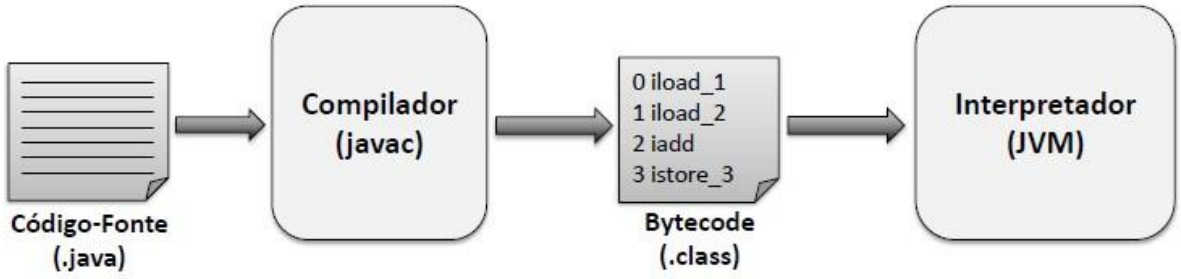
NEWTON PAIVA wyden

Lista de Exercícios da AV1

Professor: Otaviano Martins Monteiro
Aluno(a): Luiz Felipe Vieira de Paula
Curso: Análise e Desenvolvimento de Sistemas
Matrícula: 202451083601

---

1) **As etapas abaixo referem-se à linguagem de programação Java. Explique cada uma dessas etapas:**



Quando o programa Java é criado e salvo pelo usuário é gerado o arquivo (.java), logo em seguida o programa é compilado utilizando o compilador Java (javac). A saída desse processo é um arquivo de bytecode com extensão (.class). O arquivo (.class) é então lido pelo interpretador Java que converte os bytecodes em linguagem de máquina do computador que se está usando (JVM).

2) **Explique o que são tipos de dados primitivos e tipos de dados por referência.**

Os dados primitivos são:
- Byte, Short, Int e Long (utilizados para valores inteiros)
- Float e Double (utilizados para valores reais)
- Char (utilizado para caracteres)
- Boolean (usado para true ou false)

Já os dados de referência são qualquer tipo de dado que não seja primitivo, como:
- Classes
- Objetos
- Arrays
- Strings

3) **Explique o que é programação orientada a objetos.**

A Programação Orientada a Objetos (POO) é uma forma de estruturar códigos de computador que se baseia no conceito de "objetos", que são instâncias de "classes" (os moldes), representando entidades do mundo real com suas próprias características (atributos) e comportamentos (métodos). Essa abordagem organiza o código em unidades lógicas que interagem entre si, utilizando pilares como o encapsulamento (para proteger dados internos), a herança (para reutilizar código) e o polimorfismo (para

permitir que comandos ajam de formas diferentes em objetos distintos), tornando os programas mais intuitivos, fáceis de manter e de expandir.

4) Diferencie programação orientada a objetos da programação estruturada.
A principal diferença é a organização do código:

- Na Programação Estruturada (PE) foca nas ações (funções ou procedimentos) que manipulam os dados. Os dados e as funções ficam separados, e o código é visto como uma sequência de passos a serem executados.

- Na Programação Orientada a Objetos (POO) foca nos objetos, que são estruturas que unem tanto os dados (atributos) quanto as ações (métodos) que os controlam. Essa abordagem trata o programa como um conjunto de entidades interativas (os objetos), facilitando o reuso e a manutenção através de conceitos como encapsulamento e herança.

5) Associe:
A – Modelo
B – Classe
C – Objeto
D – Método
E – Atributo

(E) Representa as características dos objetos
(D) Determina as ações e o comportamento dos objetos de uma determinada classe.
(B) É uma espécie de molde para a criação de objetos.
(C) Consiste na materialização da classe.
(A) É uma representação simplificada de algo do mundo real.

6) Na classe “Carro”, a cor “verde” e a ação de “acelerar” são respectivamente:
(A) Objeto e atributo;
(B) Atributo e método; Resposta: letra B
(C) Atributo e objeto;
(D) Objeto e método;

7) Uma classe “Pessoa” em um sistema acadêmico terá sempre os mesmos atributos e métodos de uma classe “Pessoa” em um sistema de pacientes de uma clínica médica? Explique:

Ambas as classes herdarão atributos básicos (como Nome, DataDeNascimento, e Endereco), mas os atributos e métodos específicos que definem o comportamento e as características da "Pessoa" serão completamente diferentes, pois refletem as prioridades e a lógica de negócios de cada ambiente (um na educação e outro na saúde).

8) Explique os modificadores de acesso private, protected, default e public.

Modificador	Nível de Acesso	Onde é Acessível?	Conceito
public	Mais Amplo	Em qualquer lugar (dentro da classe, em classes filhas, em outras classes no mesmo pacote, e em classes fora do pacote).	É como uma informação **pública e aberta** para todos.
protected	Intermediário	Dentro da classe, e por classes que **herdam** dela (classes filhas), mesmo que estejam em outro pacote. Também é acessível por classes no mesmo pacote.	É como um segredo de **família** que só os herdeiros conhecem.
default	Restrito	Apenas dentro da classe e por outras classes que estão no **mesmo pacote** (pasta lógica).	É um segredo de **vizinhos** (do mesmo pacote).
private	Mais Restrito	**Somente dentro da própria classe** onde foi definido.	É um segredo **pessoal** que ninguém mais pode ver ou tocar.

9) Se não for definido no código, o nível de acesso para um atributo, ele será considerado automaticamente com qual nível de acesso?

Se o modificador de acesso for omitido (não for definido), o atributo será considerado automaticamente com o nível de acesso default.

10) Escreva os nomes dos modificadores de acesso na ordem do mais restrito ao de menor restrição.

Private > Default > Protected > Public

11) Como um atributo privado pode ser acessado através de outra classe?

O acesso a um atributo privado por outra classe não é feito diretamente, mas sim de forma indireta e controlada, através de métodos públicos definidos dentro da própria classe.

- Para ler (acessar) o valor: Usamos o método Getter (ex: getNome()).
- Para modificar (escrever) o valor: Usamos o método Setter (ex: setNome()).

Este mecanismo garante o Encapsulamento, permitindo que a classe valide ou processe os dados antes que eles sejam lidos ou alterados, protegendo a integridade interna do objeto.

12)Explique a palavra reservada this.

Ela serve para que um objeto se refira a si mesmo dentro do seu próprio código (dentro de seus métodos e construtores).

13)O que é um construtor na programação orientada a objetos?

O construtor é um método especial que é chamado automaticamente toda vez que uma nova instância (objeto) de uma classe é criada. Sua função principal é inicializar o objeto, definindo os valores iniciais dos seus atributos e garantindo que o objeto esteja em um estado válido e pronto para ser usado. O construtor sempre tem o mesmo nome da classe e não possui tipo de retorno.

14)Qual é a importância do construtor?

A importância do construtor reside em sua função de garantir a validade e a integridade inicial de um objeto. Ele é crucial porque:

- Garante um Estado Válido: Assegura que todos os atributos essenciais sejam inicializados corretamente no momento da criação, impedindo que o objeto seja usado em um estado incompleto ou inválido.

- Impõe Regras: Permite ao desenvolvedor exigir parâmetros obrigatórios (como um ID ou um nome) e aplicar validações iniciais, forçando que o objeto cumpra as regras de negócio desde o seu nascimento.

15)Explique o que é encapsulamento.

O Encapsulamento é um dos pilares da Programação Orientada a Objetos que visa proteger a integridade dos dados de um objeto. Isso é feito em duas etapas:

- Ocultação: Tornando os atributos (dados internos) da classe private, o que impede o acesso ou a modificação direta por códigos externos.

- Controle: Expondo o acesso a esses dados apenas através de métodos públicos (Getters para leitura e Setters para modificação).

Esses métodos públicos atuam como um filtro, permitindo que o objeto valide as informações antes de aceitá-las ou fornecê-las, garantindo que o objeto mantenha um estado válido e consistente.

16)Explique a sintaxe necessária para criar um objeto no Java.

Para criar um objeto em Java, é necessário seguir três etapas principais: declarar a variável que fará referência ao objeto, instanciá-lo utilizando a palavra-chave new e, se necessário, inicializá-lo por meio de um construtor, passando os valores correspondentes.

“Nome da classe” “nome do objeto” = new “Nome da classe”(parâmetro, se tiver);

Exemplo: Pessoa pessoa1 = new Pessoa(“Luiz”, 29);

17) **Explique os tipos existentes de associação na programação orientada a objetos.**

Os três principais tipos de associação (relacionamentos) em POO, ordenados pela força da dependência, são:

- Dependência (Usa): É o relacionamento mais fraco e temporário, onde uma classe usa a outra (geralmente como um argumento de método), mas não a armazena.

- Agregação (Tem um Fraco): Uma classe contém a outra como atributo, mas ambas têm existências independentes. Se o objeto "todo" for destruído, o objeto "parte" sobrevive.

- Composição (Parte de Forte): É o relacionamento mais forte, onde o objeto "parte" é essencial e tem seu ciclo de vida totalmente dependente do objeto "todo". Se o "todo" for destruído, a "parte" também é destruída.