**Instituto Federal de Educação, Ciência e Tecnologia do Ceará**

**Campus Maracanaú**

**Coordenadoria de Computação**

**Curso de Bacharelado em Ciência da Computação**

**Disciplina: Programação Orientada a Objetos**

**Professor: Igor Rafael Silva Valente**

# ATIVIDADE

**Assunto:**

Interfaces.

**Orientações:**

A atividade deve ser executada individualmente e entregue através do ambiente *Google Classroom*.

**Regras de criação dos programas:**

Crie um novo projeto Java denominado **AtividadeInterfaces**. As classes devem possuir os nomes informados no texto. Ao final, o projeto deve ser exportado para um arquivo em formato ZIP.

**Nome completo: Francisco Aldenor Silva neto**

1. Quais as diferenças entre classes abstratas e interfaces? Explique.

Uma classe abstrata é uma classe que pode conter métodos abstratos(prototicos) é não pode ser instanciada porém ela pode possuir também métodos concretos em sua implementação

Uma interface possui apenas métodos abstratos(prototicos) não podendo implementar métodos concretos com exceção do deful método ela também pode possuir atributos desde que seja statico

1. Interfaces podem ter métodos concretos? Explique.

Não! Existe uma possibilidade de implementação de métodos dentro de uma interface que é denominada *default methods* para evitar uma repetição de código pois esse método servirá como um método padrão para as classes que implementarem essa interface não sendo necessário assim a implementação dele na própria classe evitando uma repetição de código.

1. Demonstre como o uso de *default methods* pode evitar a repetição de código.
2. Em uma situação em que classes abstratas e interfaces são opções viáveis, qual deve ser utilizada prioritariamente?

Quando as duas opções são viáveis deve-se usar a interface pois o Java permite termos múltiplas implementação de interface porém só nos permite apenas uma herança por classe portanto ao usarmos a implementação da interface nós não queimamos a única oportunidade de usarmos a herança que temos.

1. A tabela a seguir descreve um conjunto de classes abstratas e concretas que participam de uma hierarquia de herança. Analise as características de cada classe e faça o que se pede.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nome da classe** | **Tipo da classe** | **Herda de** | **Lista de atributos** | **Lista de métodos** |
| Transporte | Abstrata | – | - nome: String  - numeroPassageiros: int  - interfacevelocidadeAtual: int | + estaParado(): boolean |
| TransporteAereo | Abstrata | Transporte | - altitudeAtual: int | + subir(metros: int): void  + descer(metros: int): void |
| TransporteTerrestre | Abstrata | Transporte | - tipo: String | + estacionar(): void |
| Aviao | Concreta | TransporteAereo | - numeroMotores: int | + estaparado(): boolean  + subir(metros: int): void  + descer(metros: int): void  + curvar(angulo: float): void  + abastecer(numLitros: int): void  + ligarMotor(): void |
| Balao | Concreta | TransporteAereo | + pesoLargada: int | + estaparado(): boolean  + subir(metros: int): void  + descer(metros: int): void  + largarPeso(peso: float): void  + aquecerAr(temp: float): void |
| Carro | Concreta | TransporteTerrestre | - numeroCilindrada: int | + estaParado(): boolean  + estacionar(): void  + curvar(angulo: float): void  + abastecer(numLitros: int): void  + ligarMotor(): void  + embraiar(): void |
| Bicicleta | Concreta | TransporteTerrestre | - numeroRaios: int | + estaParado(): boolean  + estacionar(): void  + curvar(angulo: float): void  + pedalar(): void |

**O que se pede:**

* Crie uma interface de nome Motorizado em que são declarados os métodos void ligarMotor() e void abastecer(int numLitros).
* Implemente a interface Motorizado nas classes Aviao e Carro.
* Escreva um programa de teste capaz de verificar a implementação anterior.
* Crie uma interface de nome Conduzivel onde é declarado o método void curvar(float angulo).
* Implemente a interface Conduzivel nas classes Aviao, Carro e Bicicleta.
* Complete o programa de teste criado anteriormente por forma a testar estas últimas implementações.

Boa sorte!

Prof. Igor.