PROGRAMAÇÃO II

Revisão

Observações

1. Os exercícios não práticos devem ser feitos em uma folha apartada contendo seu nome e data atual.
2. Os exercícios práticos devem ser desenvolvidos no Eclipse e enviado por e-mail.
3. A partir da questão 19 todas as classes devem ser criadas no pacote “model”.
4. A realização das atividades garante uma pontuação extra, se concluído.

Enviar para
mthscouto2@gmail.com

Centro Educacional Cozzolino

Confie em quem faz, confie na experiência!

PROGRAMAÇÃO II

REVISÃO – 2º BIMESTRE

Aluno(a):	**Nº**
Professor: Matheus Couto	
Ano de Escolaridade:	**Turma: 3º CTI** / **Data: 21/05/2025**

1. Explique o conceito de herança em Java. Qual palavra-chave é utilizada para implementá-la?

2. Por que a herança é útil em projetos orientados a objetos? Dê um exemplo de uso comum.

3. **(PRÁTICA)** Suponha a seguinte classe:

```
class Animal {
    void emitirSom() {
        System.out.println("Som genérico");
    }
}
```

Crie uma classe Cachorro que herda de Animal e sobrescreve o método emitirSom().

4. O que será impresso no console ao executar o seguinte código?

```
class Pessoa {
    void saudacao() {
        System.out.println("Olá");
    }
}

class Aluno extends Pessoa {
    void saudacao() {
        System.out.println("Olá, sou um aluno");
    }
}

public class Teste {
    public static void main(String[] args) {
        Pessoa p = new Aluno();
        p.saudacao();
    }
}
```

5. **(PRÁTICA)** Crie uma hierarquia de classes: Veiculo (classe base), Carro e Moto (subclasses). Adicione um método mover() em Veiculo e sobrescreva nas subclasses.

6. Dado o seguinte código:

```
class A {
    int x = 10;
}

class B extends A {
    int x = 20;
}
```

Se criarmos um objeto A obj = new B();, qual valor será impresso ao acessar obj.x?

7. Explique o conceito de polimorfismo em Java.

8. O que será impresso?

```
class Forma {
    void desenhar() {
        System.out.println("Desenhando forma");
    }
}

class Circulo extends Forma {
    void desenhar() {
        System.out.println("Desenhando círculo");
    }
}

public class Teste {
    public static void main(String[] args) {
        Forma f = new Circulo();
        f.desenhar();
    }
}
```

9. O que é encapsulamento em Java e qual a sua importância?

10. (PRÁTICA) Crie uma classe Produto com atributos nome e preco, ambos privados. Crie os métodos get e set para acessar e modificar esses atributos.

11. (PRÁTICA) Crie uma classe Carro com um atributo privado velocidade. Implemente métodos para acelerar e frear que modifiquem esse valor com regras.

12. (PRÁTICA) Crie um novo projeto Java e crie 3 pacotes (model, dao e main). (**OBS:** Para esta questão enviar o print da estrutura do projeto após a criação dos 3 pacotes).

13. Explique o que está errado neste código e corrija:

```
class Conta {
    double saldo;
}

Conta c = new Conta();
c.saldo = -1000;
```

14. Analise e explique o comportamento:

```
class Pessoa {
    private String nome;

    public void setNome(String nome) {
        if (nome.length() > 2)
            this.nome = nome;
    }

    public String getNome() {
        return nome;
    }
}
```

15. Por que o código abaixo compila, mas gera comportamento indesejado?

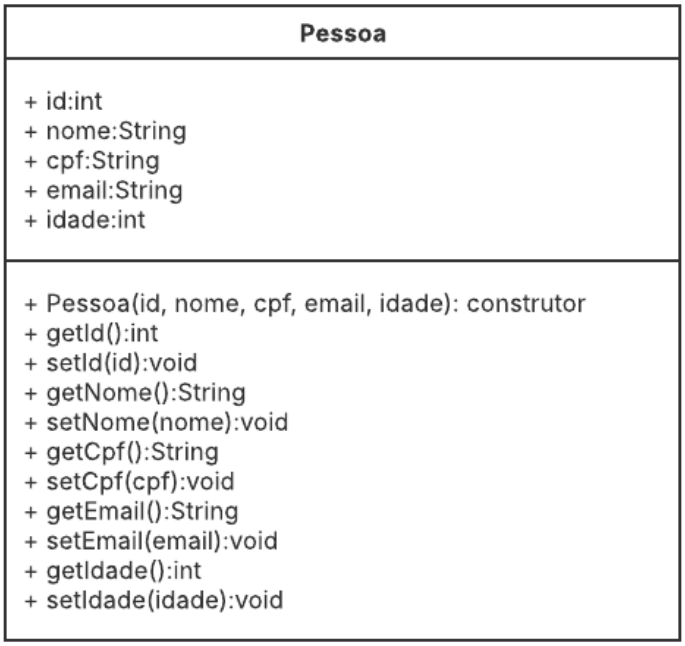
```
class Aluno {
    String nome;
}

Aluno a = new Aluno();
a.nome = null;
```

16. Na sua opinião, quais são as vantagens de se utilizar pacotes em Java?

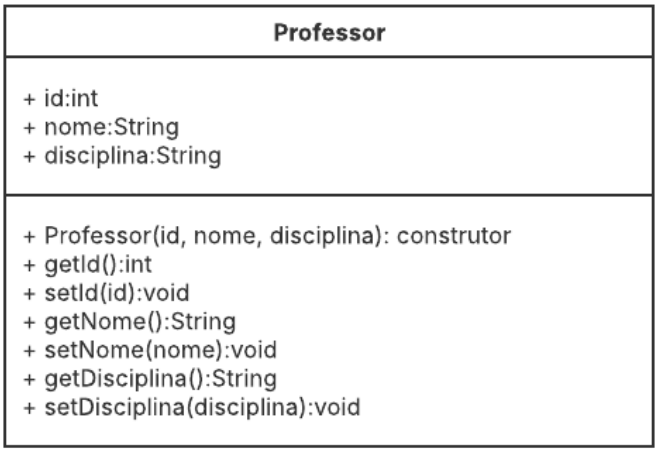
17. Qual a diferença entre import pacote.* e import pacote.Classe?

18. Marque V para Verdadeiro e F para Falso:
( ) A herança permite que uma classe reutilize atributos e métodos de outra classe.
( ) O encapsulamento consiste em tornar todos os atributos e métodos de uma classe públicos.
( ) O polimorfismo permite que diferentes classes respondam de maneira diferente a uma mesma mensagem (método).
( ) Os pacotes ajudam a organizar melhor as classes e evitar conflitos de nomes em projetos grandes

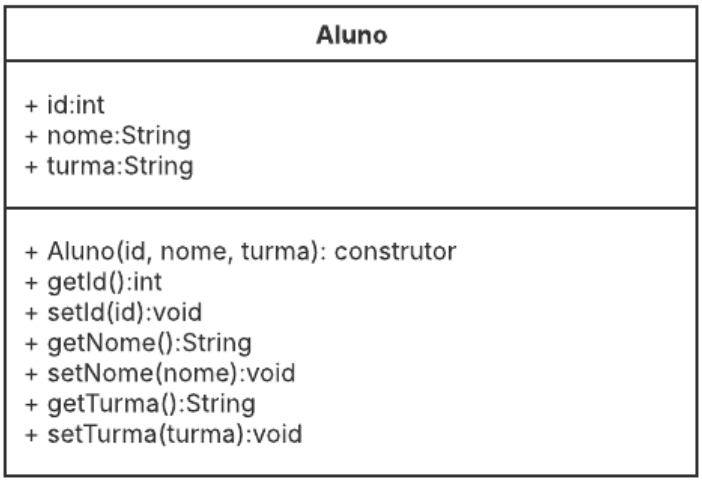
19. **(PRÁTICA)** Crie a classe abaixo com seus atributos e métodos:



20. **(PRÁTICA)** Crie a classe abaixo com seus atributos e métodos:



21. **(PRÁTICA)** Crie a classe abaixo com seus atributos e métodos:



22. (PRÁTICA) Crie a classe abaixo com seus atributos e métodos:

Veiculo
+ id:int + marca:String + modelo:String + tipo:String
+ Veiculo(id, marca, modelo, tipo): construtor + getId():int + setId(id):void + getMarca():String + setMarca(marca):void + getModelo():String + setModelo(modelo):void + getTipo():String + setTipo(tipo):void

23. (PRÁTICA) Crie a classe abaixo com seus atributos e métodos:

Animal
+ id:int + nome:String + raca:String
+ Animal(id, nome, raca): construtor + getId():int + setId(id):void + getNome():String + setNome(nome):void + getRaca():String + setRaca(raca):void

24. (PRÁTICA) Crie a classe abaixo com seus atributos e métodos:

Perfil
+ id:int + nome:String + codigo:int
+ Perfil(id, nome, codigo): construtor + getId():int + setId(id):void + getNome():String + setNome(nome):void + getCodigo():int + setCodigo(codigo):void

24. (PRÁTICA) Crie a classe abaixo com seus atributos e métodos:

Empresa
+ id:int + cnpj:String + razaoSocial:String + nomeFantasia:String
+ Empresa(id, cnpj, razaoSocial, nomeFantasia): construtor + getId():int + setId(id):void + getCnpj():String + setCnpj(cnpj):void + getRazaoSocial():String + setRazaoSocial(razaoSocial):void + getNomeFantasia():String + setNomeFantasia(nomeFantasia):void

25. (PRÁTICA) Crie a classe abaixo com seus atributos e métodos:

Funcionario
+ id:int + nome:String + cargo:String
+ Funcionario(id,nome, cargo): construtor + getId():int + setId(id):void + getNome():String + setNome(nome):void + getCargo():String + setCargo(cargo):void