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:
Reuso de classes.

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

Nome completo:

Robert Silva Queiroz

1. Existem duas formas básicas para realizar o reuso de classes: composição e herança. Explique o significado e cite um exemplo para cada uma.

   Composição: Composição é um princípio de design onde uma classe é composta por objetos de outras classes, o que significa que a classe contém instâncias de outras classes como membros. Essa abordagem permite criar estruturas complexas combinando diferentes classes de forma flexível.

   Exemplo: Um exemplo de composição pode ser um carro que possui um motor, rodas, transmissão, etc. A classe Carro teria instâncias dessas outras classes como seus membros.

   Herança: Herança é um conceito de orientação a objetos onde uma classe pode herdar atributos e métodos de outra classe. Isso permite que uma classe filha reutilize o comportamento de uma classe pai, adicionando ou modificando funcionalidades conforme necessário.

   Exemplo: Um exemplo de herança pode ser uma hierarquia de classes Animal, onde temos a classe pai Animal e classes filhas como Cachorro, Gato, Pássaro, etc. As classes filhas herdam características da classe pai, como mover(), comer(), dormir(), etc., e também podem ter métodos e atributos próprios, como latir() para um Cachorro.

2. Explique a diferença entre sobrecarga e sobrescrita de métodos. Crie um código-fonte na linguagem Java demonstrando a diferença entre os dois conceitos.

   Sobrecarga de Métodos:

   Significado: Sobrecarga de métodos ocorre quando uma classe tem dois ou mais métodos com o mesmo nome, mas com diferentes parâmetros. Isso permite que os métodos tenham comportamentos diferentes dependendo dos argumentos que são passados para eles.

   Sobrescrita de Métodos:

Significado: Sobrescrita de métodos ocorre quando uma classe filha implementa um método que já está presente na classe pai. Isso permite que a classe filha forneça sua própria implementação do método, substituindo a implementação da classe pai.

3. Analise o código-fonte do programa Java a seguir e informe quais mensagens serão impressas de acordo com a sequência de execução do programa. **OBS: não execute o código-fonte antes de ter a sua resposta, aproveite para treinar o entendimento dos conceitos.**

```
public class ClientePF extends PessoaFisica {

	private int codCliente;

	public ClientePF() {
		this(999);
		System.out.println("ClientePF()");
	}

	public ClientePF(int codCliente) {
		super();
		System.out.println("ClientePF("+codCliente+")");
		this.codCliente = codCliente;
		super.show();
	}

	public void show() {
		System.out.println("Show ClientePF");
	}

	public static void main(String[] args) {
		new ClientePF();
	}

}

class PessoaFisica extends Pessoa {

	private int cpf;

	public PessoaFisica() {
		super();
		System.out.println("PessoaFisica() com CPF "+cpf);
		show();
	}

	public void show() {
		System.out.println("Show PessoaFisica");
	}

}

class Pessoa {

	private String nome;

	public Pessoa() {
		this.nome = "João";
		System.out.println("Pessoa()");
	}

}
```

Saída:

```
Pessoa()
PessoaFisica() com CPF 0
Show ClientePF
ClientePF(999)
Show PessoaFisica
ClientePF()
```

4. Identifique e explique o(s) erro(s) da classe a seguir. **OBS: não execute o código-fonte antes de ter a sua resposta, aproveite para treinar o entendimento dos conceitos.**

```
public class DemoConstrutor {

        private int a, b;

        public DemoConstrutor() {
                System.out.println("Sem argumentos...");
                DemoConstrutor(0,0);
        }

        public DemoConstrutor(int xa, int xb) {
                System.out.println("Com argumentos...");
                a = xa; b = xb;
        }
}
```

Na linha DemoConstrutor(0,0); : a intenção parece ser chamar o construtor que recebe dois argumentos, mas isso está incorreto. No Java, para que possamos chamar um construtor, devemos usar a palavra-chave this seguida pelos parâmetros do construtor. Portanto, a linha deve ser alterada para this(0, 0);.

Ao chamar o construtor com this(0, 0);, a chamada para o construtor padrão (public DemoConstrutor()) deve ser a primeira linha no construtor que não chama o construtor de superclasse. Portanto, o construtor padrão também precisa ser corrigido para chamar o construtor que recebe dois argumentos.

5. Escreva a classe `ObjetoGeometrico` que representa um objeto geométrico em duas dimensões. Essa classe deve ter um construtor para inicializar o objeto e métodos para mostrar seus dados, calcular e retornar sua área e perímetro. Usando essa classe como base, escreva as classes herdeiras `Circulo` (contendo duas coordenadas para o centro e um raio), `Retangulo` (contendo dois valores para os lados) e `Triangulo` (contendo três valores para os lados), que sobrescrevem os métodos em `ObjetoGeometrico`. Dicas: A área de um círculo pode ser calculada com `Math.PI*r*r`, em que `r` é o raio do círculo. O perímetro de um círculo é dado por `2*Math.PI*r`. A área do retângulo é dada por `b*h`, onde b é um dos lados e h é o outro lado. Seu perímetro é dado por `2*b+2*h`. A área de um triângulo é dada por `Math.sqrt(s*(s-a)*(s-b)*(s-c))`, onde `Math.sqrt` é a função que calcula a raiz quadrada, `a, b` e `c` são os lados do triângulo, e `s` é a metade do perímetro do triângulo. O perímetro do triângulo é calculado como (`a+b+c`).

Boa sorte!

Prof. Igor.