Questões e Respostas

1. O que é a JVM e qual seu papel na execução de um programa Java?
2. Explique a diferença entre JDK, JRE e JVM.
3. O que é um construtor em Java? Como ele funciona?

public class VeiculoDAO {

public VeiculoDAO() {

System.out.println("Objeto VeiculoDAO criado!");

}

public boolean excluirVeiculo(int id) {

System.out.println("Excluindo veículo com ID: " + id);

return true;

}

}

public class Main {

public static void main(String[] args) {

int id = 10;

VeiculoDAO veiculoDAO = new VeiculoDAO();

if (veiculoDAO.excluirVeiculo(id)) {

System.out.println("Veículo excluído com sucesso!");

} else {

System.out.println("Erro ao excluir veículo.");

}

}

}

1. Qual é a diferença entre os primitivos e objetos em Java?

public class Veiculo {

int id;

String placa;

String cpfDono;

public Veiculo(int id, String placa, String cpfDono) {

this.id = id;

this.placa = placa;

this.cpfDono = cpfDono;

}

public void exibirInfo() {

System.out.println("ID: " + id);

System.out.println("Placa: " + placa);

System.out.println("CPF do dono: " + cpfDono);

}

}

OBJETOS EM JAVA

record Pessoa(String nome, int idade) {}

public class ExemploObjetos {

public static void main(String[] args) {

Pessoa p1 = new Pessoa("Carlos", 30);

Pessoa p2 = new Pessoa("Ana", 25);

System.out.println(p1);

System.out.println(p2);

}

}

1. O que é sobrecarga de métodos em Java?

class ClassVeiculoDAO {

public function atualizarVeiculo(ClassVeiculo $veiculo) {

echo "Atualizando veículo com ID: " . $veiculo->getId() . "\n";

}

public function atualizarVeiculo($id, $placa, $cpf\_dono) {

echo "Atualizando veículo com ID: $id, Placa: $placa, CPF do Dono: $cpf\_dono\n";

require\_once 'ClassVeiculoDAO.php';

require\_once 'ClassVeiculo.php';

if (isset($\_GET['id'])) {

$id = $\_GET['id'];

$veiculoDAO = new ClassVeiculoDAO();

$veiculo = $veiculoDAO->buscarVeiculoPorId($id);

}

if ($\_SERVER['REQUEST\_METHOD'] == 'POST') {

$veiculoAtualizado = new ClassVeiculo();

$veiculoAtualizado->setId($\_POST['id']);

$veiculoAtualizado->setPlaca($\_POST['placa']);

$veiculoAtualizado->setCpfDono($\_POST['cpf\_dono']);

$veiculoDAO->atualizarVeiculo($veiculoAtualizado);  
  
header('Location:listar.php');  
exit;  
 }

1. Qual a diferença entre ArrayList e LinkedList?

import java.util.ArrayList;

public class VeiculoDAO {

public ArrayList<Veiculo> listarVeiculos() {

ArrayList<Veiculo> listaVeiculos = new ArrayList<>();

listaVeiculos.add(new Veiculo(1, "ABC1234", "12345678900"));

listaVeiculos.add(new Veiculo(2, "XYZ5678", "98765432100"));

return listaVeiculos;

}

}

import java.util.LinkedList;

public class VeiculoDAO {

public LinkedList<Veiculo> listarVeiculos() {

LinkedList<Veiculo> listaVeiculos = new LinkedList<>();

listaVeiculos.add(new Veiculo(1, "ABC1234", "12345678900"));

listaVeiculos.add(new Veiculo(2, "XYZ5678", "98765432100"));

return listaVeiculos;

}

}

1. O que é herança em Java e como funciona?

<?php

class Passagem {

public $rota, $valorp, $integracao, $qtd;

function comprar($valorp, $qtd) {

$total = $valorp \* $qtd;

return $total;

}

function mostrar($rota, $valorp, $integracao, $qtd, $total) {

echo "ROTA: $rota <br>";

echo "VALOR: $valorp <br>";

echo "INTEGRAÇÃO: $integracao <br>";

echo "QTD: $qtd <br>";

echo "TOTAL: $total <br>";

}

}

class PassagemComercial extends Passagem {

public $desconto;

function aplicarDesconto($total) {

$this->desconto = $total \* 0.1;

return $total - $this->desconto;

}

}

class PassagemInternacional extends PassagemComercial {

public $taxaImpostos;

function aplicarImposto($total) {

$this->taxaImpostos = $total \* 0.2;

return $total + $this->taxaImpostos;

}

}

$passagem = new PassagemInternacional();

$rota = "São Paulo - Rio de Janeiro";

$valorp = 200; $integracao = "Sim";

$qtd = 3;

$total = $passagem->comprar($valorp, $qtd);

$totalComDesconto = $passagem->aplicarDesconto($total);

$totalComImposto = $passagem->aplicarImposto($totalComDesconto);

$passagem->mostrar($rota, $valorp, $integracao, $qtd, $totalComImposto);

?>

1. Explique a diferença entre == e .equals() em Java.

<?php

class Conta

{

public $nome, $conta, $saldo, $saque;

function sacar()

{

return $this->saldo - $this->saque;

}

function comparar($outraConta)

{

return $this->nome == $outraConta->nome && $this->conta == $outraConta->conta;

}

}

$cliente1 = new Conta();

$cliente1->nome = "Carlos";

$cliente1->conta = "12345";

$cliente1->saldo = 1000;

$cliente1->saque = 200;

$cliente2 = new Conta();

$cliente2->nome = "Carlos";

$cliente2->conta = "12345";

$cliente2->saldo = 2000;

$cliente2->saque = 500;

echo $cliente1->comparar($cliente2) ? "As contas são iguais.<br>" : "As contas são diferentes.<br>";

echo "TOTAL: " . $cliente1->sacar() . "<br>";

?>

1. O que é o tratamento de exceções e como ele é feito em Java?

<?php

require\_once "conexao.php";

require\_once "Produto.php";

require\_once "banco-produto.php";

try {

$produto = new Produto();

$produto->nome = $\_GET["nome"];

$produto->modelo = $\_GET["modelo"];

$produto->ano = $\_GET["ano"];

if (!$conexao) {

throw new Exception("Erro de conexão com o banco de dados.");

}

if (insereProduto($conexao, $produto)) {

echo "Produto cadastrado com sucesso!<br>";

echo "<a href='listar.php'>LISTAR</a><br>";

echo "<a href='editar.php'>EDITAR</a>";

} else {

throw new Exception("Erro ao inserir produto no banco.");

}

} catch (Exception $e) {

echo "Erro: " . $e->getMessage();

}

?>

1. O que são classes abstratas e interfaces em Java? Qual a diferença entre elas?

Classe Abstrata

abstract class Veiculo {

String modelo;

int ano;

Veiculo(String modelo, int ano) {

this.modelo = modelo;

this.ano = ano;

}

abstract void acelerar();

void exibirDetalhes() {

System.out.println("Modelo: " + modelo);

System.out.println("Ano: " + ano);

}

}

class Carro extends Veiculo {

int portas;

Carro(String modelo, int ano, int portas) {

super(modelo, ano);

this.portas = portas;

}

void acelerar() {

System.out.println("Carro " + modelo + " acelerando...");

}

}

public class TesteAbstrata {

public static void main(String[] args) {

Carro meuCarro = new Carro("Fusca", 1975, 4);

meuCarro.exibirDetalhes();

meuCarro.acelerar();

}

}

Exemplo Interface

interface Veiculo {

void acelerar();

void frear();

}

class Carro implements Veiculo {

String modelo;

Carro(String modelo) {

this.modelo = modelo;

}

public void acelerar() {

System.out.println(modelo + " acelerando...");

}

public void frear() {

System.out.println(modelo + " frenando...");

}

}

public class TesteInterface {

public static void main(String[] args) {

Carro meuCarro = new Carro("Fusca");

meuCarro.acelerar();

meuCarro.frear();

}

}

1. O que é encapsulamento e como ele é implementado em Java?

package controle;

public class Aluno {

private int codigo;

private String nome;

private String email;

public Aluno() {}

public Aluno(int codigo, String nome, String email) {

this.codigo = codigo;

this.nome = nome;

this.email = email;

}

public int getCodigo() {

return codigo;

}

public void setCodigo(int codigo) {

this.codigo = codigo;

}

public String getNome() {

return nome;

}

public void setNome(String nome) {

if (nome != null && !nome.trim().isEmpty()) {

this.nome = nome;

} else {

System.out.println("Erro: Nome inválido!");

}

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

if (email.contains("@")) {

this.email = email;

} else {

System.out.println("Erro: Email inválido!");

}

}

}

1. Qual é a diferença entre sobrecarga e sobrescrita de métodos?

Sobrecarga

public class Calculadora {

public int somar(int a, int b) {

return a + b;

}

public int somar(int a, int b, int c) {

return a + b + c;

}

public double somar(double a, double b) {

return a + b;

}

}

Sobrescrista

package entidade;

import controle.Vendas;

import java.sql.PreparedStatement;

public class ManterVendasEspecial extends ManterVendas {

@Override

public void inserir(Vendas v) throws Exception {

if (v.getValor\_venda() <= 0) {

System.out.println("Erro: O valor da venda deve ser maior que zero!");

return;

}

super.inserir(v);

System.out.println("Venda registrada com sucesso!");

}

}

public class Teste {

public static void main(String[] args) {

try {

Vendas v1 = new Vendas("João", 1500.00);

Vendas v2 = new Vendas("Maria", -500.00);

ManterVendasEspecial vendasEspecial = new ManterVendasEspecial();

vendasEspecial.inserir(v1);

vendasEspecial.inserir(v2);

} catch (Exception e) {

System.out.println("Erro: " + e.getMessage());

}

}

}

1. O que é polimorfismo e como ele se manifesta em Java?

class Calculadora {

public int somar(int a, int b) {

return a + b;

}

public int somar(int a, int b, int c) {

return a + b + c;

}

public double somar(double a, double b) {

return a + b;

}

}

public class TesteSobrecarga {

public static void main(String[] args) {

Calculadora calc = new Calculadora();

System.out.println(calc.somar(2, 3));

System.out.println(calc.somar(2, 3, 4));

System.out.println(calc.somar(2.5, 3.5)); }

}

1. O que é uma interface em Java e como ela difere de uma classe abstrata?

interface ProdutoDAO {

void inserir(Produto p) throws Exception;

void excluir(int id) throws Exception;

}

class Produto {

private String nome;

private String modelo;

private int ano;

public Produto(String nome, String modelo, int ano) {

this.nome = nome;

this.modelo = modelo;

this.ano = ano;

}

public String getNome() { return nome; }

public String getModelo() { return modelo; }

public int getAno() { return ano; }

}

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

class ProdutoDAOImpl implements ProdutoDAO {

private Connection conectar() throws Exception {

return DriverManager.getConnection("jdbc:mysql://localhost/bdloja", "root", "");

}

@Override

public void inserir(Produto p) throws Exception {

String sql = "INSERT INTO produtos (nome, modelo, ano) VALUES (?, ?, ?)";

try (Connection conn = conectar(); PreparedStatement pst = conn.prepareStatement(sql)) {

pst.setString(1, p.getNome());

pst.setString(2, p.getModelo());

pst.setInt(3, p.getAno());

pst.executeUpdate();

System.out.println("Produto inserido com sucesso!");

}

}

@Override

public void excluir(int id) throws Exception {

String sql = "DELETE FROM produtos WHERE id = ?";

try (Connection conn = conectar(); PreparedStatement pst = conn.prepareStatement(sql)) {

pst.setInt(1, id);

pst.executeUpdate();

System.out.println("Produto excluído com sucesso!");

}

}

}

Classe Abstrata

import java.sql.Connection;

import java.sql.DriverManager;

abstract class AbstractProdutoDAO {

protected Connection conectar() throws Exception {

return DriverManager.getConnection("jdbc:mysql://localhost/bdloja", "root", "");

}

}

1. O que é associação, composição e agregação em POO?

Associação

class Pessoa {

String nome;

public Pessoa(String nome) {

this.nome = nome;

}

}

class Carro {

String modelo;

public Carro(String modelo) {

this.modelo = modelo;

}

}

public class TesteAssociacao {

public static void main(String[] args) {

Pessoa pessoa = new Pessoa("João");

Carro carro = new Carro("Fusca");

System.out.println(pessoa.nome + " tem um carro modelo " + carro.modelo);

}

}

Composição

class Quarto {  
 String tipo;  
 public Quarto(String tipo) {  
 this.tipo = tipo;  
 }  
}  
class Casa {  
 Quarto quarto;  
  
 public Casa(String tipoQuarto) {  
 quarto = new Quarto(tipoQuarto);

}  
}  
public class TesteComposicao {  
 public static void main(String[] args) {  
 Casa casa = new Casa("Quarto de casal");  
 System.out.println("A casa tem um " + casa.quarto.tipo);  
 }  
}

1. O que acontece se uma classe implementar duas interfaces que possuem métodos com a mesma assinatura?

<?php require\_once "conexao.php"; ?>  
<?php  
$nome = $\_POST["nome"];  
$telefone = $\_POST["telefone"];  
$endereco = $\_POST["endereco"]; /  
function inserir($conexao, $nome, $telefone, $endereco) {  
 $sql = "INSERT INTO contatos(nome, telefone, endereco) VALUES ('$nome', '$telefone', '$endereco')";  
 $resultado = mysqli\_query($conexao, $sql);  
  
 return $resultado;  
}  
  
if (inserir($conexao, $nome, $telefone, $endereco)) {  
 echo "<h3>FORMULÁRIO ENVIADO!<h3>";  
 echo "<a href='fordeletar.php'>Deseja deletar</a>";  
} else {  
 echo "ERRO";  
 echo mysqli\_error($conexao);  
}  
?>

1. O que é a palavra-chave super em Java e quando ela deve ser usada?

class SuperClasse {  
 int variavel = 10;  
}  
class SubClasse extends SuperClasse {  
 int variavel = 20;  
 public void mostrarVariaveis() {  
 System.out.println("Variável da SuperClasse: " + super.variavel);   
 System.out.println("Variável da SubClasse: " + this.variavel); }  
}  
  
public class TesteSuper {  
 public static void main(String[] args) {  
 SubClasse obj = new SubClasse();  
 obj.mostrarVariaveis();  
 }  
}

1. Como funciona o conceito de classes aninhadas (inner classes) em Java?

**Classe interna não Estática**

class FormularioDeletar {

class DeletarFormulario {

private String id;

public DeletarFormulario(String id) {

this.id = id;

}

public void deletar() {

if (id != null && !id.isEmpty()) {

System.out.println("Deletando item com id: " + id);

} else {

System.out.println("ID não fornecido ou inválido!");

}

}

}

public void processarDeletar(String id) {

DeletarFormulario deletarFormulario = new DeletarFormulario(id); deletarFormulario.deletar();

}

}

public class TesteDeletar {

public static void main(String[] args) {

FormularioDeletar formulario = new FormularioDeletar();

formulario.processarDeletar("123"); }

}

**Classe interna estática**

class FormularioDeletar {

static class DeletarFormulario {

private String id;

public DeletarFormulario(String id) {

this.id = id;

}

public void deletar() {

if (id != null && !id.isEmpty()) {

System.out.println("Deletando item com id: " + id);

} else {

System.out.println("ID não fornecido ou inválido!");

}

}

}

public static void processarDeletar(String id) {

DeletarFormulario deletarFormulario = new DeletarFormulario(id); deletarFormulario.deletar(); }

}

public class TesteDeletar {

public static void main(String[] args) {

FormularioDeletar.processarDeletar("123"); }

}

**Classe Anônima**

class FormularioDeletar {

public void processarDeletar(String id) {

DeletarFormulario deletarFormulario = new DeletarFormulario() {

@Override

public void deletar() {

if (id != null && !id.isEmpty()) {

System.out.println("Deletando item com id: " + id);

} else {

System.out.println("ID não fornecido ou inválido!");

}

}

};

deletarFormulario.deletar();

}

interface DeletarFormulario {

void deletar();

}

}

public class TesteDeletar {

public static void main(String[] args) {

FormularioDeletar formulario = new FormularioDeletar();

formulario.processarDeletar("123");

}

}

1. O que é um método final e uma classe final em Java?

Metodo Final

package estrutura.vetor;  
import java.util.Scanner;  
public class Exercicio01 {  
 public static void main(String[] args) {  
 int num[] = new int[10];  
 Scanner sc = new Scanner(System.in);  
 for (int i = 0; i < num.length; i++) {  
 System.out.println("Digite um numero ");  
 num[i] = sc.nextInt();  
 }  
   
 imprimirVetor(num);  
 }  
 public static final void imprimirVetor(int[] vetor) {  
 for (int i = 0; i < vetor.length; i++) {  
 System.out.println(vetor[i]);  
 }  
 }  
}  
 Classe Final

package estrutura.vetor;  
import java.util.Scanner;  
public final class Exercicio02 {  
 public static void main(String[] args) {  
 int num[] = new int[5];  
 Scanner sc = new Scanner(System.in);  
 int soma = 0;  
 for (int i = 0; i < num.length; i++) {  
 System.out.println("Digite um numero");  
 num[i] = sc.nextInt();  
 soma += num[i];  
 }  
 System.out.println("A Soma dos numeros " + soma);  
 }  
}