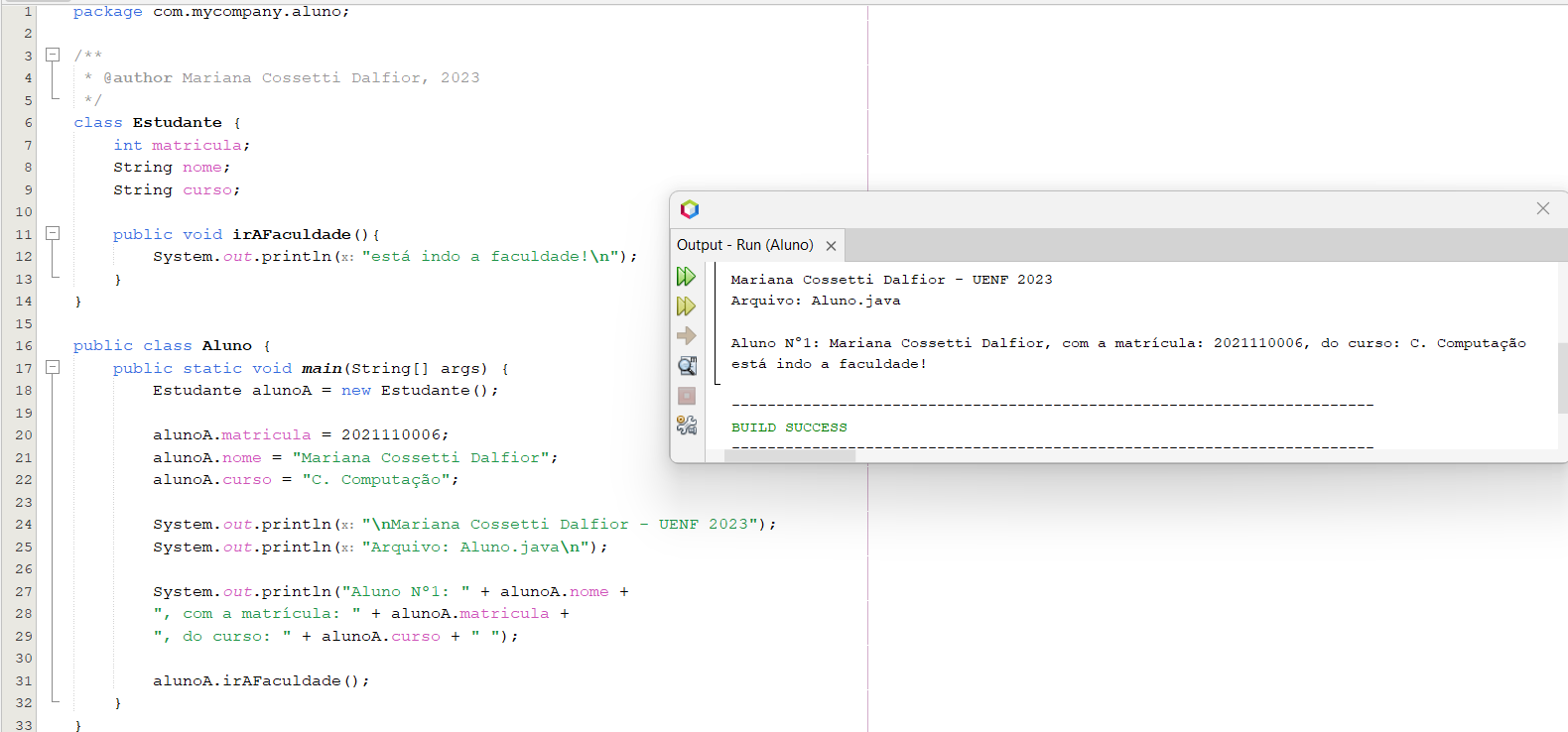
Exemplo:



* 1. Por que uma classe é uma abstração lógica?

R: Porque para conseguir se concentrar nos aspectos essenciais de uma entidade é necessário isolar os atributos comuns relevantes de um conjunto. Assim, podendo especializar uma classe utilizando a herança e o polimorfismo.

Exemplo:



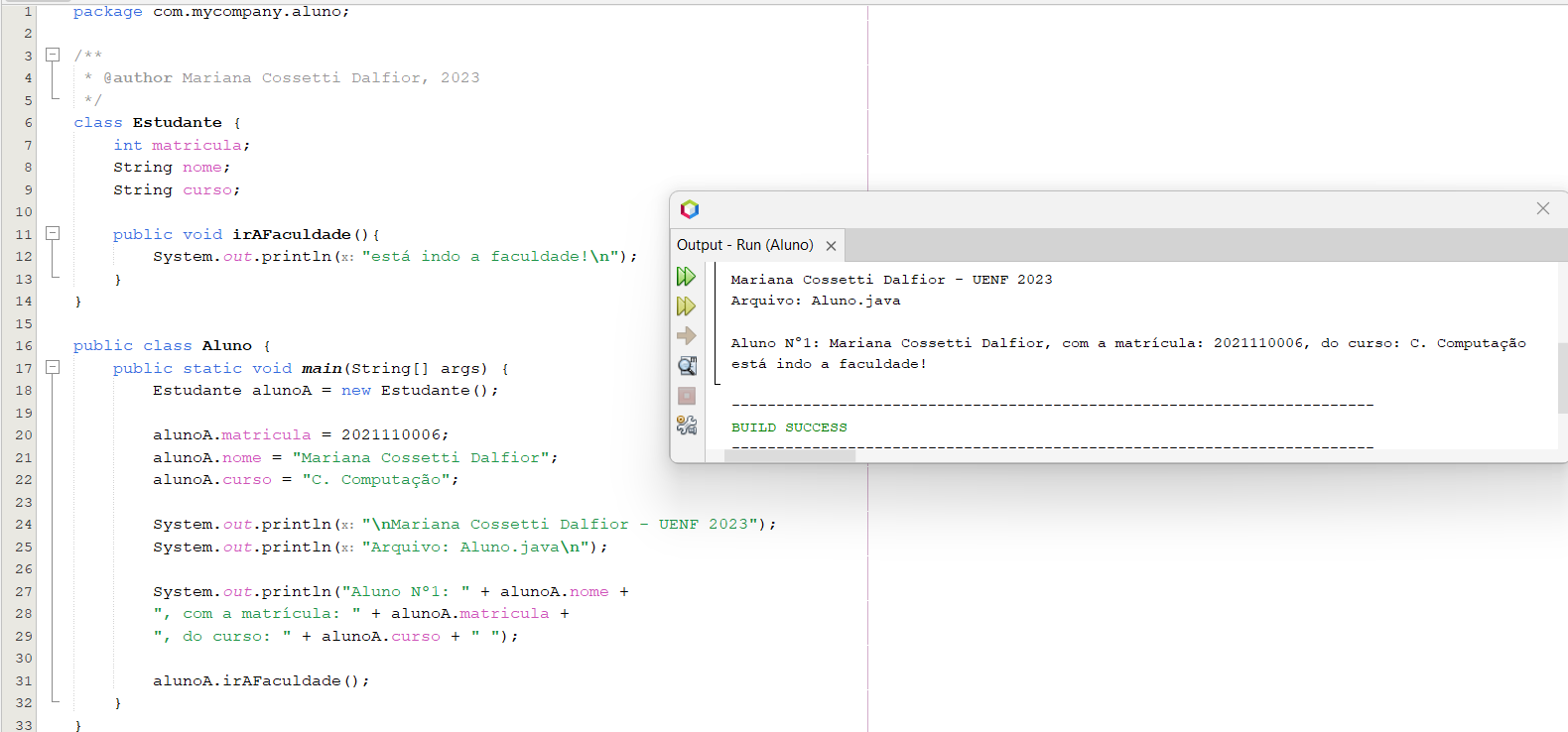
Como podemos perceber ao criar a classe Estudante foram criados também atributos para ela, em que esses possuem características comuns e relevantes que todos os estudantes possuem.

* 1. Como os objetos são criados?

R: Um objeto é criado com base em sua classe. Para realizar essa instância é utilizada a palavra reservada new. No caso da criação de um método construtor é necessário a passagem dos parâmetros desse método dentro da instanciação.

Classe objeto = new Classe();

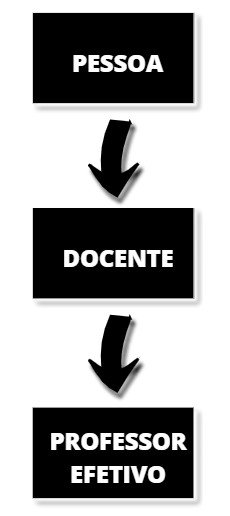
Exemplo:



* 1. O que é coleta de lixo e como ela funciona?

R: Coletor de lixo ou Garbage collector é um processo utilizado para gerenciar memória na JVM, assim garantindo melhor desempenho e escalabilidade. Com ele é possível recuperar uma área de memória inutilizada pelo programa e evitar problemas de esgotamento de memória. Nesse processo é examinada a memória, identificando quais objetos estão em uso (objetos referenciados) e quais estão inutilizados (objetos não referenciados), assim, excluindo os objetos não utilizados e liberando memória de forma automática.

1. **(1 pontos)** Desenhar todas as classes, subclasses e superclasses associadas aos seguintes objetos, indicando em cada caso, pelo menos 2 atributos e 2 métodos, e a relação de herança entre classes
   1. Docente



Classe: Pessoa (superclasse)

Atributos: nome, idade;

Métodos: andar(), comer()

Classe: Docente (subclasse de pessoa e superclasse de professor efetivo)

Atributos: disciplina, cargahoraria;

Métodos: ensinar(), aplicarProva();

Classe: Professor efetivo (subclasse)

Atributos: cadastro, especializacao;

Métodos: orientarAluno(), pesquisar() ;

Relações de herança:

A classe "Professorefetivo" herda da classe "docente" que herda da classe "Pesssoa", logo Pessoa é super classe, Docente subclasse de "Pessoa", e Professorefetivo também é uma subclasse.

Pessoa (superclasse) < - Docente (subclasse)

Docente (superclasse) < - Professorefetivo (subclasse)

* 1. Verdura



Classe: Planta

Atributos: nome, familia;

Métodos: crescer(), respirar()

Classe: Verdura

Atributos: sabor, especie;

Métodos: preparar(), cozinhar();

Classe: Folha

Atributos: cor, textura;

Métodos: lavar(), cortar() ;

Relações de herança:

A classe “Folha” herda da classe "Verdura" que herda da classe "Planta", logo Planta é super classe, Verdura é uma subclasse de "Planta", e Folha também é uma subclasse.

Planta (superclasse) < - Verdura (subclasse)

Verdura (superclasse) < - Folha (subclasse)

* 1. Ave



Classe: Animais

Atributos: nome, porte;

Métodos: comer(), respirar()

Classe: Ave

Atributos: tamanho\_asa, especie;

Métodos: voar(), emitir\_som();

Classe: Aveaquatica

Atributos: bico, habitat;

Métodos: nadar(), comer\_peixe() ;

Relações de herança:

A classe “Aveaquatica” herda da classe " Ave " que herda da classe " Animais ", logo Animais é super classe, Ave é uma subclasse de " Animais ", e Aveaquatica também é uma subclasse.

Animais (superclasse) < - Ave (subclasse)

Ave (superclasse) < - Aveaquatica (subclasse)

* 1. Computador



Classe: Eletronicos

Atributos: modelo, preco;

Métodos: ligar(), desligar()

Classe: Computador

Atributos: marca, processador;

Métodos: rodar\_programa(), inicializar();

Classe: Notebook

Atributos: bateria, memoria;

Métodos: carregar(), abrir() ;

Relações de herança:

A classe “Notebook” herda da classe "Computador " que herda da classe " Eletronicos ", logo Eletronicos é super classe, Computador é uma subclasse de " Eletronicos ", e Notebook também é uma subclasse.

Eletronicos (superclasse) < - Computador (subclasse)

Computador (superclasse) < - Notebook (subclasse)

* 1. Rede



Classe: Rede

Atributos: ip, velocidade;

Métodos: conectar(), desconectar()

Classe: Roteador

Atributos: portas, preco;

Métodos: configurar(), monitorar();

Classe: Servidor

Atributos: armazenamento, funcoes;

Métodos: processar(), fornecer() ;

Relações de herança:

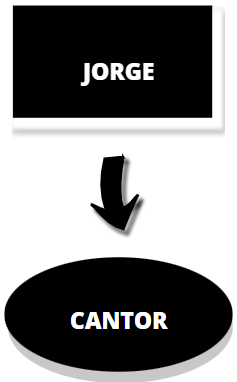
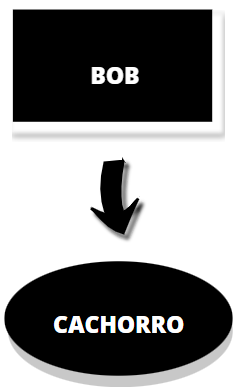
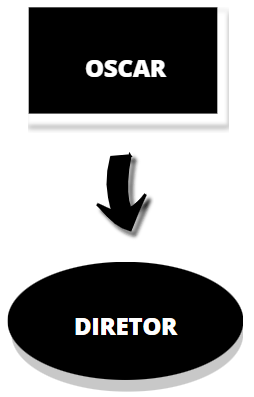
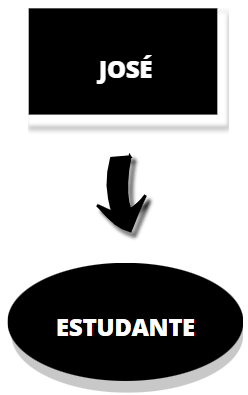
A classe “Servidor” herda da classe " Roteador" que herda da classe " Rede ", logo Rede é super classe, Roteador é uma subclasse de " Rede", e Servidor também é uma subclasse.

Rede (superclasse) < - Roteador (subclasse)

Roteador (superclasse) < - Servidor (subclasse)

1. **(01 ponto)** Dar cinco exemplos de cada operação abstrata em orientação a objetos. Ilustre com imagens ou diagramas.

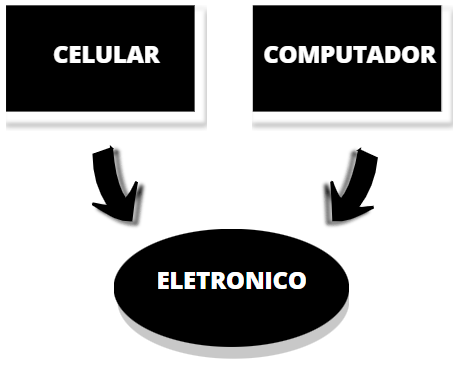
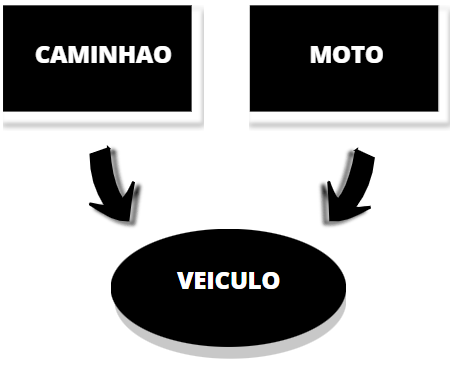
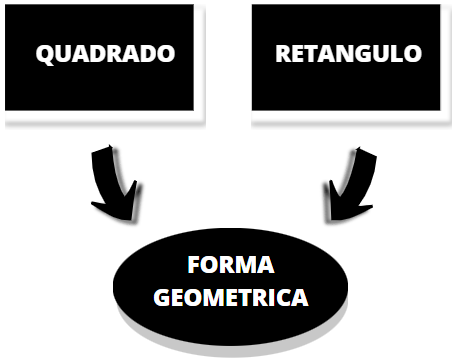
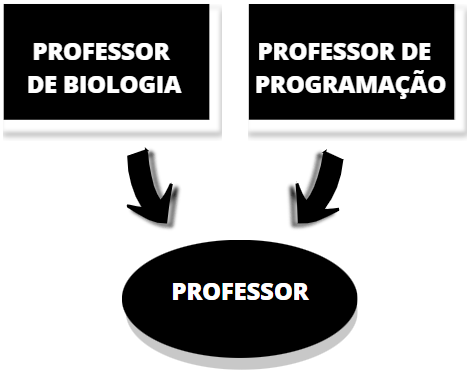
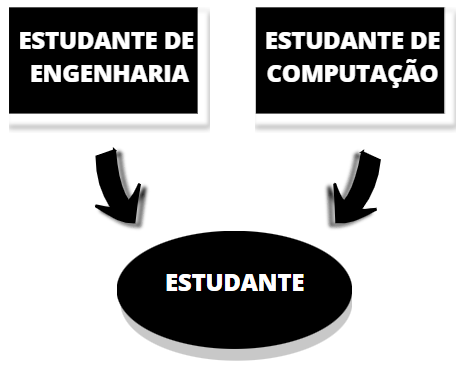
Classificação:



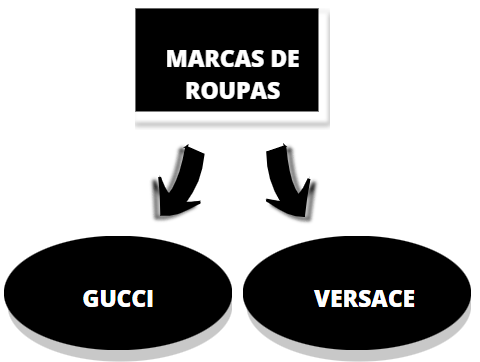
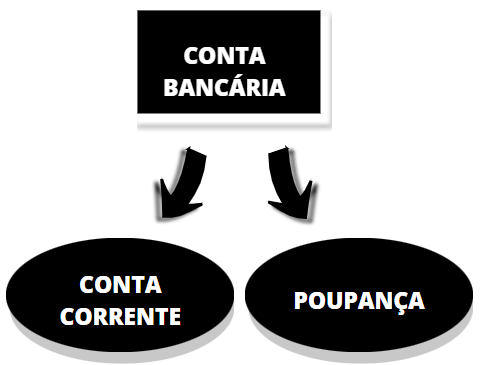
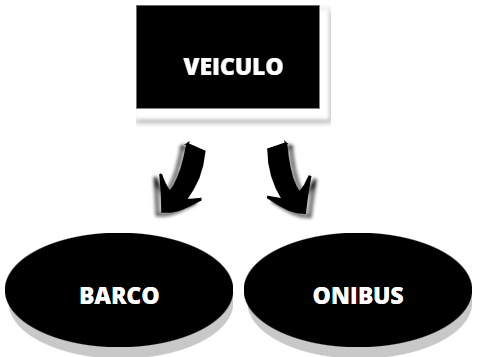
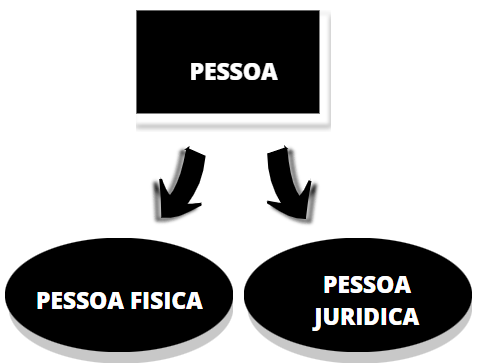
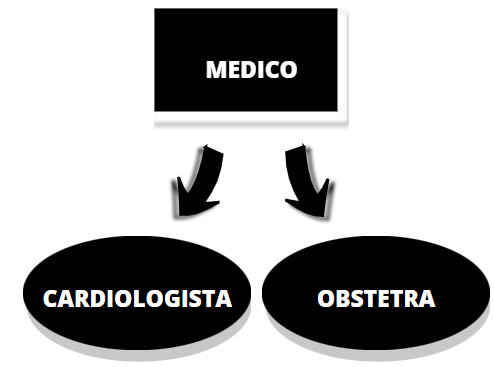
Instanciação:



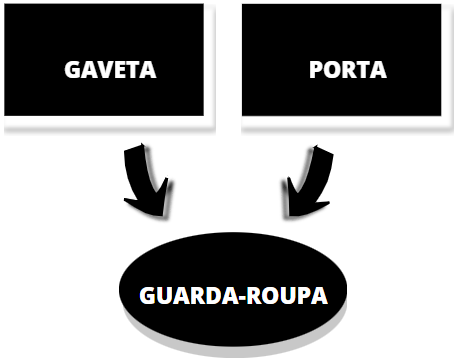
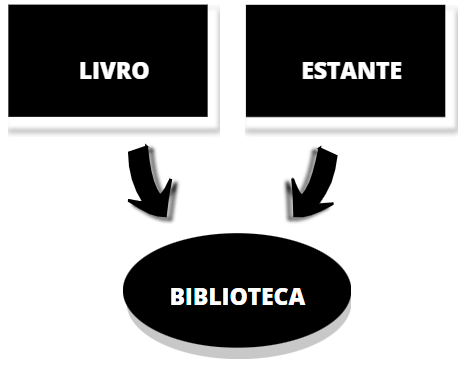
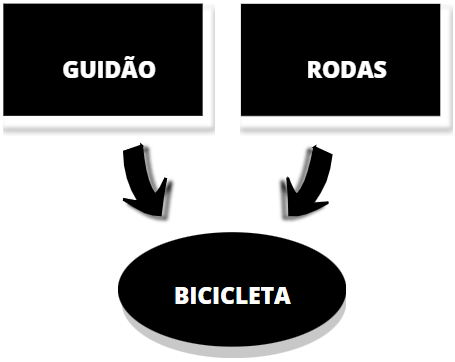
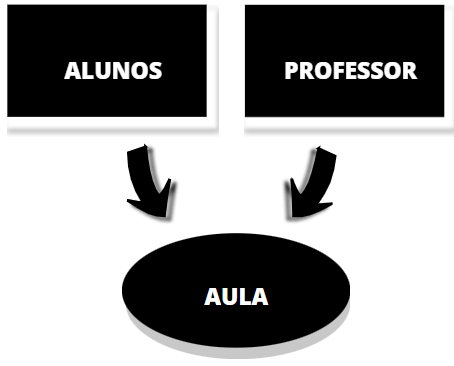
Generalização:



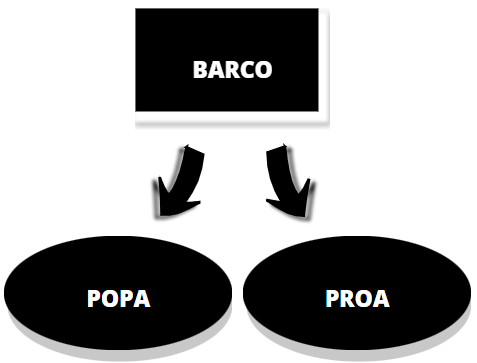
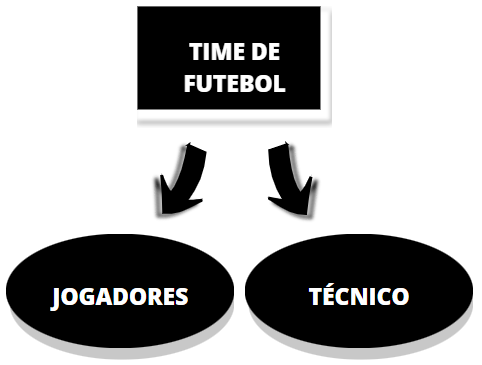
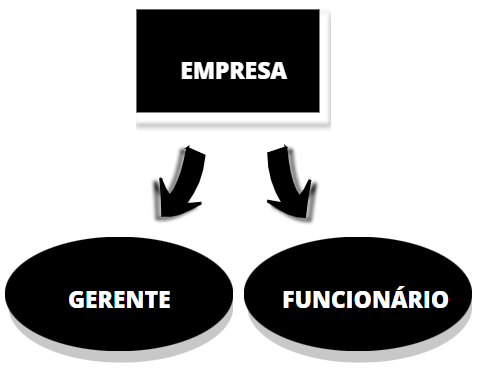
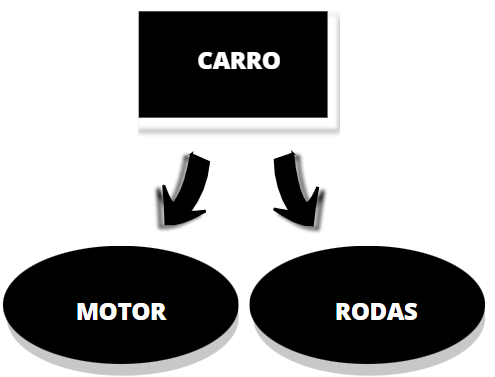
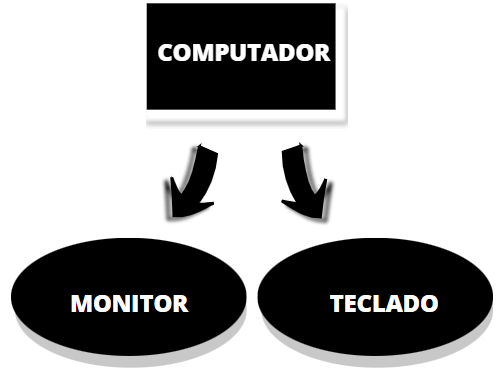
Especialização:



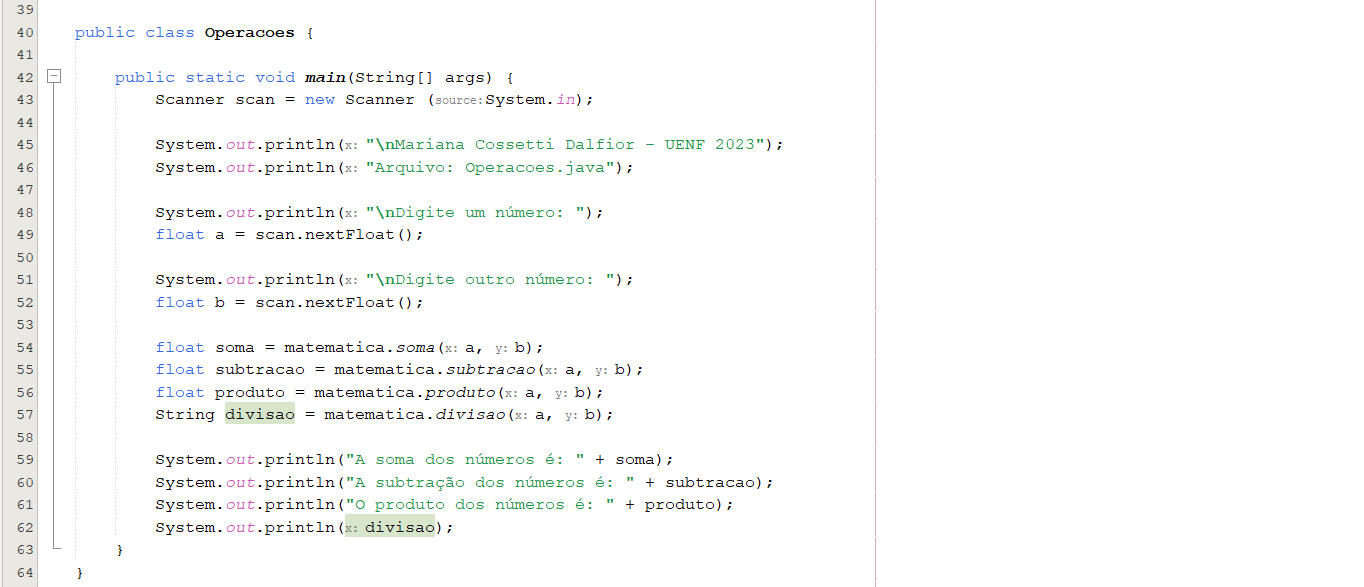
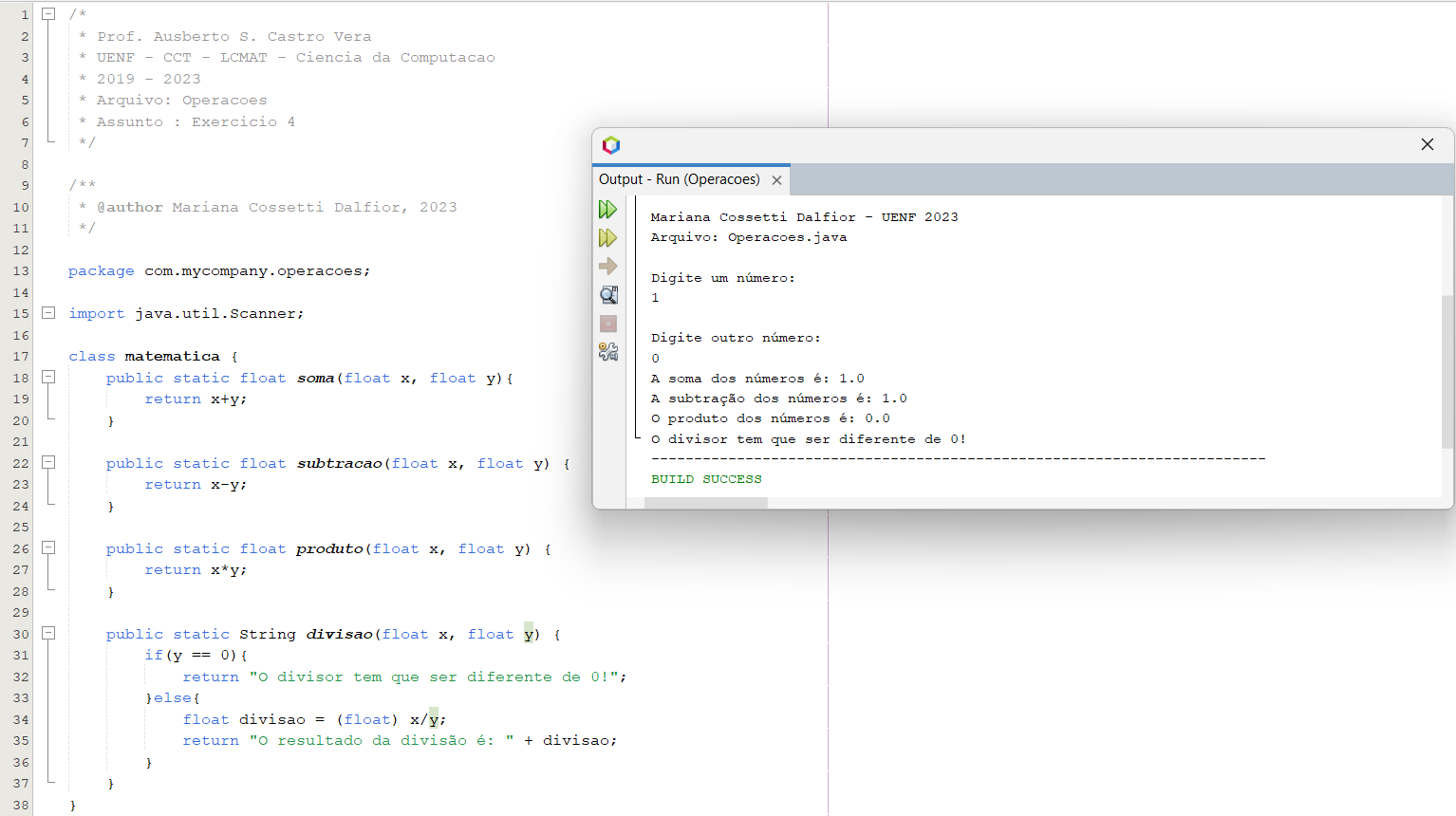
Agregação:



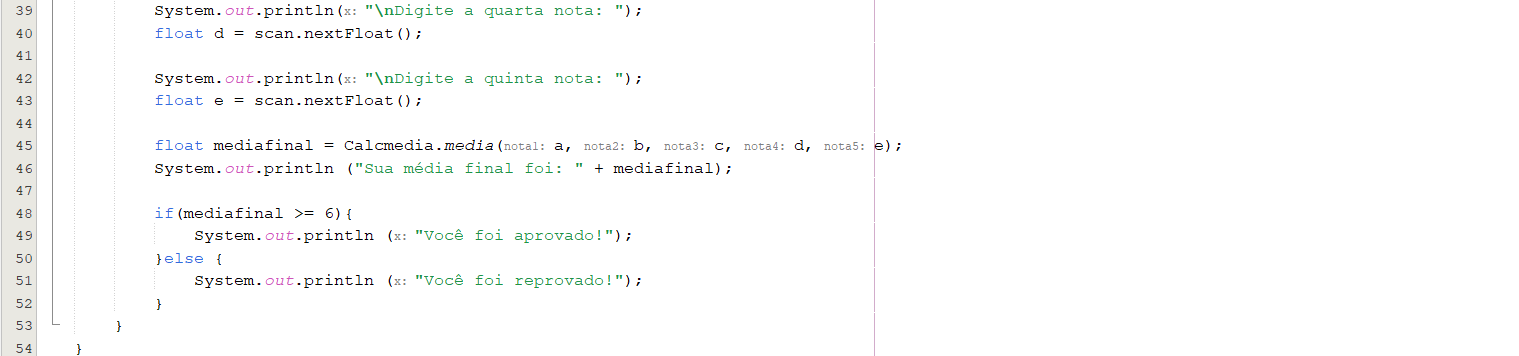
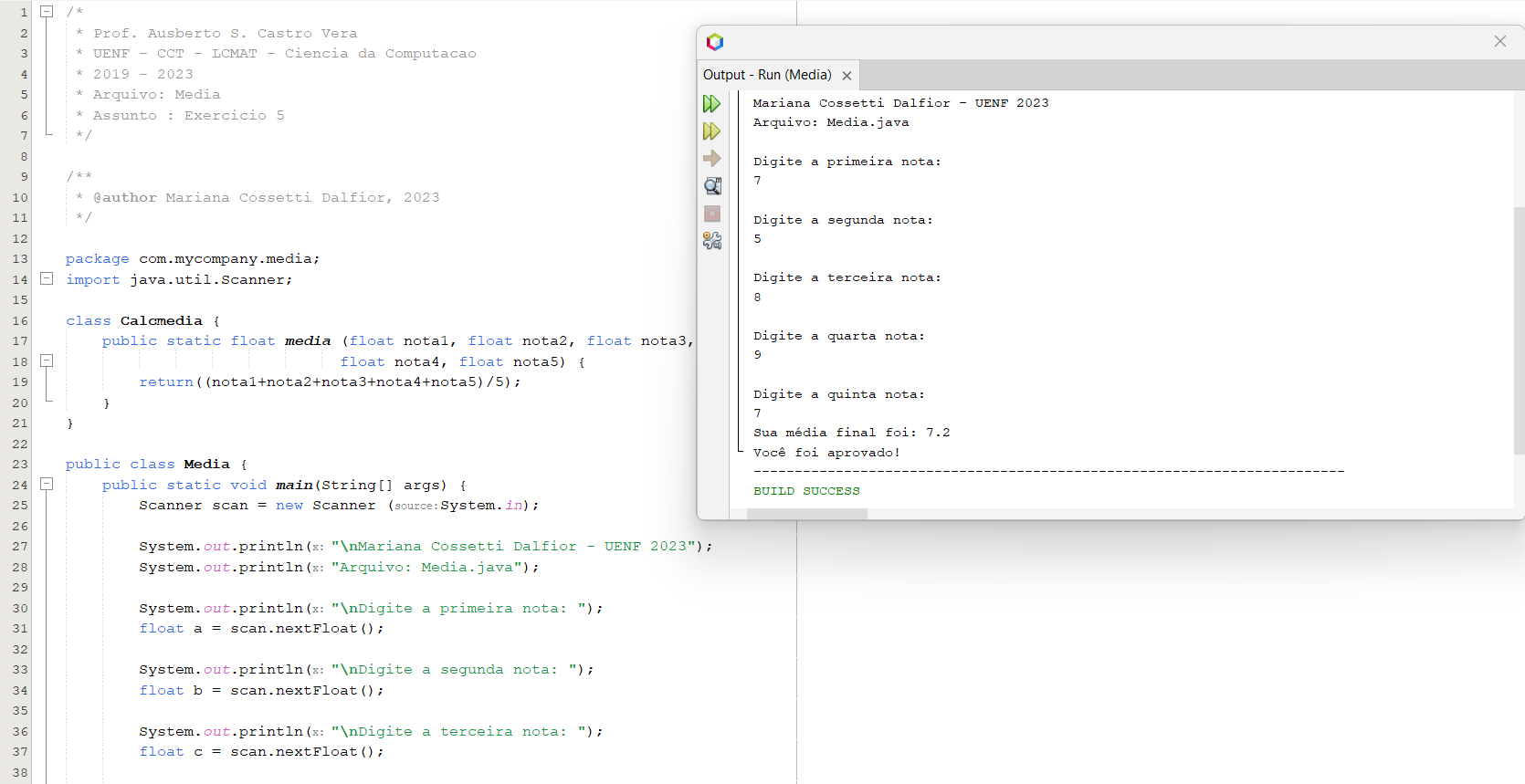
Decomposição:



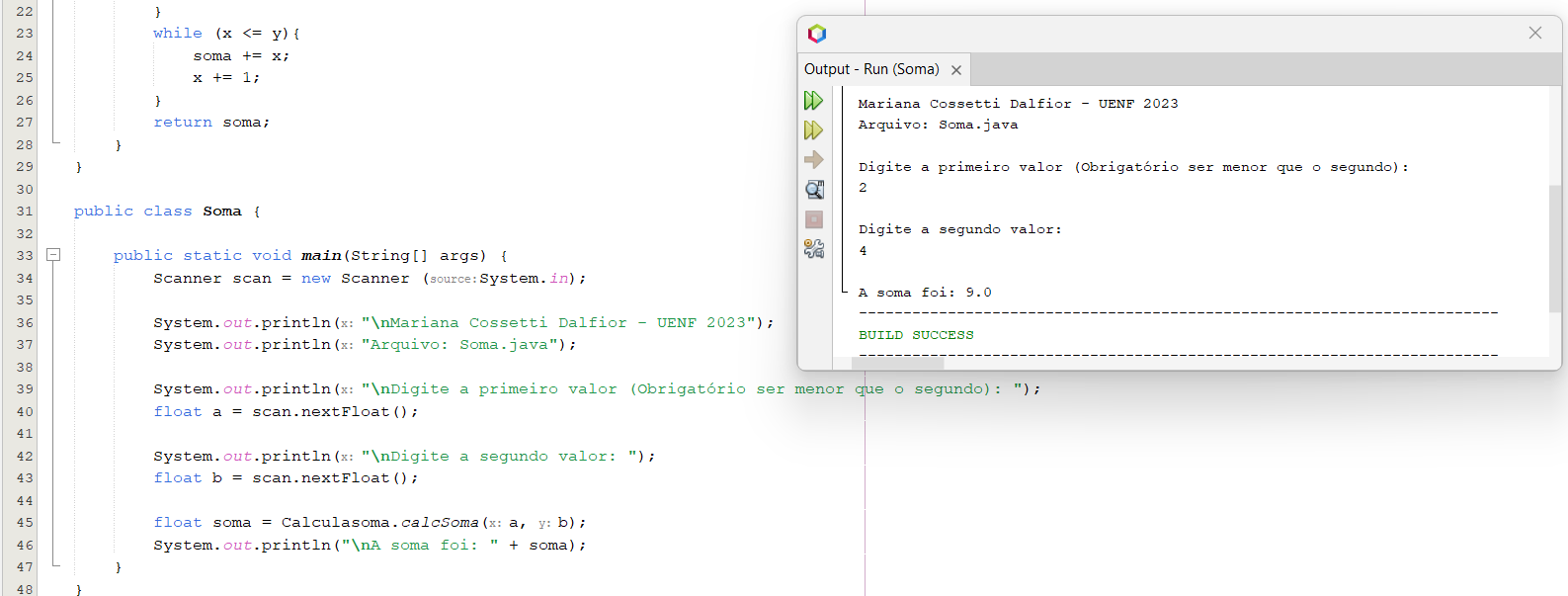
1. **(01 ponto)** Escreva um programa em Java que execute as quatro operações aritméticas de números reais.



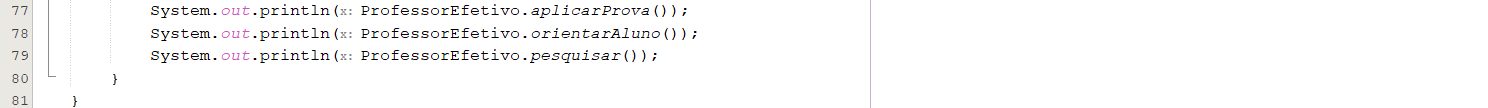
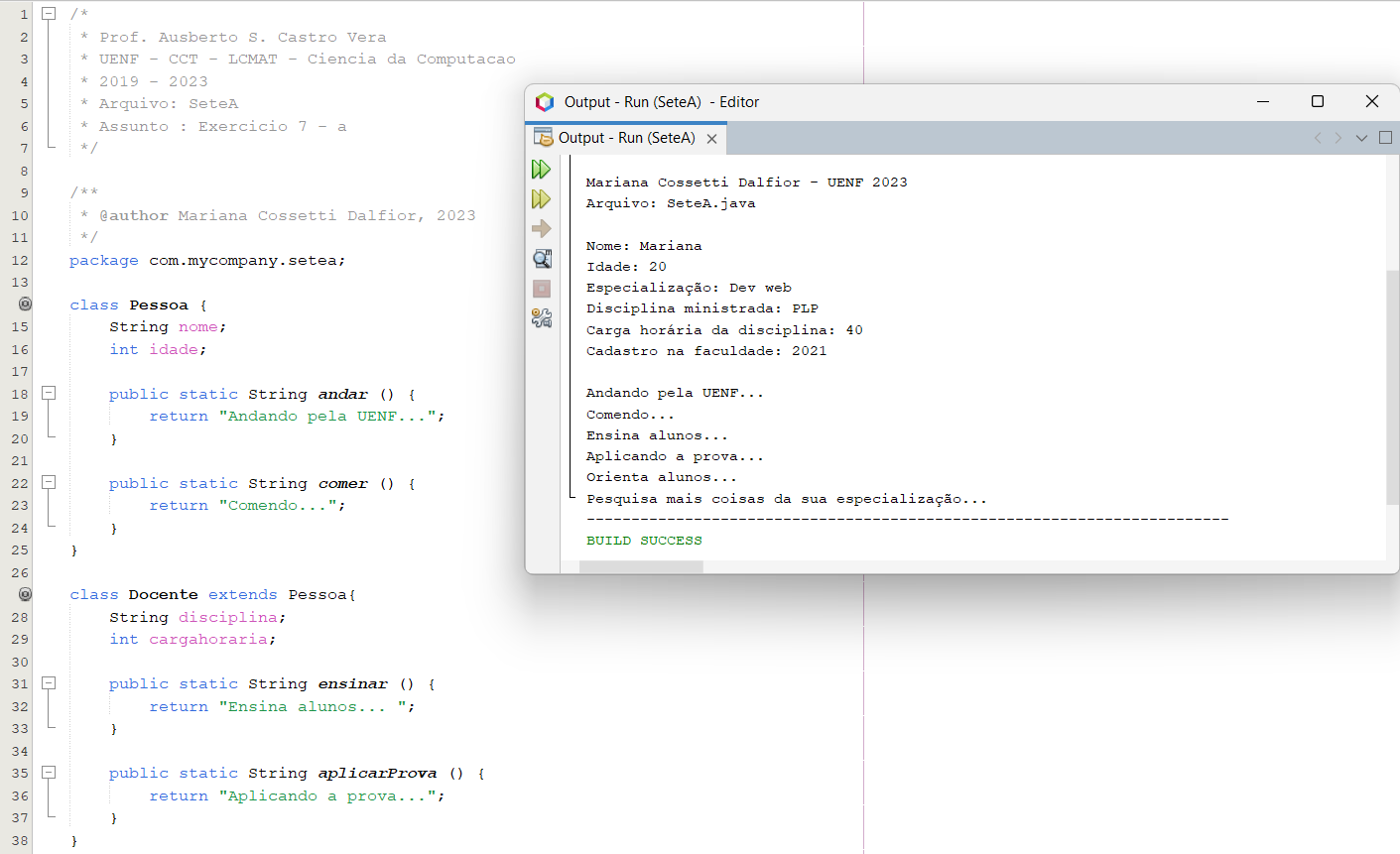
1. **(01 ponto)** Escreva um programa JAVA que calcule a média de cinco notas. O programa deve mostrar o valor da média e indicar se foi aprovado ou reprovado.



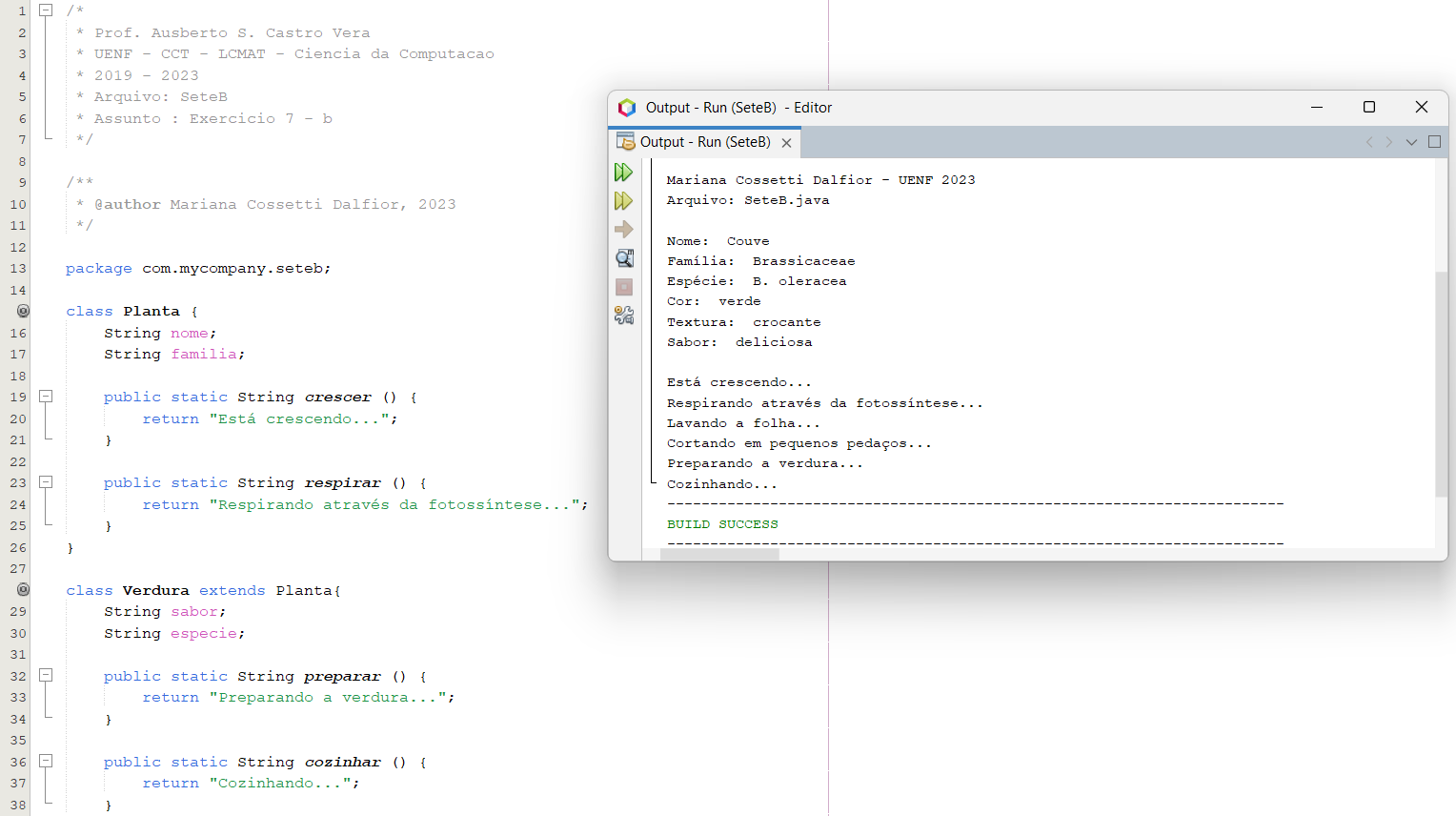
1. **(01 ponto)** Escreva um programa JAVA que calcule a soma dos inteiros maiores o iguales a um número inteiro A e menores ou iguais a um inteiro B.



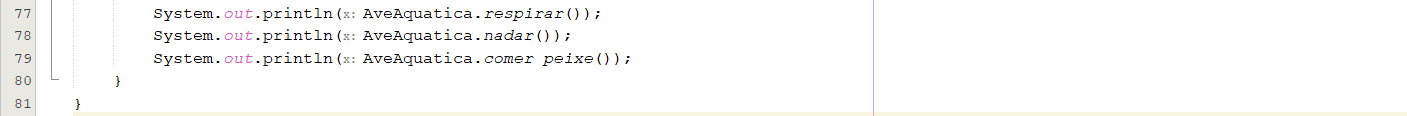
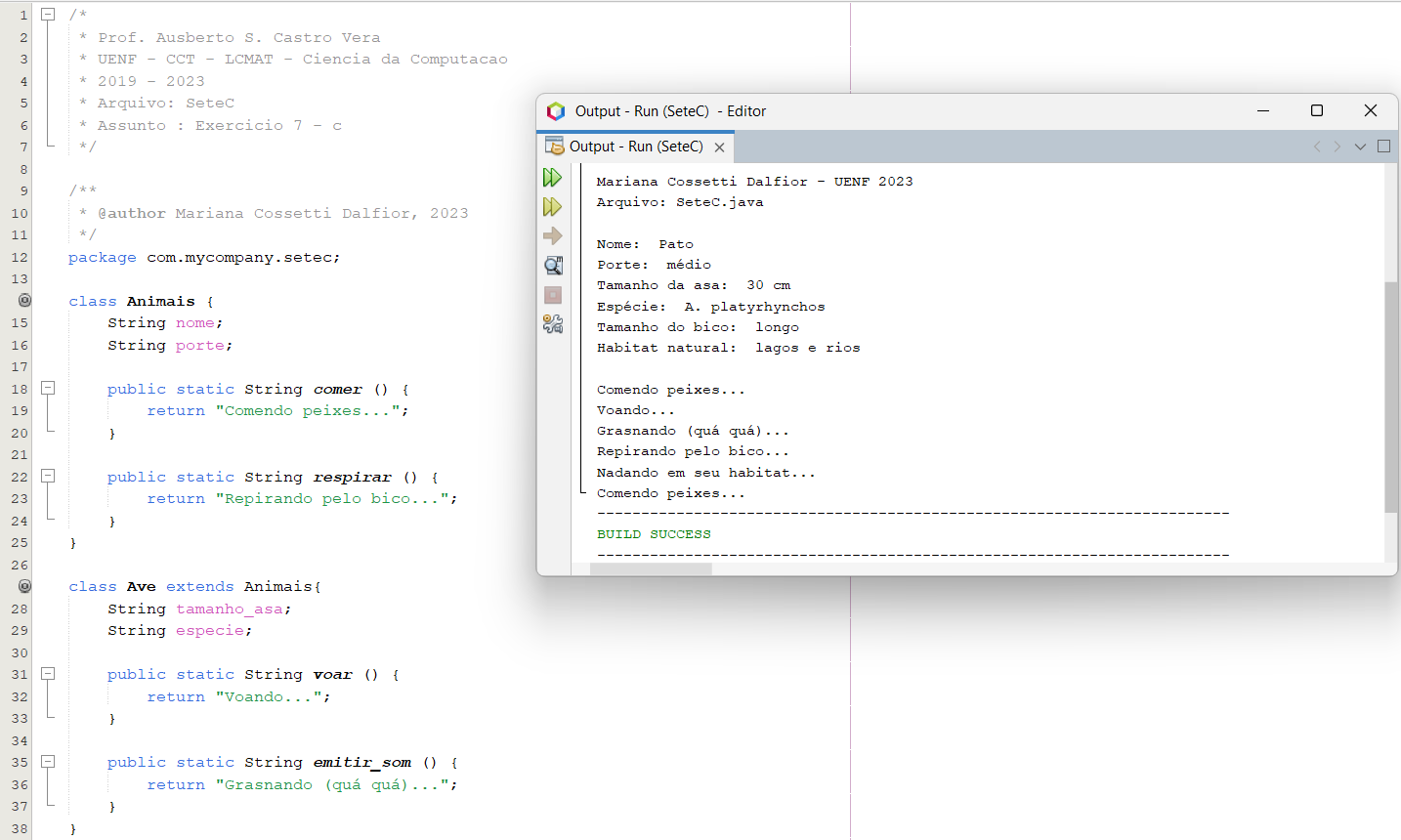
1. **(2 pontos)** Para cada caso da questão 2, escrever um programa JAVA, mostrando as classes, as operações (métodos), a criação de objetos e outras características da Programação Orientada a Objetos.
2. Docente:



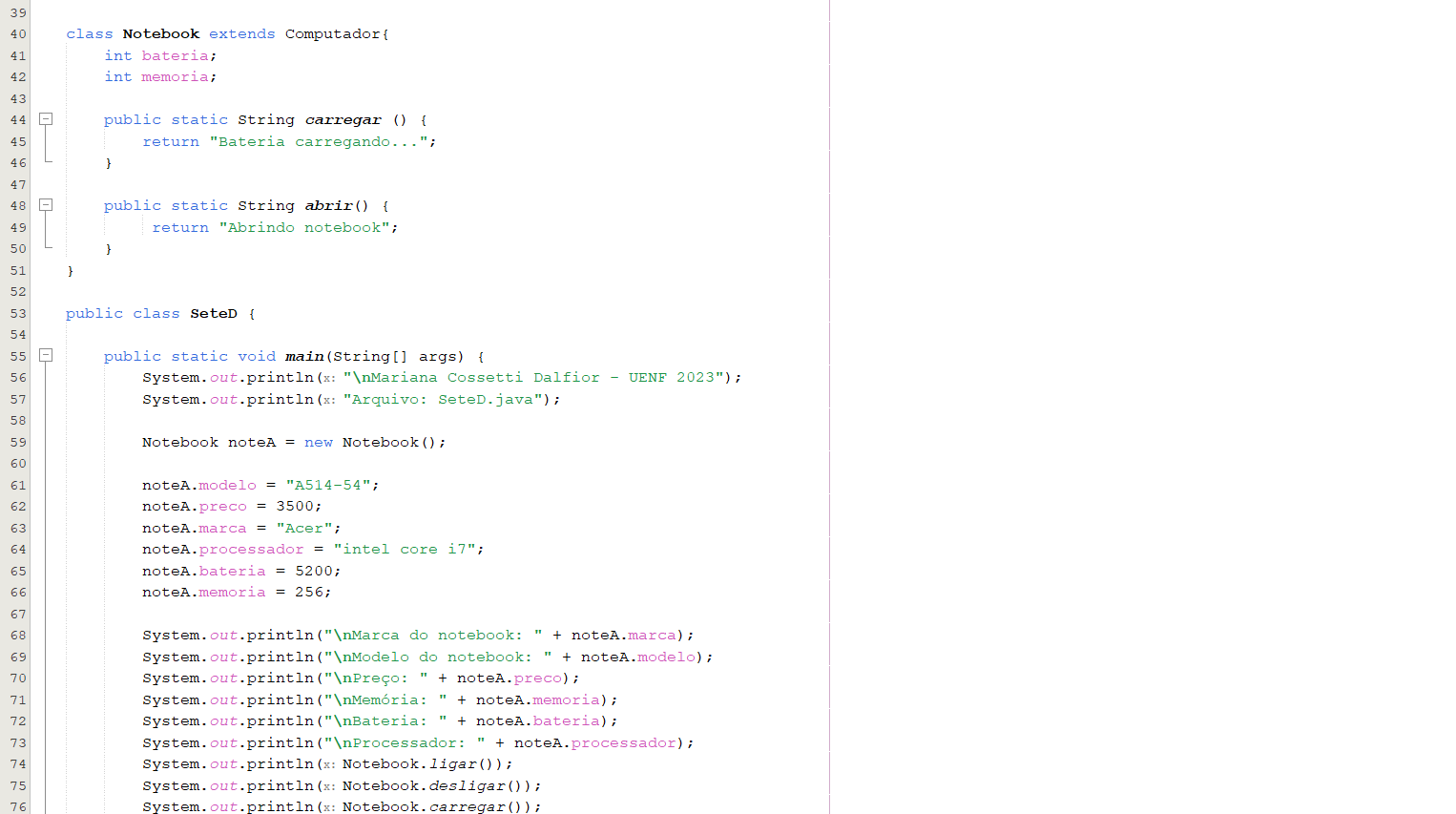
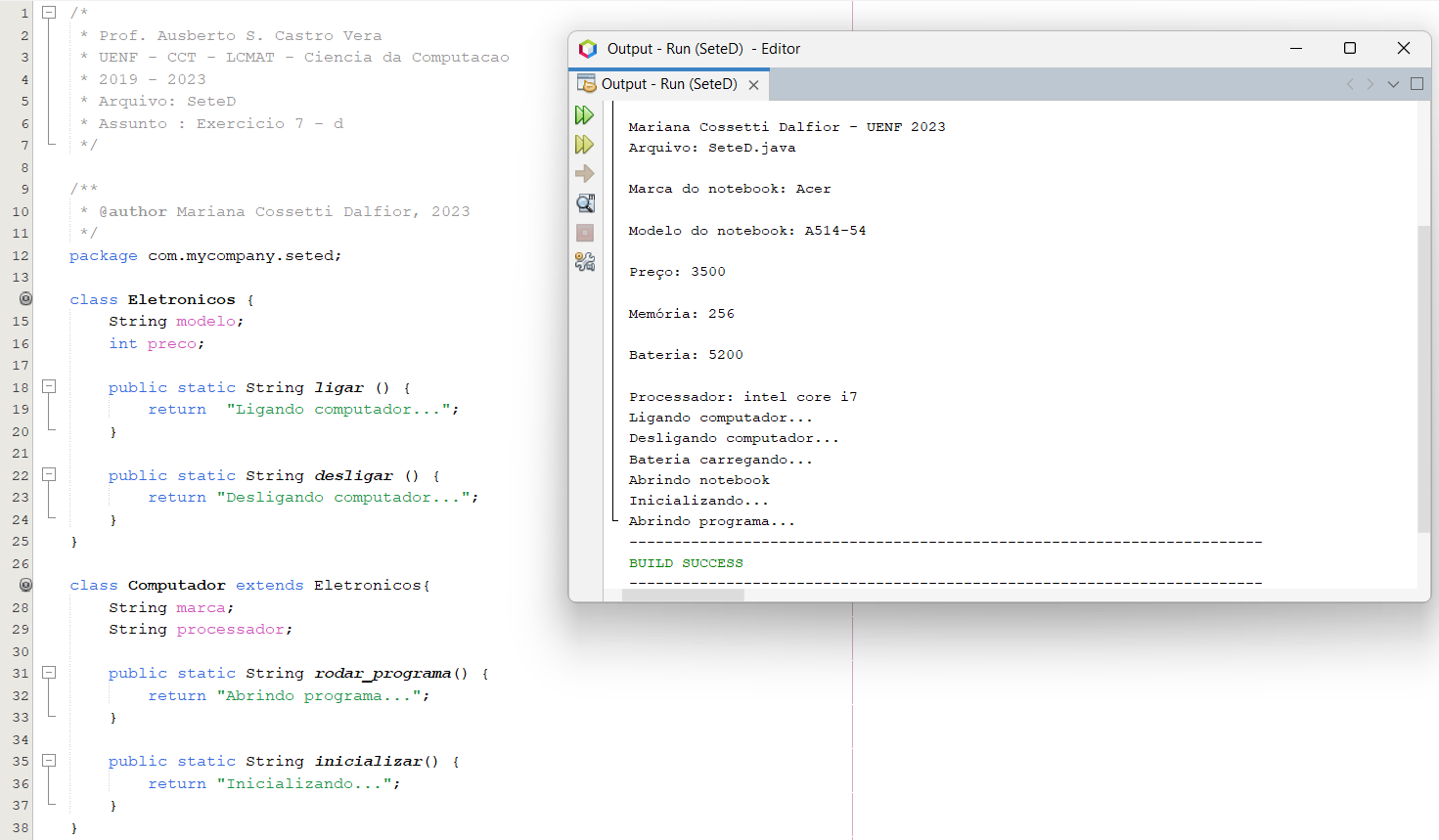
1. Verdura:



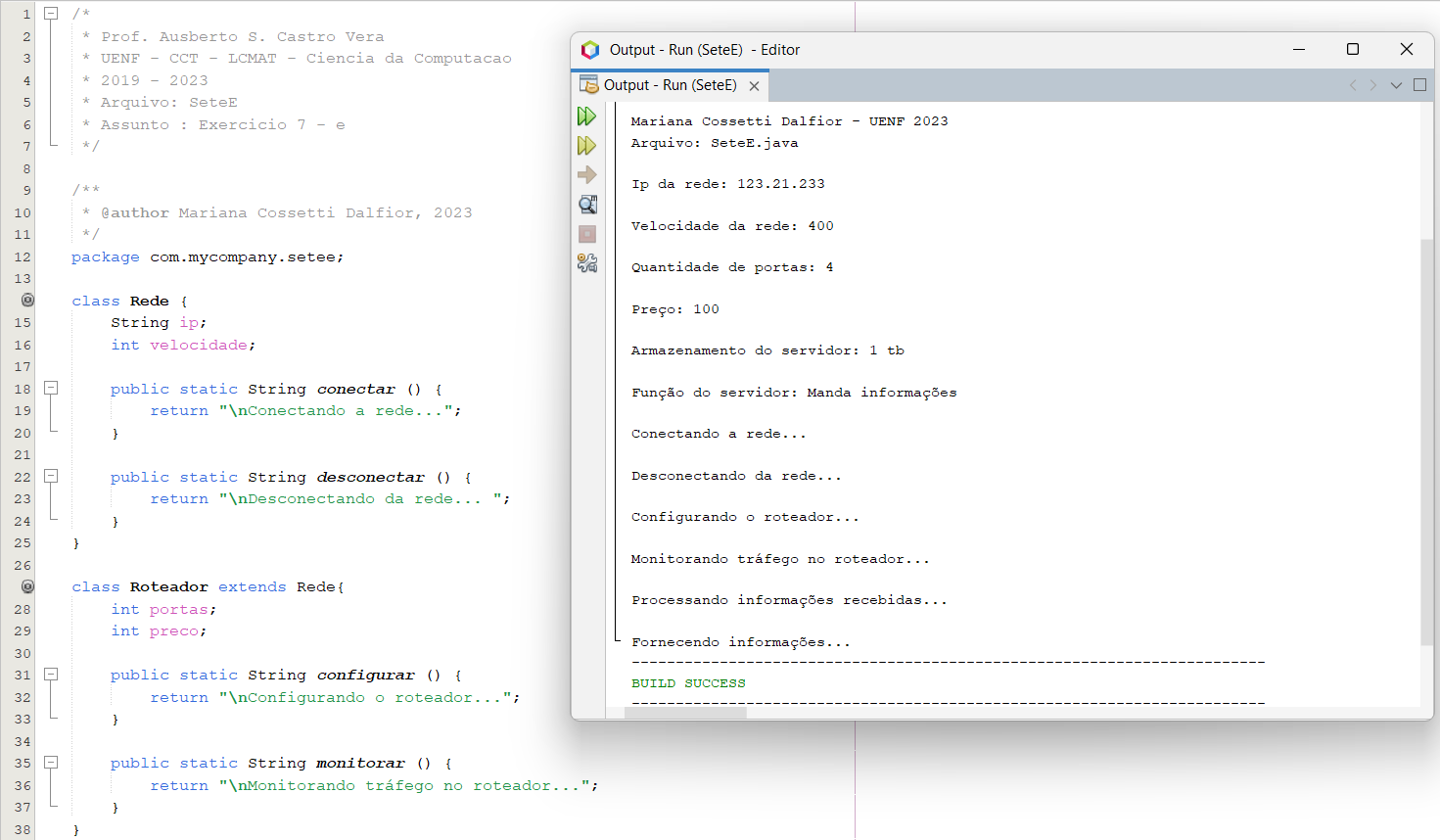
1. Ave:



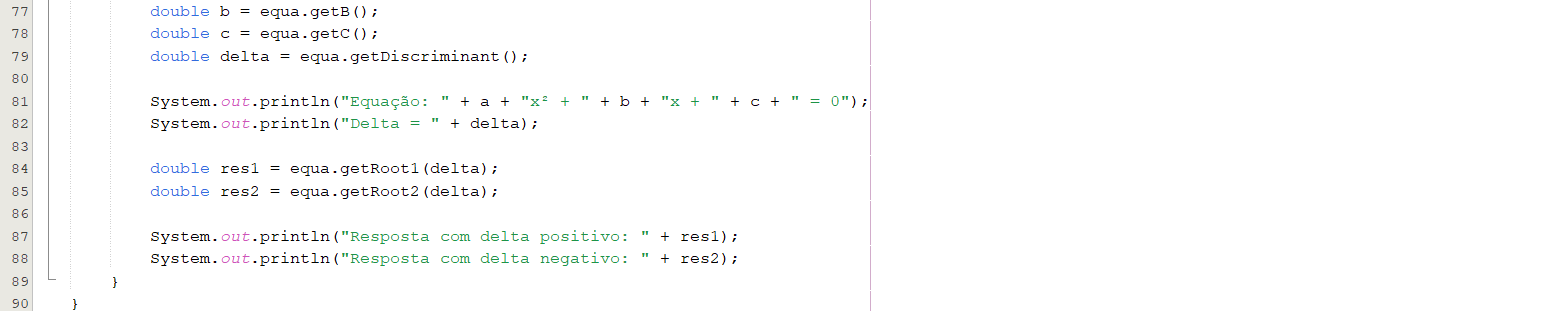
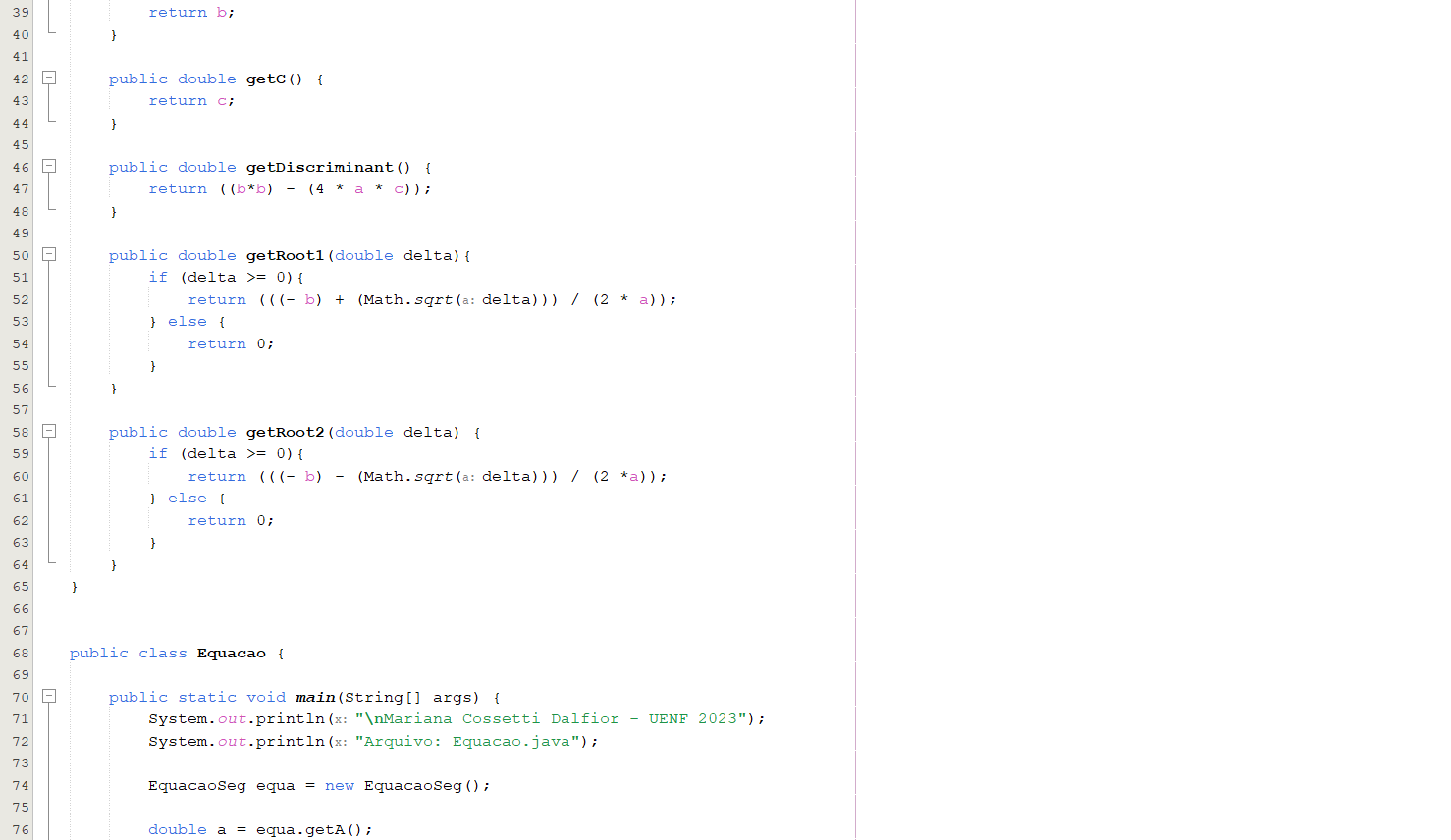
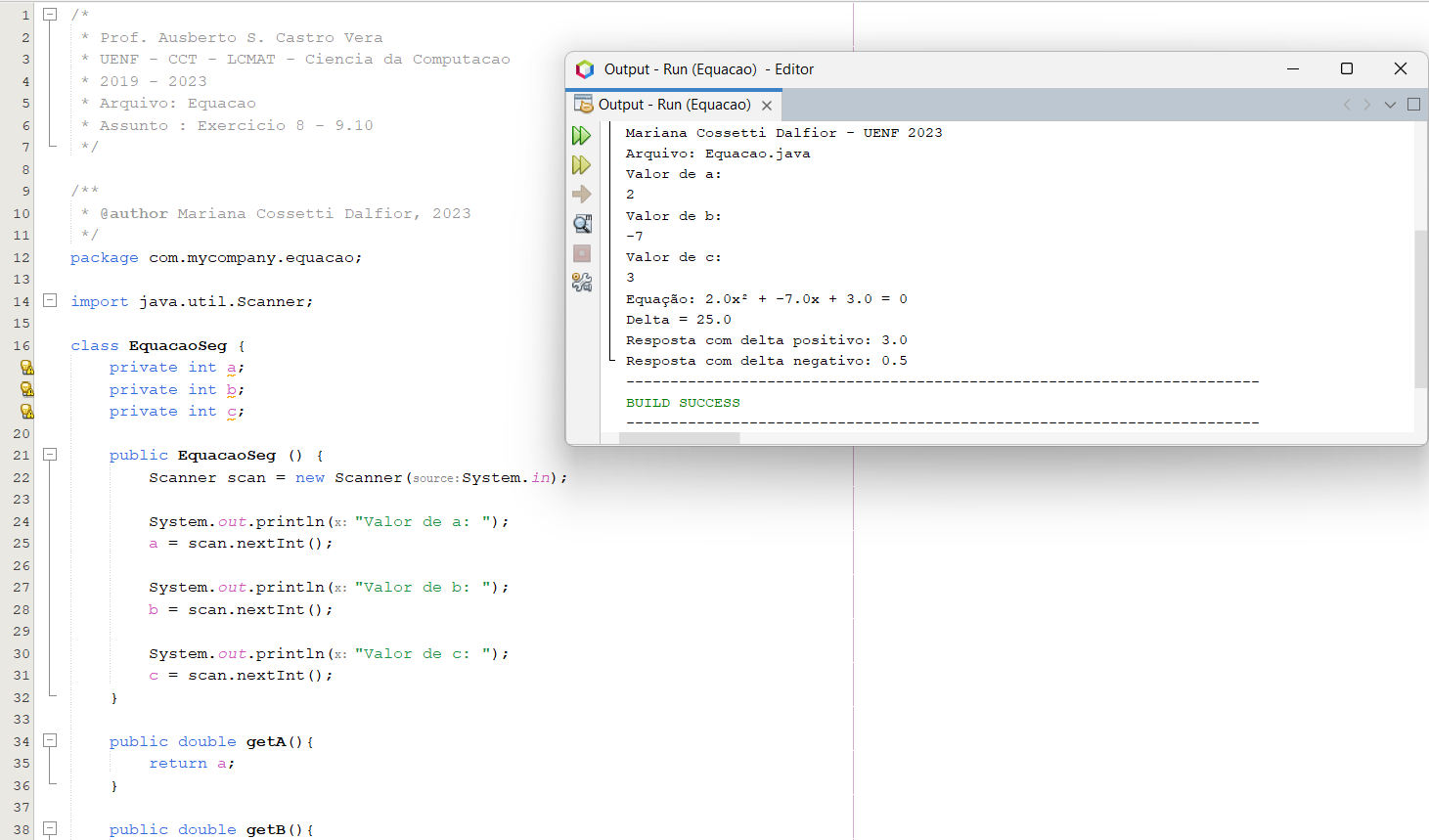
1. Computador:



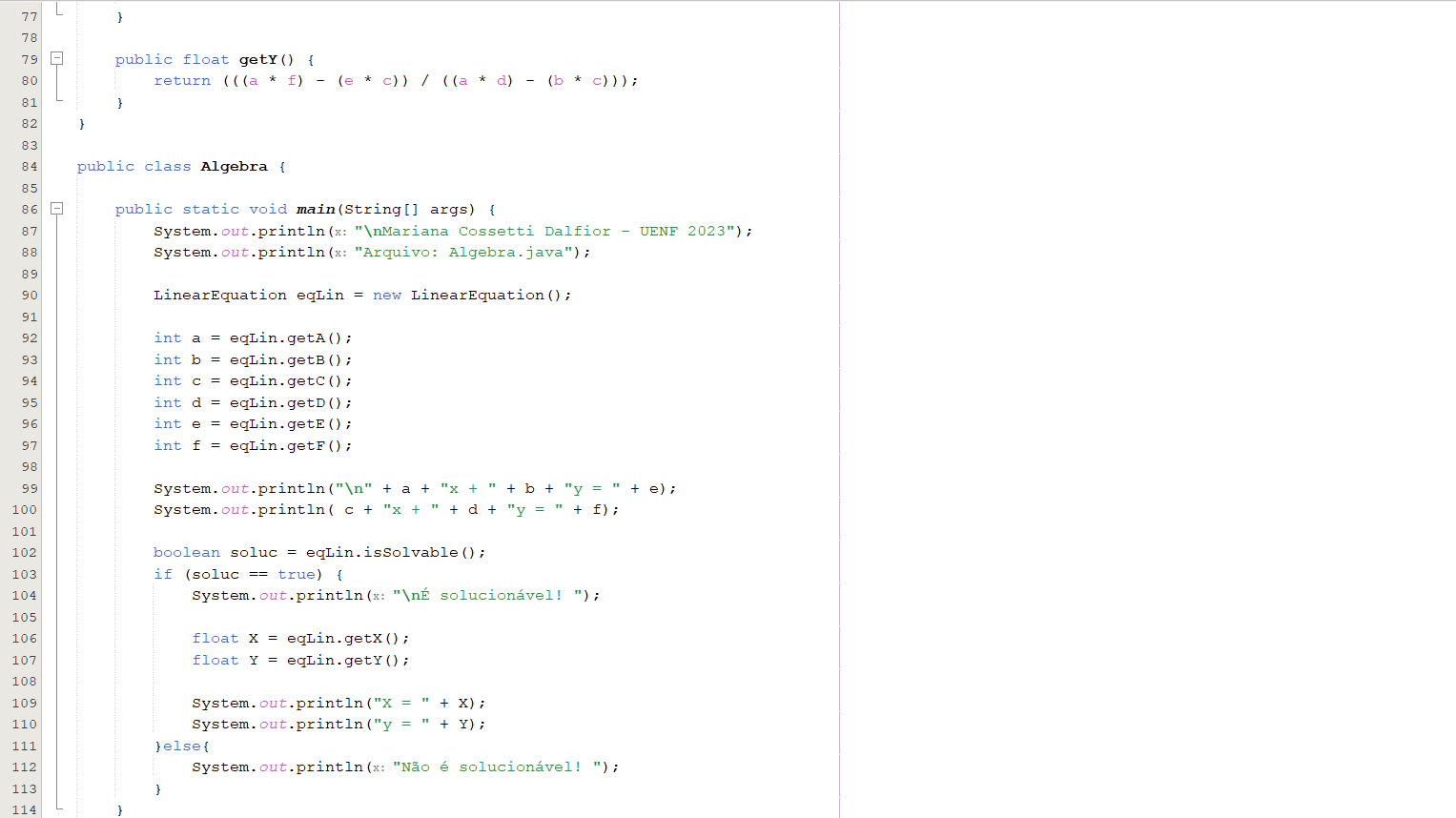
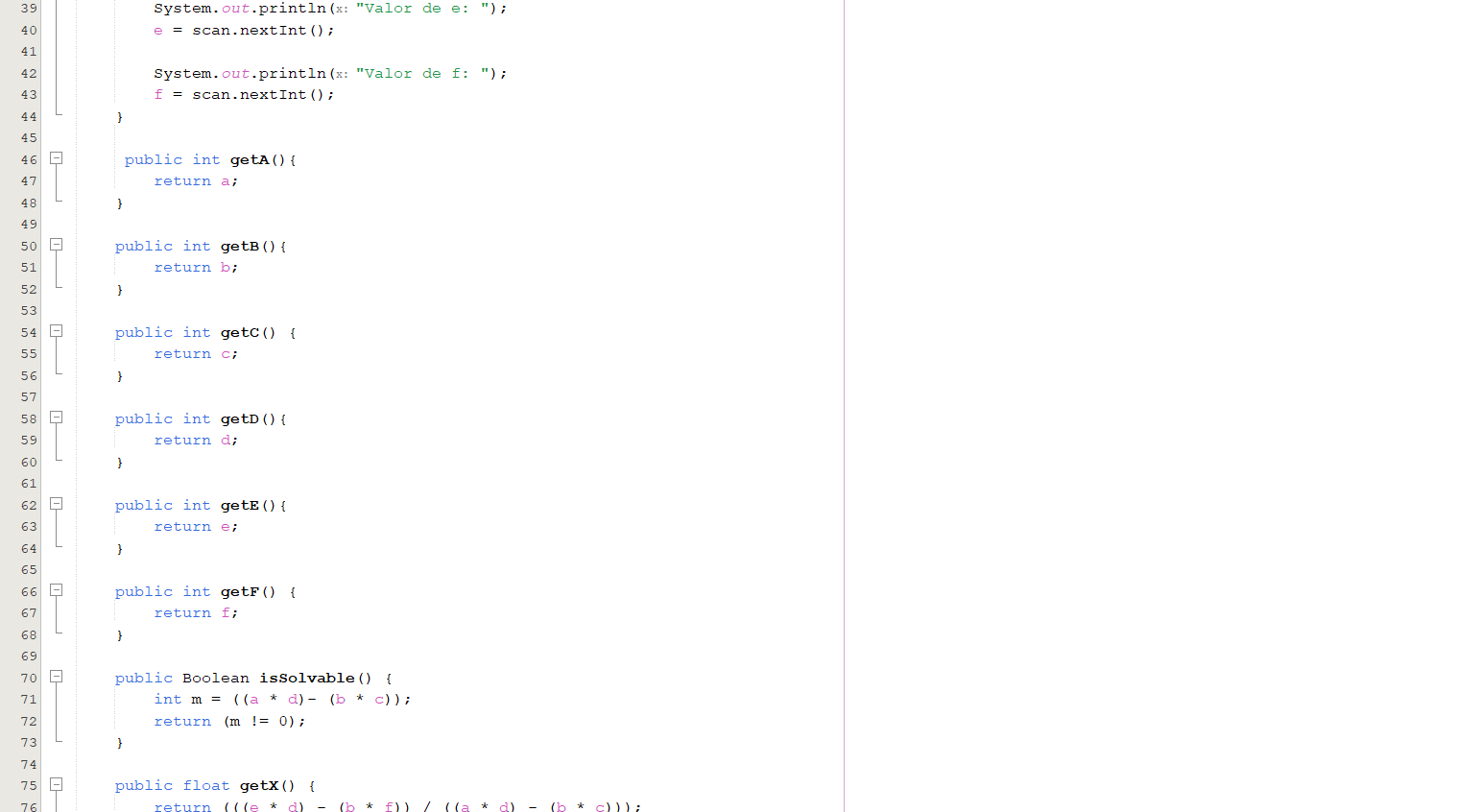
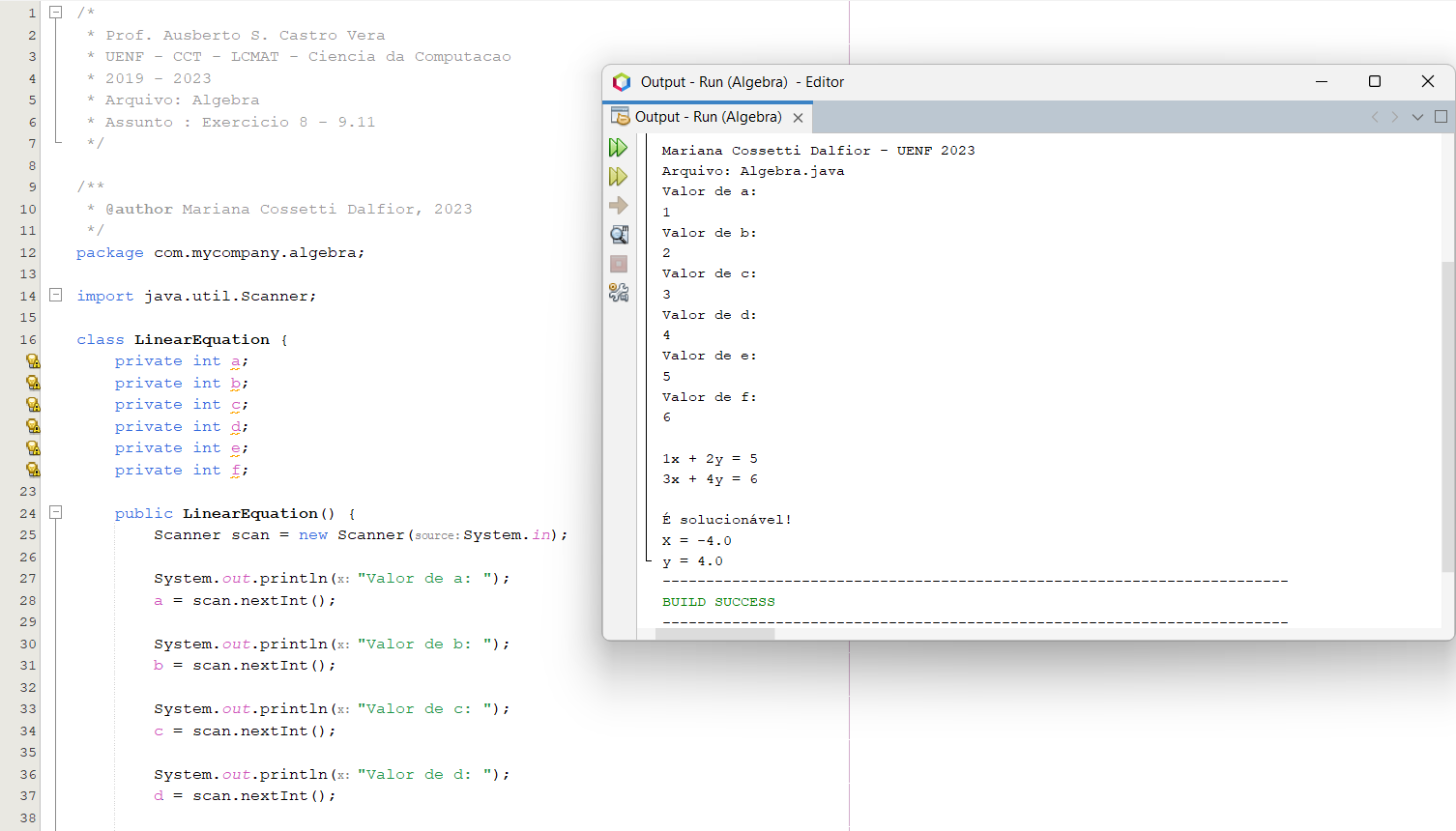
1. Rede:



1. **(02 pontos)** Resolver a Questão **9.10** e **9.11** (pág.363) do livro “Introduction to JAVA Programming”, Y. Daniel Liang, 2014. Não precisa desenhar os diagramas UML. “Implemente a classe” = “Faça um programa em Java”
2. 9.10:



1. 9.11:



**O professor**

