JAVA BOOTCAMP SANTANDER GIRLS CODE 2022

- Marcela Amorim –

Variáveis, Tipos de Dados e Operadores Matemáticos em Java

1. Variáveis

Uma variável é um recurso das linguagens de programação utilizado para armazenar valores em memória.

Em Java, podemos declarar variáveis, variáveis finais e constantes.

2. Tipos de Dados

Java possui dois tipos de dados que são divididos em tipos primitivos (por valor) e tipos por referência (por referência).

Os tipos primitivos são boolean, byte, char, short, int, long, float e double.

Os tipos por referência são classes que especificam os tipos de objeto Strings, Arrays Primitivos e Objetos.

3. Operadores Matemáticos

Operadores aritméticos são encontrados no início da matemática e incluem adição (+), subtração (-), multiplicação (*), divisão (/) e módulo (%).

Incluem também os operadores unários, ++ e --.

Todos os operadores aritméticos podem ser aplicados a quaisquer primitivos Java, exceto booleano e String. Além disso, somente os operadores de adição + e += podem ser aplicados aos valores de String, o que resulta na concatenação de String.

O módulo, ou operador de resto (%), é simplesmente o resto de uma divisão, quando dois números são divididos.

Entendendo Métodos Java

1. Métodos

Métodos são blocos de código que pertencem a uma classe e tem por finalidade realizar uma tarefa (análogos às funções em C).

Métodos geralmente correspondem a uma ação do objeto, uma sub-rotina que é executada por um objeto ao receber uma mensagem.

Os métodos determinam o comportamento dos objetos de uma classe e são análogos a funções ou procedimentos da programação estruturada.

O envio de mensagens (chamada de métodos) pode alterar o estado de um objeto.

2. Sobrecarga de métodos em Java

A sobrecarga de métodos (overload) é um conceito do polimorfismo que consiste basicamente em criar variações de um mesmo método, ou seja, a criação de dois ou mais métodos com nomes totalmente iguais em uma classe.

Uma classe pode fazer a sobrecarga nos métodos que foram declarados dentro dela e nos métodos herdados.

Métodos sobrecarregados devem possuir o mesmo nome e listas de parâmetros diferentes.

Métodos sobrecarregados podem ter modificadores de acesso e tipos de retorno diferentes, mas eles não podem ser usados para diferenciar um método do outro.

Controle de Fluxos em Java

1. Definição

Controle de fluxo é a habilidade de ajustar a maneira como um programa realiza suas tarefas. Por meio de instruções especiais essas tarefas podem ser executadas seletivamente, repetidamente ou excepcionalmente.

Sem o controle de fluxo, um programa poderia executar apenas uma única sequência de tarefas, perdendo completamente a dinâmica que é uma das características mais interessantes da programação.

Estruturas de Repetição e Arrays em Java

1. Estruturas de Repetição

As estruturas de repetição também são conhecidas como laços (loops) e são utilizados para executar, repetidamente, uma instrução ou bloco de instrução enquanto determinada condição estiver sendo satisfeita.

Qualquer que seja a estrutura de repetição, ela contém quatro elementos fundamentais: inicialização, condição, corpo e iteração.

A inicialização compõe-se de todo código que determina a condição inicial da repetição.

A condição é uma expressão booleana avaliada após cada leitura do corpo e determina se uma nova leitura deve ser feita ou se a estrutura de repetição deve ser encerrada.

O corpo compõe-se de todas as instruções que são executadas repetidamente.

A iteração é a instrução que deve ser executada depois do corpo e antes de uma nova repetição.

2. Arrays em Java

Um array é uma estrutura de dados usada para armazenar dados do mesmo tipo.
Os arrays armazenam seus elementos em localizações sequenciais contínuas da memória.

Em Java, arrays são objetos. Todos os métodos da classe Object podem ser invocados em um array.

Collections Java

1. Definição

Na versão atual de Java, uma coleção é um objeto que pode agrupar outros objetos.
Uma coleção tem por objetivo facilitar a inserção, busca e recuperação destes objetos agrupados sistematicamente.

Collections Framework é um conjunto bem definido de interfaces e classes para representar e tratar grupos de dados como uma única unidade, que pode ser chamada coleção, ou collection.

2. Elementos

A Collections Framework contém os seguintes elementos:

Interfaces: tipos abstratos que representam as coleções. Permitem que coleções sejam manipuladas tendo como base o conceito "Programar para interfaces e não para implementações", desde que o acesso aos objetos se restrinja apenas ao uso de métodos definidos nas interfaces.

Implementações: são as implementações concretas das interfaces.

Algoritmos: são os métodos que realizam as operações sobre os objetos das coleções, tais como busca e ordenação.

Paradigma de Programação Orientado a Objetos (POO)

1. Visão

A visão de Orientação a Objetos (OO) é aquela de um mundo de objetos que interagem.

Este paradigma é um modelo de análise, projeto e programação baseado na aproximação entre o mundo real e o mundo virtual, através da criação e interação entre classes, atributos, métodos, objetos, entre outros.

2. Pilares

São 4 os pilares principais do POO: ABSTRAÇÃO, ENCAPSULAMENTO, HERANÇA E POLIMORFISMO.

- ABSTRAÇÃO:

Habilidade de concentrar-se nos aspectos essenciais de um domínio, ignorando características menos importantes ou acidentais. Nesse contexto, objetos são abstrações de entidades existentes no domínio em questão.

- ENCAPSULAMENTO:

Encapsular significa esconder a implementação dos objetos. O encapsulamento favorece principalmente dois aspectos de um sistema: a manutenção e a evolução.

- HERANÇA:

Permite que você defina uma classe filha que reutiliza (herda), estende ou modifica o comportamento de uma classe pai. A classe cujos membros são herdados é chamada de classe base. A classe que herda os membros da classe base é chamada de classe derivada.

- POLIMORFISMO:

Capacidade de um objeto poder ser referenciado de várias formas. Cuidado, polimorfismo não quer dizer que o objeto fica se transformando, muito pelo contrário, um objeto nasce de um tipo e morre daquele tipo, o que pode mudar é a maneira como nos referimos a ele. A capacidade de tratar objetos criados a partir das classes específicas como objetos de uma classe genérica é chamada de polimorfismo.

3. Conceitos Fundamentais POO

- DOMÍNIO:

Domínio da aplicação, também conhecida como camada de negócio ou de objetos de negócio, é aquela onde estão localizadas as classes que fazem parte do domínio do problema, ou seja, classes correspondentes a objetos que fazem parte da descrição do problema.

- CLASSE:

Um elemento do código que tem a função de representar objetos do mundo real. Dentro dela é comum declararmos atributos e métodos, que representam, respectivamente, as características e comportamentos desse objeto.

- ATRIBUTO:

Atributos são, basicamente, a estrutura de dados que vai representar a classe. Os atributos também são conhecidos como VARIÁVEL DE CLASSE, e podem ser divididos em dois tipos básicos: atributos de instância e de classe.

- VARIÁVEL:

Uma “região de memória (do computador) previamente identificada cuja finalidade é armazenar os dados ou informações de um programa por um determinado espaço de tempo”.

- MÉTODO:

Os métodos representam os estados e ações dos objetos e classes.

- OBJETO:

Em POO, objeto é um "molde" de uma determinada classe, que passa a existir a partir de uma instância da classe. A classe define o comportamento do objeto, usando atributos (propriedades) e métodos (ações). Objeto em ciência da computação, é uma referência a um local da memória que possui um valor. Um objeto pode ser uma variável, função, ou estrutura de dados.

- INSTÂNCIA:

Uma instância de uma classe é um novo objeto criado dessa classe, com o operador new. Instanciar uma classe é criar um novo objeto do mesmo tipo dessa classe. Uma classe somente poderá ser utilizada após ser instanciada.

Linguagem de Programação vs Paradigma de Linguagem de Programação

1. Linguagem de Programação

É uma linguagem formal que, através de uma série de instruções, permite que um programador escreva um conjunto de ordens, ações consecutivas, dados e algoritmos para criar programas que controlam o comportamento físico e lógico de uma máquina.

Seguem alguns exemplos de como as linguagens de programação podem ser classificadas:

- Nível de abstração:

Baixo Nível: Assembly

Médio Nível: C, C++, D, Objective C, etc.

Alto Nível: Java, C#, PHP, Javascript, etc.

Altíssimo Nível: Python, Ruby, Elixir, etc.

- Paradigma de programação:

Programação Estruturada: C, Pascal, Ada, etc.

Programação Orientada a Objetos: Java, C#, C++, Objective C, D, etc.

Programação Funcional: Lisp, Scheme, Erlang, Elixir, etc.

- Linguagens classificadas pela arquitetura da aplicação:

Desktop: C, C++, Object Pascal, Java, etc.

Web: PHP, Ruby, Javascript, Java, etc.

- Tipo de execução:

Linguagens compiladas: C, C++, Pascal, D, GO, etc.

Linguagens Interpretadas: Python, Ruby, PHP, Javascript, etc.

Linguagens Hibridas: Java, Erlang, Elixir, etc.

2. Paradigma de Linguagem de Programação

É um conjunto de características que podem ser utilizados para categorizar determinado grupo de linguagens. Um paradigma pode oferecer técnicas apropriadas para uma aplicação específica.

3. Paradigma Principais e seus Subparadigmas

- 1. Paradigma Imperativo

Neste paradigma, o programa descreve o processamento necessário para solucionar o problema. Assim, o paradigma imperativo é caracterizado por execução sequencial de instruções, pelo uso de variáveis que representam posições de memória e pelo uso de instruções de atribuição que alteram os valores dessas variáveis.

Vejamos alguns Subparadigmas do Paradigma Imperativo e exemplos linguagens de programação que adotam esses subparadigmas.

- 1.1 Paradigma estruturado: ALGOL 58 e ALGOL 60
- 1.2 Paradigma concorrente: Java e Ada
- 1.3 Paradigma Orientado a Objetos: Smalltalk e Java

- 2. Paradigma Declarativo

Este paradigma é o modelo no qual os resultados são descritos, mas os passos para chegar aos resultados não são estabelecidos.

Vejamos alguns Subparadigmas do Paradigma Declarativo e exemplos linguagens de programação que adotam esses subparadigmas:

- 2.1 Paradigma Funcional: Lisp e Haskell
- 2.2 Paradigma Lógico: Prolog