MODULO 6
Herança

Classe Pessoa

```
public class Pessoa {
	private String nome;
	private String sobrenome;
	private int idade;

	public Pessoa(String nome, String sobrenome, int idade) {
		this.nome = nome;
		this.sobrenome = sobrenome;
		this.idade = idade;
	}

	public String getNome() { return this.nome;	}

	public String getSobrenome() {  return this.sobrenome;	}

	public int getIdade() {	return this.idade;	}

	public void printDados(){
		System.out.println("---Dados da Pessoa:---"+
				"\nNome: " + this.nome +
				"\nSobrenome: " + this.sobrenome +
				"\nIdade: " + this.idade );

	}
}
```

Classe Aluno

```
public class Aluno extends Pessoa{
	private String ra;

	public Aluno(String nome, String sobrenome, int idade) {
		super(nome, sobrenome, idade);
	}

	public Aluno(String nome, String sobrenome, int idade, String ra) {
		super(nome, sobrenome, idade);
		this.ra=ra;
	}

	public void printDados() {
		super.printDados();
		System.out.println("RA: " + this.ra );
	}
	public void estudar(String disciplina){
		System.out.println("Sendo aluno, eu estudo a disciplina "+
disciplina);
	}
}
```

Classe Professor

```
public class Professor extends Pessoa {

	private int numAlunos; // quantidade de alunos

	public Professor(String nome, String sobrenome, int idade) {
		super(nome, sobrenome, idade);
	}
	public Professor(String nome, String sobrenome, int idade, int numAl) {
		super(nome, sobrenome, idade);
		this.numAlunos=numAl;

	}

	public void printDados() {
		super.printDados();
		System.out.println("NumAlunos: " + this.numAlunos );
	}

	public void ensinar(String disciplina){
		System.out.println("Sendo profe, eu ministro a disciplina "+
disciplina);
	}

	public void ensinar(){
		System.out.println("Sendo profe, eu ensino a "+ numAlunos + "
alunos");
	}
}
```

Classe TestAlunoEProfessor

```
public class TestAlunoEProfessor {
	public static void main(String[] args) {
		Aluno alunoRoberto= new Aluno("Roberto","Silva",17, "9300873C");
		alunoRoberto.printDados();  // usando metodo do pai Pessoa
		alunoRoberto.estudar("ALPOO"); // usando metodo proprio de Aluno

		Professor profSheila = new Professor("Sheila","Caceres",15,500);
		profSheila.printDados();  // usando metodo do pai Pessoa
		profSheila.ensinar("ALPOO");// usando metodo proprio de Professor
		profSheila.ensinar();

	}
}
```

Saída ao rodar a classe TestAlunoEProfessor:

```
---Dados da Pessoa:---
Nome: Roberto
Sobrenome: Silva
Idade: 17
RA: 9300873C
Sendo aluno, eu estudo a disciplina ALPOO
---Dados da Pessoa:---
Nome: Sheila
Sobrenome: Caceres
Idade: 15
NumAlunos: 500
Sendo profe, eu ministro a disciplina ALPOO
Sendo profe, eu ensino a 500 alunos
```

Lista de Exercícios 6

1. Implemente o cenário a seguir:
   a) Escreva uma classe chamada **Produto** que siga a especificação abaixo:

Atributos

- codigo (de tipo String): único para cada produto.
- descricao (de tipo String)
- quantidade (de tipo int)

Métodos

- Implemente os métodos get e set para todos os atributos.
- Implemente os métodos a seguir:

Método	**Parâmetros**	**Descrição**
construtor	codigo, descricao, quantidade	Inicializa os atributos da classe.
adicionar	int numero	Adiciona numero à quantidade do produto.
retirar	Int numero	Retira um certo número da quantidade disponível quantidade. Se houver menos produtos dos que os pedidos, retire todos os produtos. Deverá retornar a quantidade de produtos que foram retirados.

   b) Escreva uma classe chamada **ProdutoPerecivel** que herda de **Produto** e que siga a especificação abaixo:

Atributos

- dataValidade (podem usar o tipo Data da lista de exercícios 3, criar sua própria classe Data ou usar alguma classe já disponível em Java como por exemplo java.util.Date).

Métodos

- Implemente os métodos get para todos os atributos novos.
- Implemente o construtor que receba como parâmetros codigo, descricao, quantidade e data. Na sua implementação, deverá chamar o método construtor da classe base.

   c) Escreva uma classe chamada **ProdutoComPreco** que herda de **Produto** e que siga a especificação abaixo:

Atributos

- preco (de tipo double).

Métodos

- Implemente os métodos get e set para todos os atributos novos.
- Implemente o construtor que receba como parâmetros codigo, descricao, quantidade e preco. Na sua implementação, deverá chamar o método construtor da classe base.
- O método set que altera o preço unitário deverá verificar que o preço seja sempre positivo.

d) Escreva uma classe chamada **Estoque** que mantenha uma lista com os produtos em estoque. Todos os produtos serão do tipo ProdutoComPreco. A classe deverá seguir a especificação a seguir:

Atributos

- lstProdutos (de tipo ArrayList).

Métodos

- Implemente o construtor.
- Implemente os métodos descritos a seguir:

Método	**Parâmetros**	**Descrição**
cadastrarProduto	String codigo, String descricao, int quantidade, double preco	Permite o cadastramento de produtos novos inicializando os seus atributos
consultarProduto	String codigo	Imprimirá informação que descreve um produto
adicionarProduto	int quantidade, String codProduto	Permitirá adicionar produtos já cadastrados.
retirarProduto	int quantidade, String codProduto	Permitirá a retirada de produtos.
custoTotal	--	Devolverá um número informando o custo total do estoque armazenado

Testando:

- Escreva uma classe de teste chamada EstoqueTeste que demonstra as capacidades das classes criadas.
- Cadastre vários produtos, adicione e retire unidades dos produtos e depois obtenha o custo total do estoque

2. Estudo de caso

Os professores de uma universidade dividem-se em 2 categorias

- professores em dedicação exclusiva (DE): possuem um salário fixo para 40 horas de atividade semanais
- professores horistas: recebem um valor estipulado por hora

O cadastro de professores desta universidade armazena o nome, idade, matrícula e informação de salário.
Modele e implemente classes que permitam criar todos os professores da universidade. Deverá usar herança nessa modelagem para evitar código redundante.