INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PIAUÍ	**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ** **Curso: ADS** **Disciplina: Programação Orientada a Objetos** **Professor: Ely**

Exercício 06

1. Dadas as três classes abaixo:

```
public class Empregado {
    private double salario = 500;
    public double calcularSalario() { ...}
}
```

```
public class Diarista extends Empregado {
    public double calcularSalario() { ...}
}
```

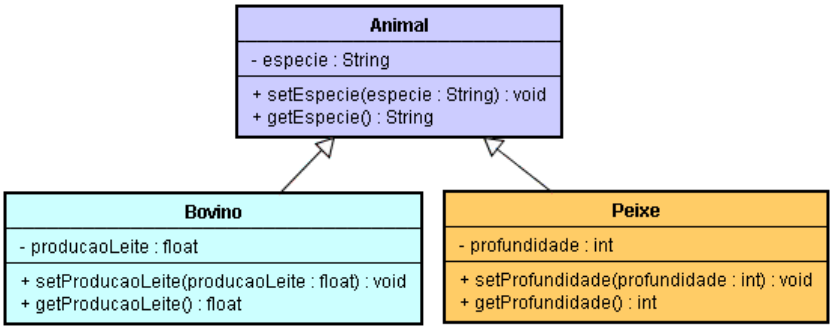
```
public class Horista extends Diarista {
    public double calcularSalario() { ... }
}
```

   Implemente os métodos calcularSalario() de cada classe da seguinte forma:

   a) Empregado: apenas retorna o valor do atributo salário;
   b) Diarista: sobrescreve calcularSalario(), chamando o método homônimo de Empregado e dividindo o resultado por 30;
   c) Horista: sobrescreve calcularSalario(), chamando o método homônimo de Diarista e dividindo o resultado por 24.

2. Crie uma classe chamada CalculadoraCientifica que herda da classe Calculadora do exercício passado.
   a. Implemente um método **exponenciar** que tenha como parâmetros uma base e um expoente e retorne um double;
   b. Caso não exista, crie um método de divisão na classe Calculadora. Crie uma nova versão do método de divisão na classe CalculadoraCientifica que tenha um parâmetro lógico que represente se a operação deve ou não retornar um valor arredondado. Dica: chame com o super a definição anterior para reaproveitar o código

3. Implemente o diagrama abaixo com classes Java e crie uma classe para testar as 3 classes:



4. Utilizando as classes do exercício anterior, responda:
   a. Dado o código abaixo, diga qual o problema e qual a solução para que se possa definir a produção de leite.

```
Animal a = new Bovino();
a.setProducaoLeite(5);
```

b. O código abaixo é possível? Justifique.
```
Peixe p = new Animal();
```

5. Adicione um atributo privado nome e métodos de acesso na classe animal e:
   a. Em um método main() de uma classe de testes crie um array de classes do tipo Animal;
   b. Instancie e adicione ao array 2 objetos da classe Animal, 2 da classe Peixe e 3 da classe Bovino
   c. Por que o array da classe Animal aceita também os descendentes na hierarquia?
   d. Execute um for fazendo 3 ifs (sem else) que utilizam o *instanceof* para verificar a que classe pertence cada objeto do array. Imprima uma String o tipo do animal (“Animal”, “Bovino” ou “Peixe”)

6. Crie uma classe Pessoa com:
   a. Atributos privados nome (tipo String) e sobrenome (tipo String). Cada um desses atributos deve ter métodos para lê-los e alterá-los (getters e setters).
   b. Um método chamado getNomeCompleto que não possui parâmetros de entrada e que retorna a concatenação do atributo nome com o atributo sobrenome.
   c. Um construtor sem parâmetros e um outro construtor que recebe como parâmetros o nome e o sobrenome da pessoa e altera respectivamente os atributos nome e sobrenome.

7. Crie uma subclasse de Pessoa, chamada Funcionario que deve possuir:
   a. Os atributos matricula (tipo int) e salario (tipo double), com seus respectivos métodos para leitura e alteração (getters e setters).
   b. O salário de um funcionário jamais poderá ser negativo. Todo funcionário recebe seu salário em duas parcelas, sendo 60% na primeira parcela e 40% na segunda parcela. Assim, escreva os métodos getSalarioPrimeiraParcela que retornam o valor da primeira parcela do salário (60%) e getSalarioSegundaParcela que retorna o valor da segunda parcela do salário (40%).

8. Uma subclasse de Funcionario, chamada Professor tendo:
   a. Um atributo titulacao (String) com seus métodos get e set
   b. Todo professor recebe seu salário em uma única parcela. Assim, deve-se sobrescrever os métodos getSalarioPrimeiraParcela e getSalarioSegundaParcela. O método getSalarioPrimeiraParcela da classe Professor deve retornar o valor integral do salário do professor e o método getSalarioSegundaParcela do professor deve retornar o valor zero.

9. Crie uma classe que teste todos os métodos das classes das questões antereriores.

10. Altere a implementação da classe Banco para:
    a. manipular contas Poupanca;
    b. manipular ContaImposto:
       - crie os métodos na interface e na classe banco necessários para cadastro e operações específicas para o novo tipo de conta

- Teste sempre com *instanceof* que a operação for específica se a classe é realmente do tipo necessário

Nota: as classes Poupanca e ContaImposto são as mesmas apresentadas nos slides

Considere as classes A e B abaixo para as 4 próximas questões:

```
public class A {
    private int atributo1;

    public String metodo1() {
        return "metodo 1, classe A";
    }
}
```

```
public class B extends A {
    public String metodo1() {
        return "metodo 1, classe B";
    }

    public String metodo2() {
        return "metodo 2, classe B";
    }
}
```

11. Marque V ou F:
    ( ) em uma instancia b da classe B, se fizermos b instanceof A retorna true
    ( ) em uma instancia a da classe A, se fizermos a instanceof B retorna false
    ( ) B possui o atributo1
    ( ) Em B o atributo não é visível
    ( ) Caso o atributo fosse modificado de private para protected, poderia ser acessado normalmente em B

12. Qual a exibição do código abaixo:

```
A a = new B();
System.out.println(a.metodo1());
```

13. Diga o que ocorre com o código abaixo e justifique:

```
A a = new B();
System.out.println(a.metodo2());
```

14. Caso a classe B fosse reescrita conforme abaixo:

```
public class B extends A {
    public String metodo1() {
        return super.metodo1();
    }
}
```

Qual seria a nova saída do código da questão 2?

15. Implemente métodos de leitura e escrita para o atributo1 da classe A.

16. Dada a classe abaixo:

```
public class B {
    public String metodo1() {
        return "metodo 1 da classe B";
    }

    public String metodo1() {
        return "metodo 1 da classe B, segunda versão";
    }
```

```
    ...
}
```

Justifique por que o código acima não é compilável.

17. Dada a classe abaixo:

```
public class Calculadora {
    public int soma(int op1, int op2) {
        return op1 + op2;
    }

    public double soma(int op1, int op2) {
        return op1 + op2;
    }
}
```

Qual o problema da definição acima?