
SENAI

FUNDAMENTOS DE JAVA
TÓPICOS ABORDADOS

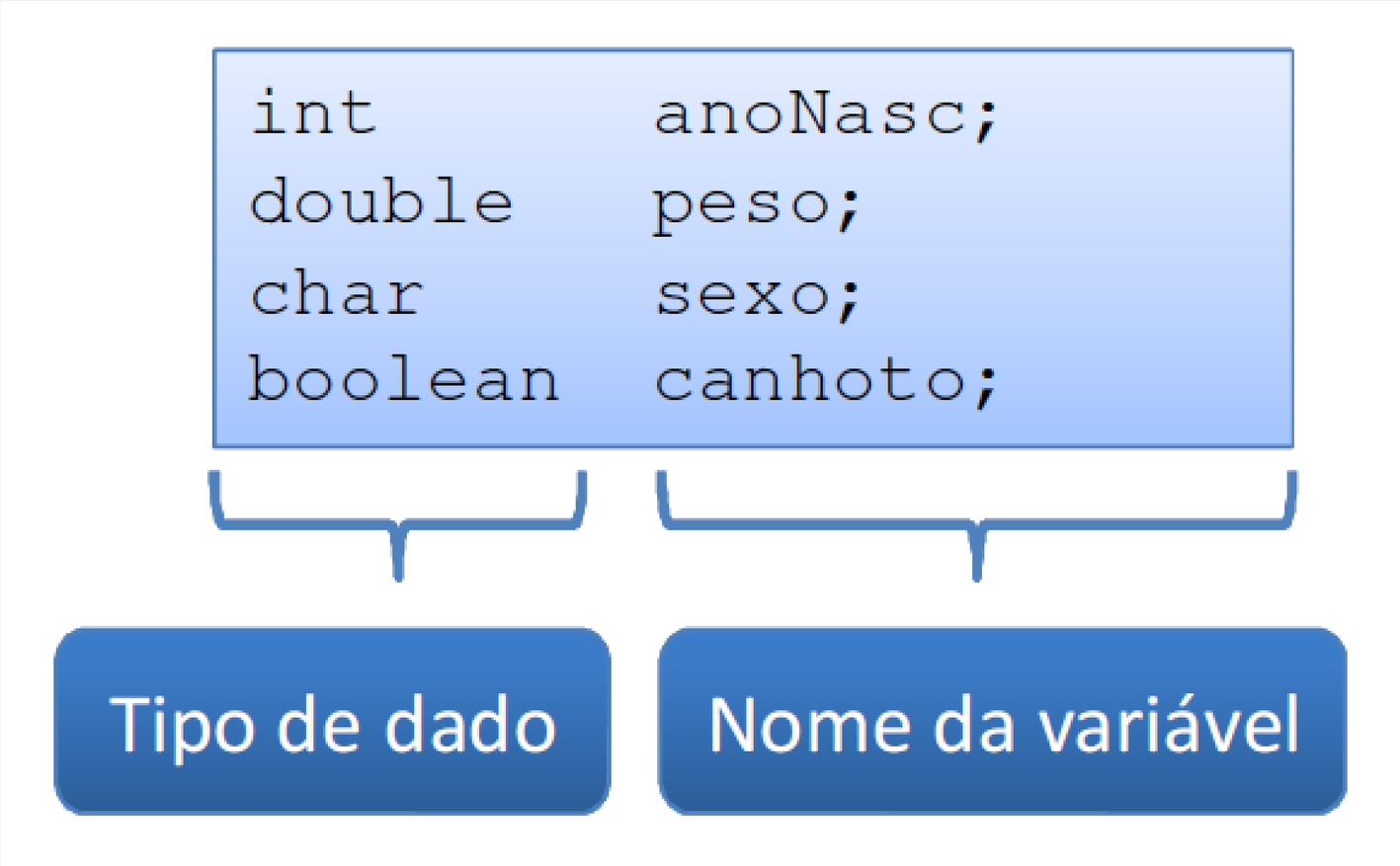
- Variáveis
  - Declaração, inicialização e alteração de valor
- Tipos primitivos do Java
- Operadores
  - Aritméticos, Comparação, Lógicos.
- O tipo de dados char
- O que são String’s
- Entrada de Dados - Scanner
- Comentando código em Java

FUNDAMENTOS DE JAVA: **VARIÁVEIS**

SENAI

FUNDAMENTOS DE JAVA
DECLARAÇÃO DE VARIÁVEIS

- Variáveis devem possuir um tipo e um nome



FUNDAMENTOS DE JAVA
INICIALIZAÇÃO DE VARIÁVEIS

- Para inicializar variáveis,
  utilizamos o operador "=" (atribuição)

```
anoNasc = 1980;
peso = 65.7;
sexo = 'M';
```

- É possível também declarar e inicializar simultaneamente

```
double altura = 1.8;
```

- O Java não inicializa as variáveis automaticamente

FUNDAMENTOS DE JAVA
ALTERAÇÃO DO VALOR DE VARIÁVEIS

- Outros exemplos de uso de variáveis

```
int contador = 20;
int novoContador = contador + 1;
```

novoContador = 21

```
int x = 15;
x = x + 1;
```

x = 16

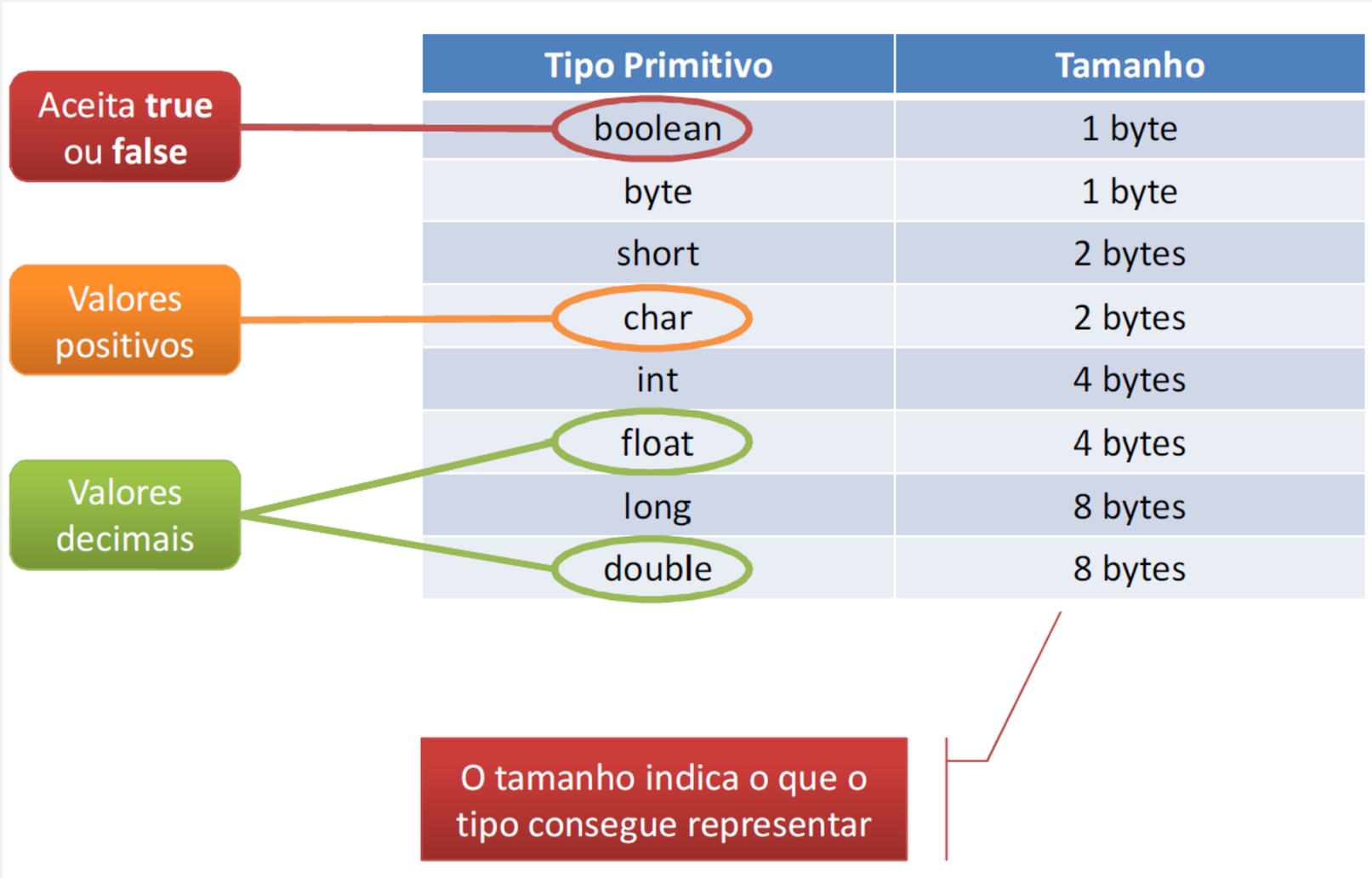
```
int y = x + x - 10;
```

y = 22

FUNDAMENTOS DE JAVA: **TIPOS PRIMITIVOS**

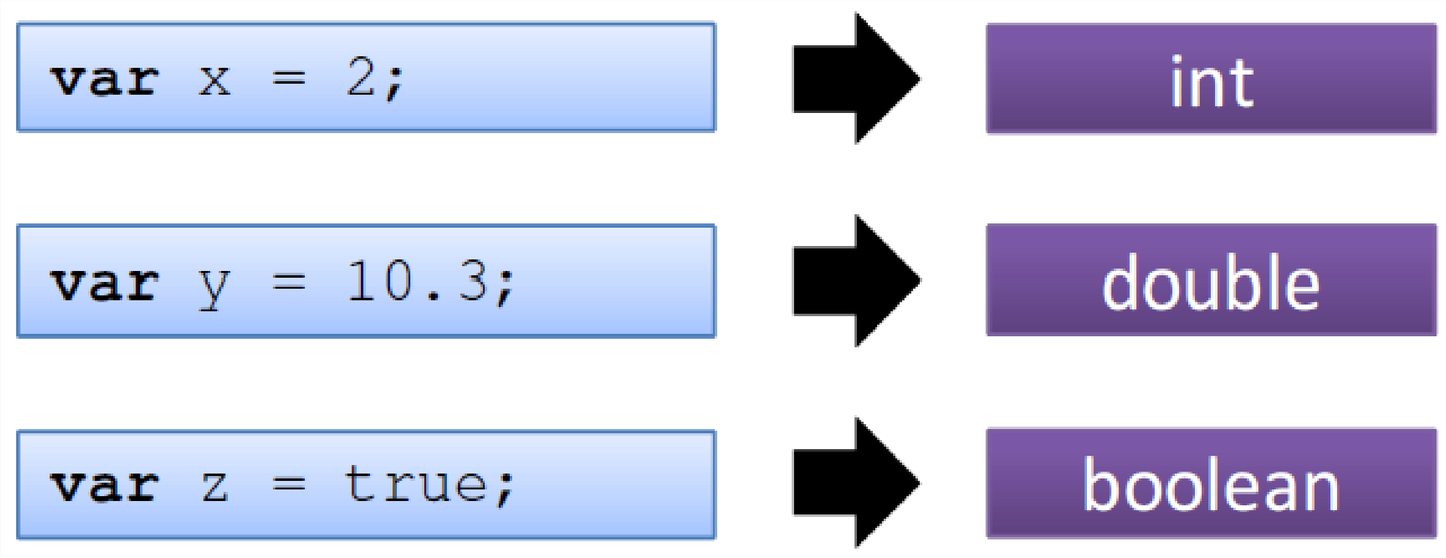
SENAI

FUNDAMENTOS DE JAVA
TIPOS PRIMITIVOS DO JAVA



FUNDAMENTOS DE JAVA
A VARIÁVEL *VAR*

- A partir do Java 10 é possível declarar uma variável como **var.**
- O tipo que a variável vai assumir vai depender do valor colocado nela.



FUNDAMENTOS DE JAVA: **OPERADORES**

SENAI

FUNDAMENTOS DE JAVA
OPERADORES ARITMÉTICOS

SENAI

Operador	Descrição
+	Soma
-	Subtração
*	Multiplicação
/	Divisão
%	Módulo

FUNDAMENTOS DE JAVA
OPERADORES DE COMPARAÇÃO

SENAI

Operador	Descrição
==	Igual
!=	Diferente
<	Menor que
>	Maior que
<=	Menor ou igual a
>=	Maior ou igual a

FUNDAMENTOS DE JAVA
OPERADORES LÓGICOS

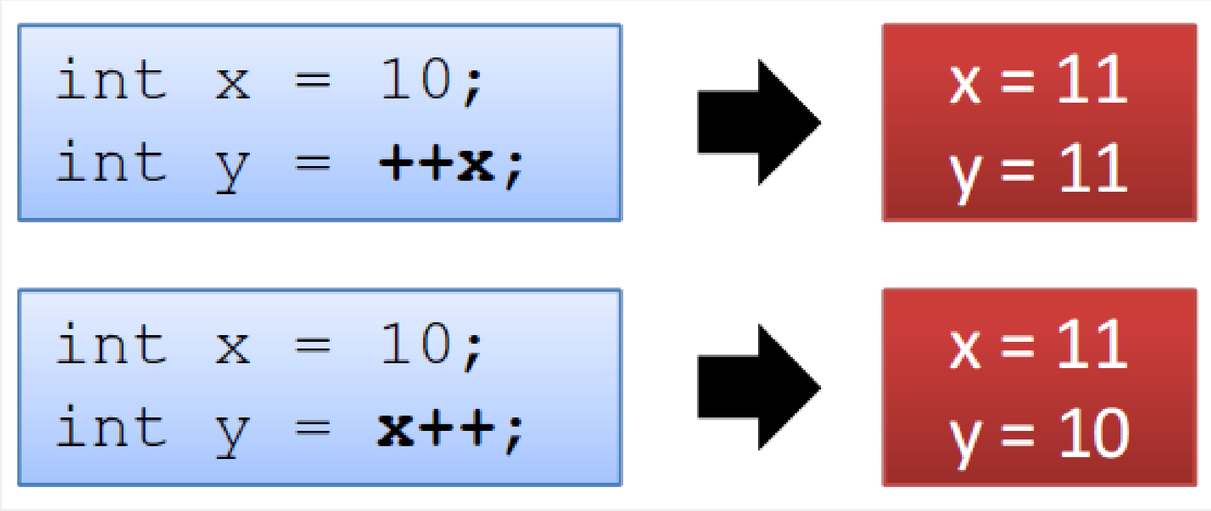
Operador	Descrição
!	Negação
||	OU
&&	E

FUNDAMENTOS DE JAVA
OUTROS OPERADORES IMPORTANTES

Operador	Descrição	Exemplo
++	Incremento	x++
--	Decremento	x--
+=	Soma com valor e atribui o resultado à própria variável	x += 2
-=	Subtrai do valor e atribui o resultado à própria variável	x -= 5
*=	Multiplica pelo o valor e atribui o resultado à própria variável	x *= 3
/=	Divide pelo valor e atribui o resultado à própria variável	x /= 4

FUNDAMENTOS DE JAVA
OPERADORES DE INCREMENTO E DECREMENTO

- Os operadores de incremento (“++”) e decremento (“--”) podem ser de dois tipos
  - Pré-fixados
    - Ex: ++x;
  - Pós-fixados
    - Ex: x++;



FUNDAMENTOS DE JAVA: **CASTING**

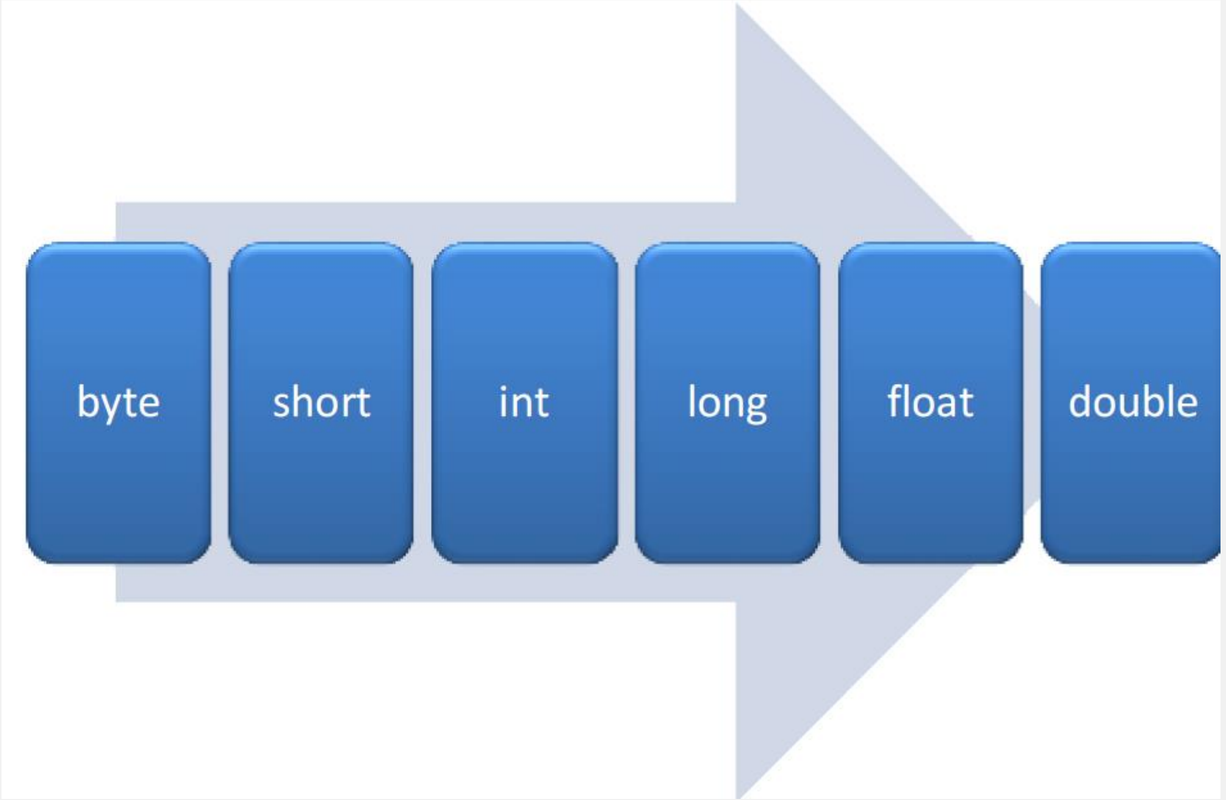
SENAI

FUNDAMENTOS DE JAVA
CASTING

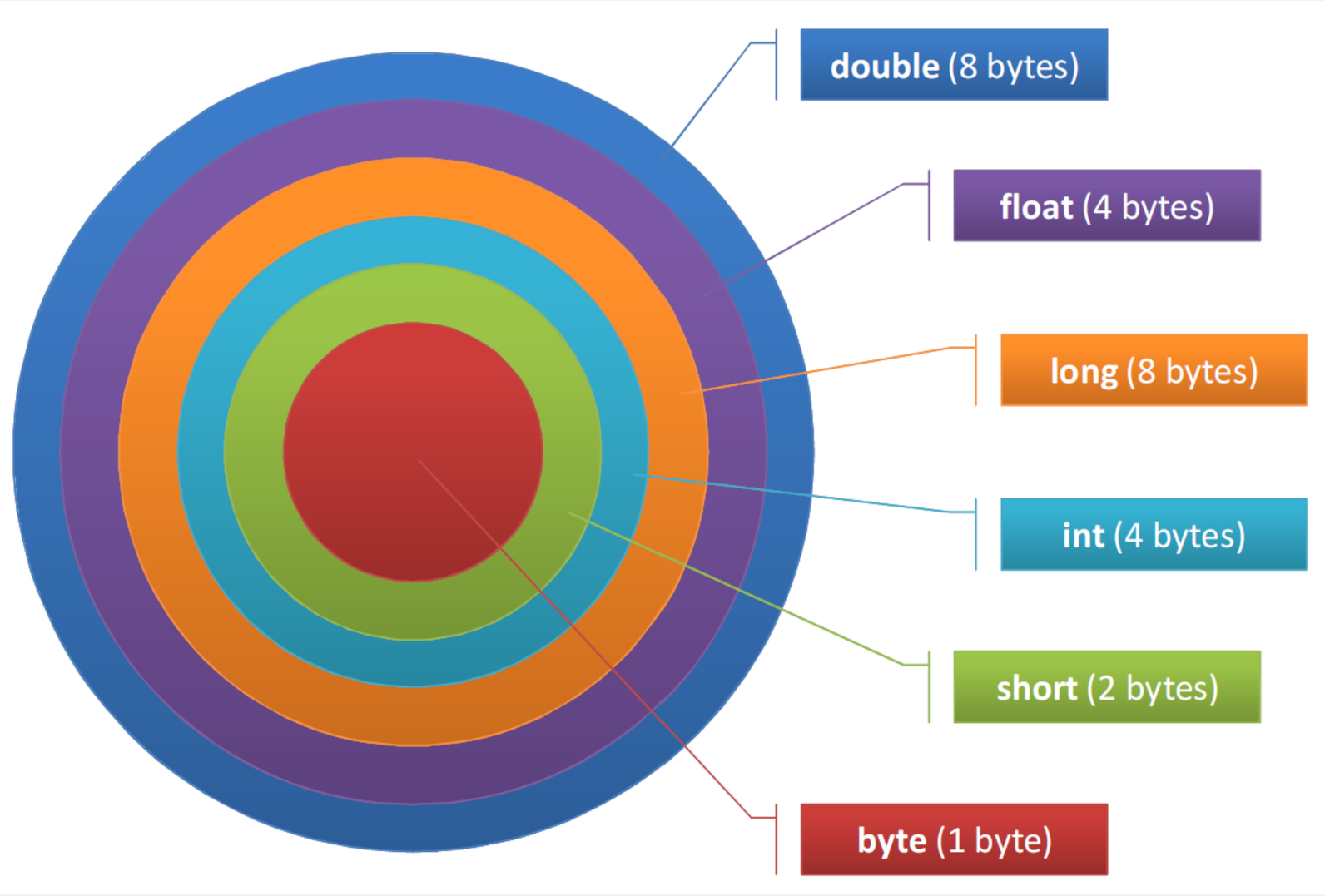
- O casting consiste em atribuir uma variável/valor de um tipo a uma variável de outro tipo
- Podem ser **implícitos** ou **explícitos**

FUNDAMENTOS DE JAVA
CASTING IMPLÍCITO

- O Java faz a conversão do tipo de dado automaticamente



FUNDAMENTOS DE JAVA
CASTING IMPLÍCITO



FUNDAMENTOS DE JAVA
EXEMPLOS DE CASTING IMPLÍCITO

SENAI

```
long n1 = 10;
```

10 é do tipo **int** e pode ser atribuído a uma variável **long**

```
float n2 = 5L;
```

5L é do tipo **long** e pode ser atribuído a uma variável **float**

```
double n3 = 2.3f;
```

2.3f é do tipo **float** e pode ser atribuído a uma variável **double**

```
int n4 = 3.5;
```

3.5 é do tipo **double** e não pode ser atribuído a uma variável **int**. É necessário um casting explícito.

FUNDAMENTOS DE JAVA
CASTING EXPLÍCITO

SENAI

- A conversão deve ser feita pelo programador

```
double d = 100.0;
int i = d;
```

```
double d = 100.0;
int i = (int) d;
```

- Cuidado com o casting explícito!

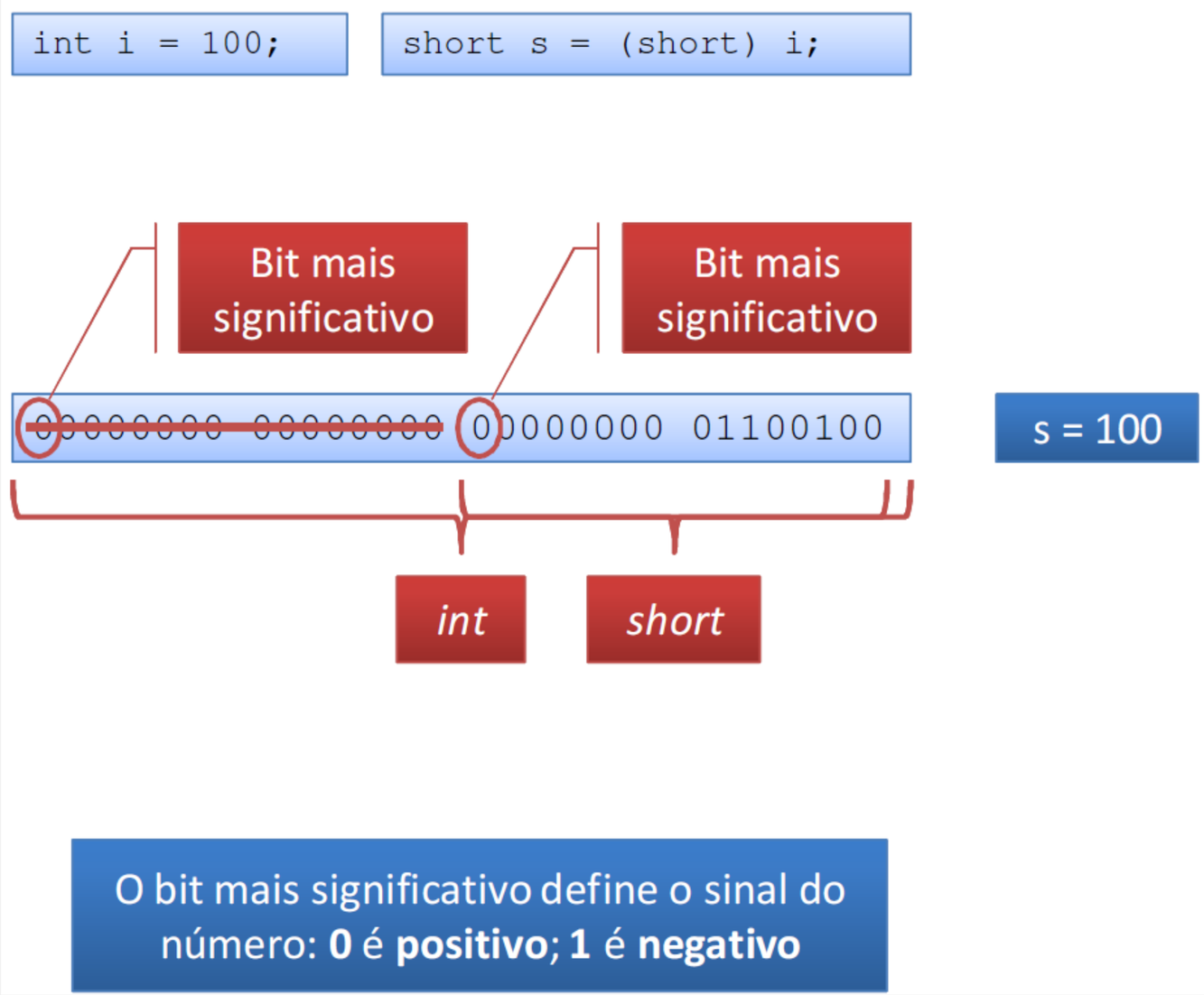
```
int n1 = (int) 3.5;
```

O resultado é **3**.
Como o *int* não armazena a parte decimal, ela é perdida.

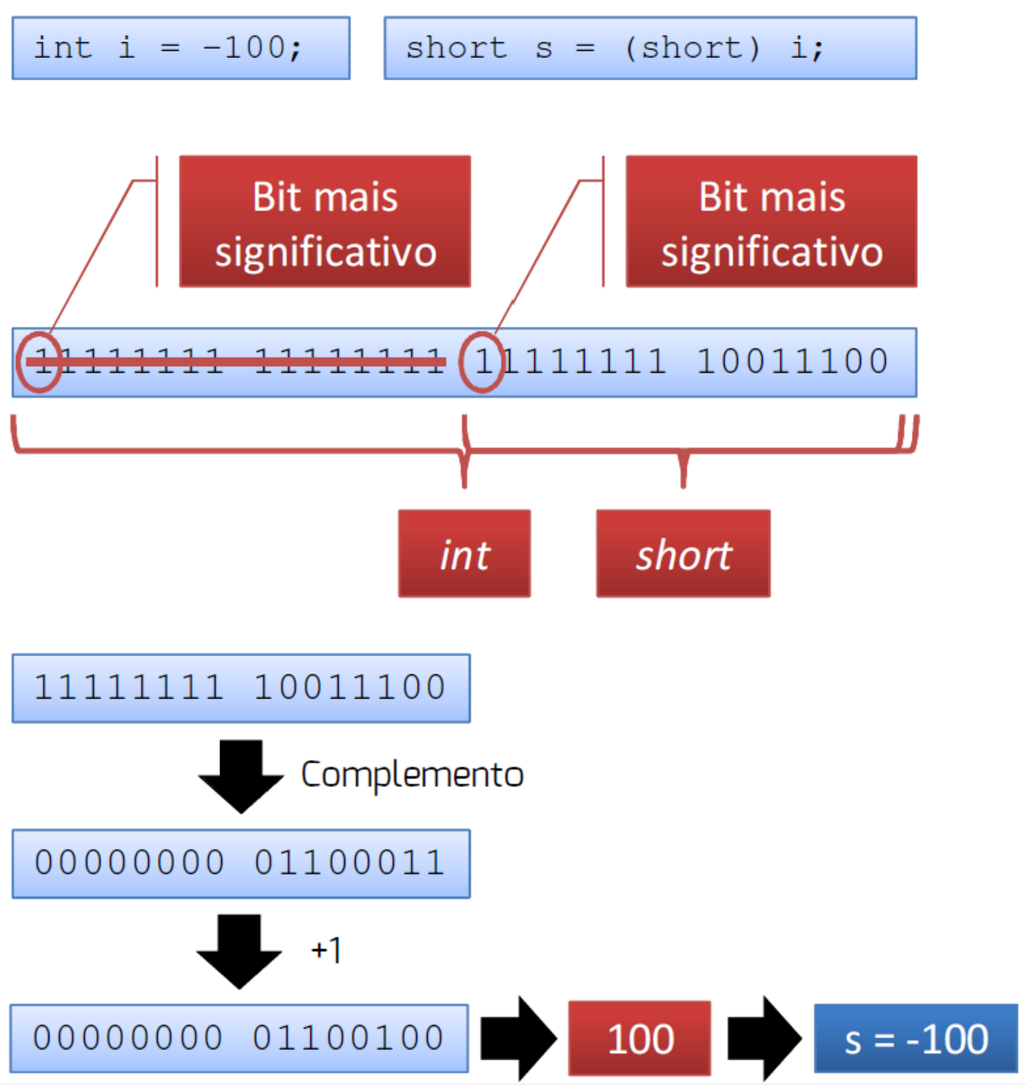
```
byte n2 = (byte) 129;
```

O resultado é **-127**.
O número 129 é muito grande para caber dentro de uma variável do tipo *byte*.

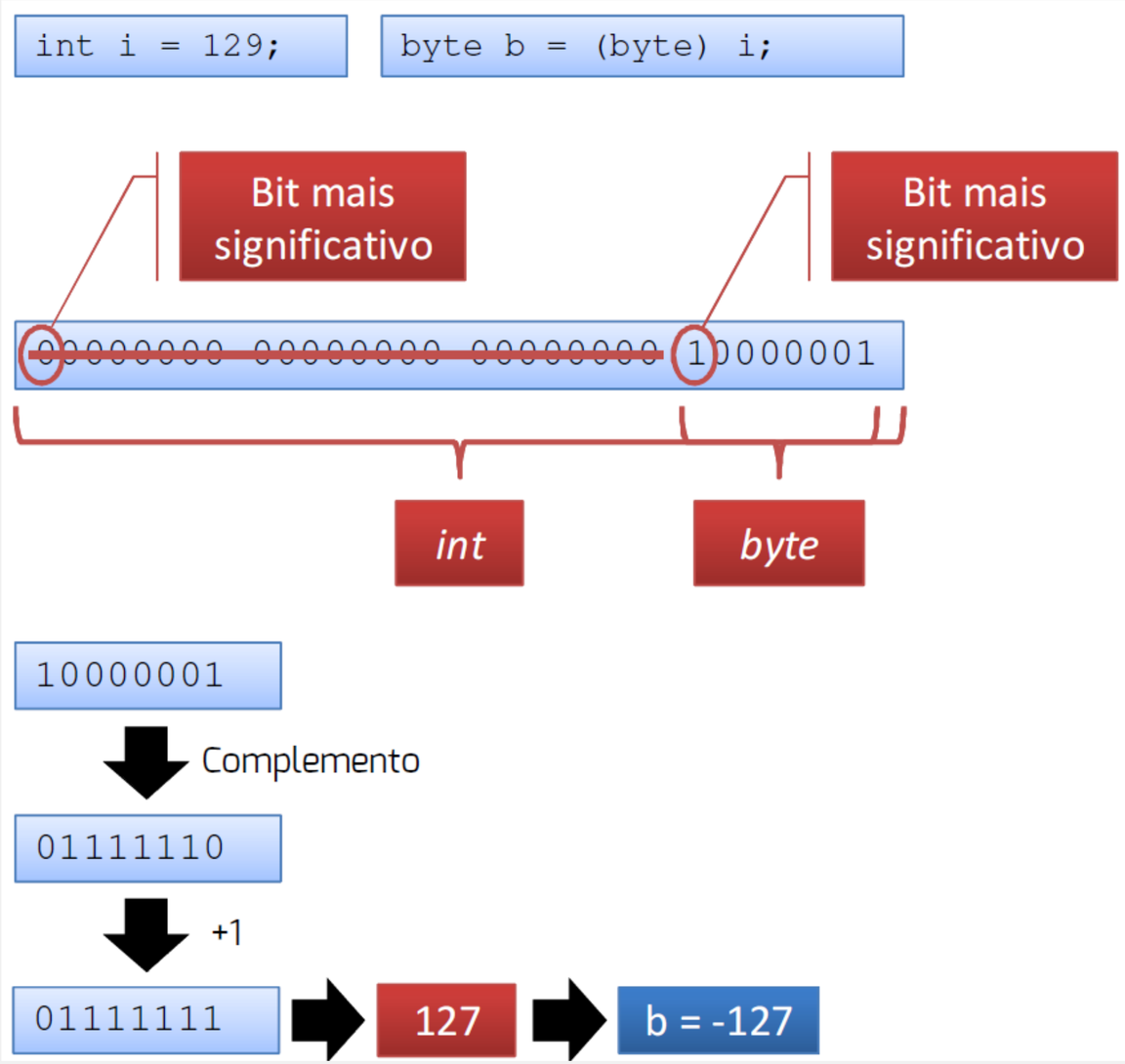
FUNDAMENTOS DE JAVA
CASTING EXPLÍCITO: EXEMPLO 1



FUNDAMENTOS DE JAVA
CASTING EXPLÍCITO: EXEMPLO 2



FUNDAMENTOS DE JAVA
CASTING EXPLÍCITO: EXEMPLO 3



FUNDAMENTOS DE JAVA: **CHAR**

SENAI

FUNDAMENTOS DE JAVA
TIPOS DE DADOS *CHAR*

- O **char** é o único tipo primitivo em Java sem sinal
- Um **char** indica um caractere, sendo utilizadas aspas simples na sua representação

```
char c = 'A';
```

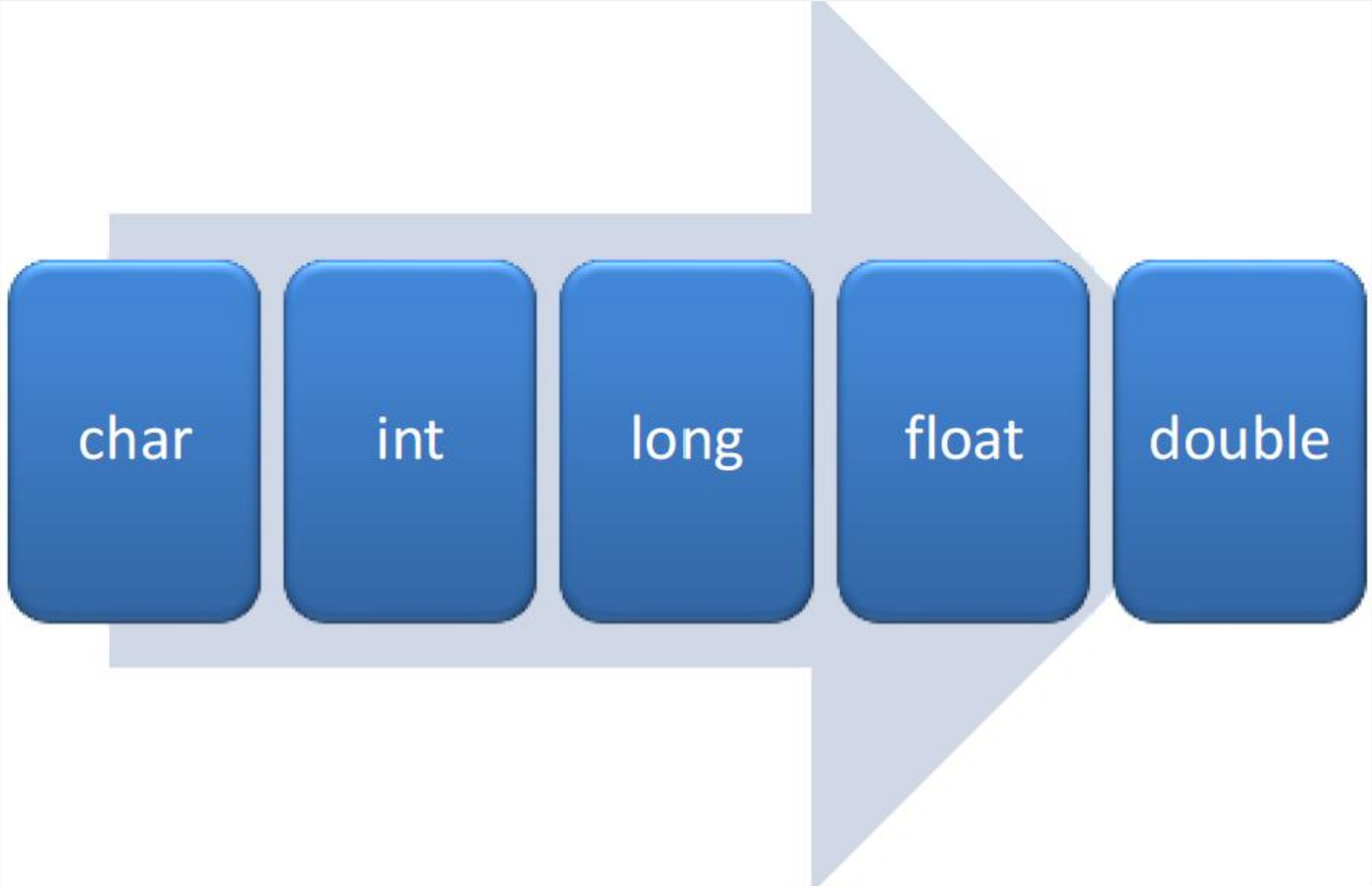
- A atribuição de números a um **char** também é válida

```
char c = 65;
```

Código ASCII do 'A'

FUNDAMENTOS DE JAVA
TIPOS DE DADOS *CHAR*

- O **cast** implícito ocorre a partir do tipo **int**



FUNDAMENTOS DE JAVA: **STRINGS**

SENAI

FUNDAMENTOS DE JAVA
O QUE SÃO STRINGS?

- Uma **String** é uma sequência de caracteres.
  Exemplos: “Olá, mundo!”, “12345”, “@”.
- Diferente de outras linguagens, em Java, uma **String** não é um tipo de dado primitivo.
- É um Objeto e por isso contém diversos métodos que podem ser usados (esse conceito será explicado em outras aulas).
- Tem comprimento variável, suportam desde um caractere até o conteúdo de várias páginas de um livro.

FUNDAMENTOS DE JAVA
DECLARAÇÃO DE UMA STRING?

- Em Java, a classe **String** inicia com letra maiúscula por convenção, seguindo o padrão de nomenclatura para classes.
- Os valores atribuídos a variáveis do tipo **String** devem estar entre aspas duplas para indicar que se trata de uma sequência de caracteres.

```
String s = "abc";
```

FUNDAMENTOS DE JAVA
NOVO RECURSO PARA STRINGS (“““)

- A partir do Java 15, foi introduzido um novo recurso sintático que simplifica a criação de strings de múltiplas linhas:

```
// Antes do Java 15:
String textoMultiLinha = "Este é um texto de múltiplas linhas.\n" +
                         "Ele pode se estender por várias linhas\n" +
                         "mas requer o uso de caracteres de escape.";

// A partir do Java 15:
String textoMultiLinha = """
    Este é um texto de múltiplas linhas.
    Ele pode se estender por várias linhas
    sem a necessidade de caracteres de escape.
""";
```

FUNDAMENTOS DE JAVA: **ENTRADA DE DADOS: SCANNER**

SENAI

FUNDAMENTOS DE JAVA
ENTRADA DE DADOS

- Para que nossos programas sejam interativos, vamos aprender a receber dados do usuário diretamente pelo console.
- Os métodos **print’s** nos ajudam a fazer perguntas.
- Para capturar as respostas (entrada de dados) iremos usar o **Scanner.**

FUNDAMENTOS DE JAVA
SCANNER

- O **Scanner** faz parte da biblioteca **java.util**. Ela permite capturar a entrada de diferentes tipos de dados, como **inteiros**, **floats**, **strings**, entre outros, diretamente do console.
- Antes de usar o Scanner, você precisa importar a classe.

```
import java.util.Scanner;
```

FUNDAMENTOS DE JAVA
SCANNER

- **Criar um objeto Scanner:** O Scanner deve ser instanciado para poder capturar as entradas do usuário.
    - **Atenção:** Para instanciar o objeto **Scanner** apenas copie a linha de comando abaixo, mais a frente iremos aprender o conceito de **objetos** e de como funciona cada comando a seguir:

```
Scanner scanner = new Scanner(System.in);
```

FUNDAMENTOS DE JAVA
SCANNER

- **Receber diferentes tipos de dados:** O Scanner possui métodos específicos para capturar tipos de dados diferentes.
- Alguns exemplos são:
  - **nextInt()** para inteiros.
  - **nextDouble()** para números decimais **(double).**
  - **nextFloat()** para números decimais menores **(float).**
  - **next()** para capturar uma única palavra **(String).**
  - **nextLine()** para capturar uma linha inteira de texto**.**

FUNDAMENTOS DE JAVA
SCANNER

- **Encerrar o uso do Scanner:** Para evitar desperdício de recursos, é uma boa prática fechar o Scanner ao final do programa.

```
scanner.close();
```

FUNDAMENTOS DE JAVA
SCANNER - EXEMPLO 1

- Receber um número inteiro do usuário

```java
import java.util.Scanner;

public class ReceberInteiro {
    public static void main(String[] args) {
        // Criando um objeto Scanner
        Scanner scanner = new Scanner(System.in);

        // Solicitando um número inteiro ao usuário
        System.out.print("Digite um número inteiro: ");
        int numero = scanner.nextInt();

        // Exibindo o número digitado
        System.out.println("Você digitou o número: " + numero);

        // Fechando o Scanner
        scanner.close();
    }
}
```

FUNDAMENTOS DE JAVA
SCANNER - EXEMPLO 2

- Receber uma string (texto) do usuário

```java
import java.util.Scanner;

public class ReceberTexto {
    public static void main(String[] args) {
        // Criando um objeto Scanner
        Scanner scanner = new Scanner(System.in);

        // Solicitando o nome do usuário
        System.out.print("Digite o seu nome: ");
        String nome = scanner.nextLine();

        // Exibindo o nome digitado
        System.out.println("Olá, " + nome + "!");

        // Fechando o Scanner
        scanner.close();
    }
}
```

FUNDAMENTOS DE JAVA
SCANNER - EXEMPLO 3

- Receber múltiplos tipos de dados

```java
import java.util.Scanner;

public class ReceberVariosDados {
    public static void main(String[] args) {
        // Criando um objeto Scanner
        Scanner scanner = new Scanner(System.in);

        // Solicitando o nome
        System.out.print("Digite o seu nome: ");
        String nome = scanner.nextLine();

        // Solicitando a idade
        System.out.print("Digite a sua idade: ");
        int idade = scanner.nextInt();

        // Solicitando o peso
        System.out.print("Digite o seu peso (kg): ");
        double peso = scanner.nextDouble();

        // Exibindo os dados recebidos
        System.out.println("Nome: " + nome);
        System.out.println("Idade: " + idade);
        System.out.println("Peso: " + peso + " kg");

        // Fechando o Scanner
        scanner.close();
    }
}
```

FUNDAMENTOS DE JAVA
PONTO DE ATENÇÃO

- **Problema com nextLine() após nextInt():** Quando você usa o **nextInt()** ou **nextDouble(),** o Scanner não consome a linha de separação (nova linha), por isso, se você usar **nextLine()** logo após um **nextInt()**, pode haver um problema de captura da entrada. Para evitar isso, adicione um **scanner.nextLine()** extra após a leitura de um número, para consumir a linha restante.

```
scanner.nextLine(); // Consome a quebra de linha após o nextInt ou nextDouble
```

FUNDAMENTOS DE JAVA: **COMÉNTANDO O CÓDIGO**

SENAI

FUNDAMENTOS DE JAVA
COMENTANDO CÓDIGO JAVA

SENAI

- Comentários em uma linha (//)

```
//exemplo de comentário no código em apenas uma linha
```

- Comentários em múltiplas linhas (/* */)

```
/*
Exemplo de comentário no código onde mais de uma linha
pode ser utilizada
*/
```

FUNDAMENTOS DE JAVA
COMENTANDO CÓDIGO JAVA

- No Intellij existe combinações de teclas de atalho para fazer comentários:
  - **Ctrl + /** : Comenta a linha selecionada.
  - **Ctrl + Shift + /** : Comenta várias linhas selecionadas criando um bloco de comentários.

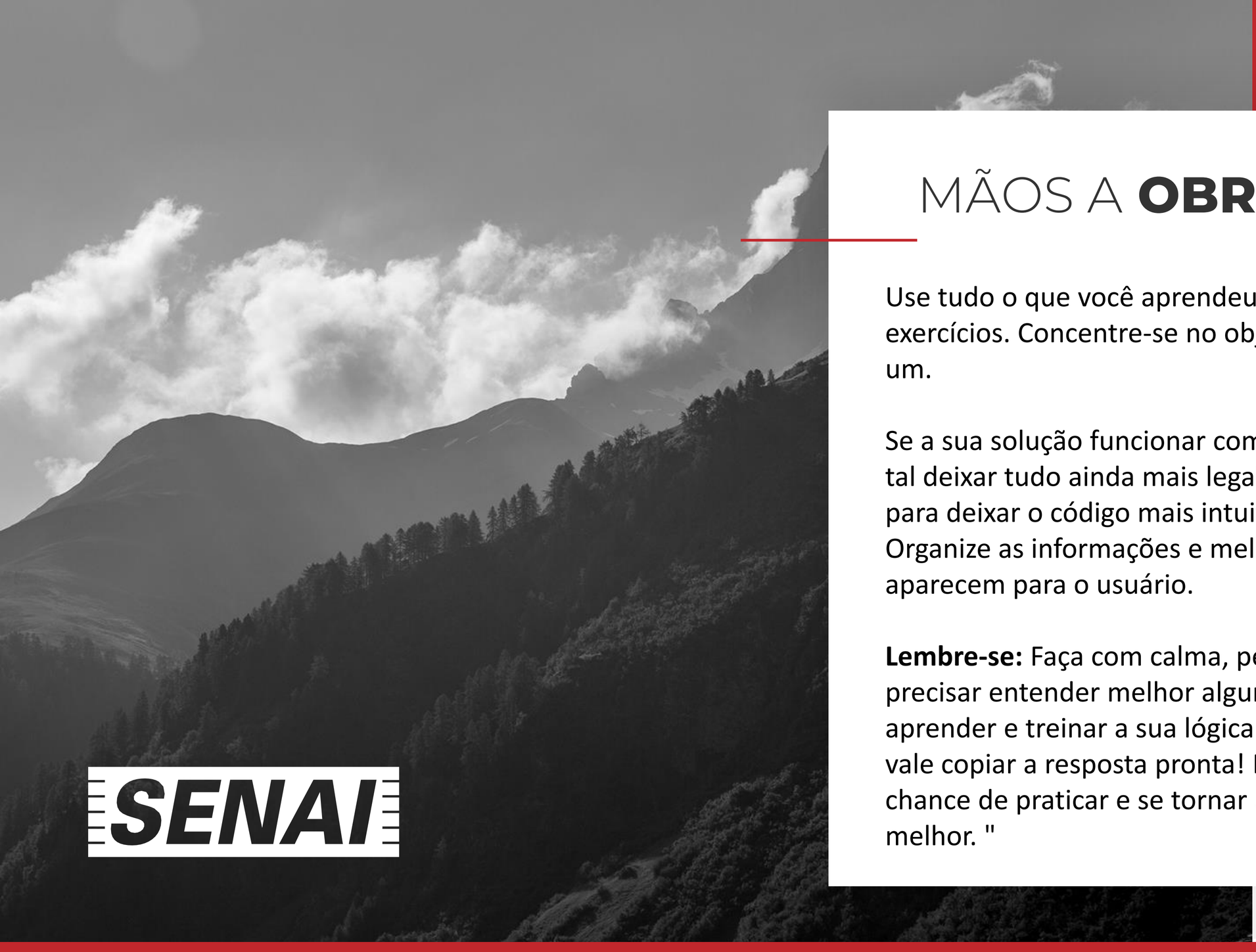
FUNDAMENTOS DE JAVA

EXERCÍCIOS:

CRIAÇÃO E UTILIZAÇÃO **DE VARIÁVEIS:**

- TIPOS DE DADOS
- OPERADORES
  - ARITIMÉTICOS
  - COMPARAÇÃO
  - LÓGICOS
- CASTING

SENAI



SENAI

MÃOS A **OBRA**

Use tudo o que você aprendeu para resolver os exercícios. Concentre-se no objetivo principal de cada um.

Se a sua solução funcionar como o exercício pede, que tal deixar tudo ainda mais legal? Use a sua criatividade para deixar o código mais intuitivo e fácil de entender. Organize as informações e melhore os textos que aparecem para o usuário.

Lembre-se: Faça com calma, peça ajuda quando precisar entender melhor alguma coisa. O importante é aprender e treinar a sua lógica de programação. Não vale copiar a resposta pronta! Esses exercícios são a sua chance de praticar e se tornar um programador ainda melhor. "

FUNDAMENTOS DE JAVA
EXERCÍCIOS

SENAI

Exercício 1: Declaração e Inicialização de Variáveis

Declare e inicialize variáveis para armazenar a idade (int), altura (float), e peso (double) de uma pessoa. Atribua valores às variáveis e exiba-os no console.

Exercício 2: Operações Matemáticas Simples

Crie um programa que declare duas variáveis inteiras para representar a quantidade de horas trabalhadas e o valor pago por hora. Calcule o salário total (int) multiplicando essas duas variáveis e exiba o resultado no console.

FUNDAMENTOS DE JAVA
EXERCÍCIOS

Exercício 3: Casting Implícito

Declare uma variável do tipo int e atribua a ela um valor. Em seguida, crie uma variável do tipo double e atribua a ela o valor da variável int. Exiba o valor da variável double no console.

Exercício 4: Casting Explícito

Declare uma variável do tipo double com um valor fracionado. Realize o casting explícito para uma variável do tipo int e exiba ambos os valores no console.

FUNDAMENTOS DE JAVA
EXERCÍCIOS

SENAI

Exercício 5: Operadores Aritméticos

Crie um programa que declare três variáveis inteiras para representar as notas de três provas. Calcule e exiba a soma total das notas.

Exercício 6: Comparação de Dois Números

Declare duas variáveis inteiras com valores diferentes. Compare-as utilizando operadores de comparação (>, <, >=, <=, ==, !=) e exiba os resultados das comparações no console.

Exercício 7: Operadores Lógicos Básicos

Crie um programa que declare três variáveis booleanas representando condições quaisquer (ex: cond1, cond2, cond3). Utilize os operadores lógicos AND (&&) e OR (||) para combinar essas condições e exiba os resultados no console.

Exercício 8: Cálculo de Área de um Retângulo

Declare duas variáveis do tipo float para armazenar a largura e a altura de um retângulo. Calcule a área do retângulo multiplicando essas variáveis e exiba o resultado.

FUNDAMENTOS DE JAVA
EXERCÍCIOS

Exercício 9: Conversão de Tipos (Casting Implícito e Explícito)

Declare uma variável do tipo float e atribua um valor fracionado. Em seguida, declare uma variável do tipo long e atribua a ela o valor da variável float usando casting explícito. Por fim, declare uma variável do tipo double e atribua a ela o valor da variável long (casting implícito). Exiba todos os valores no console.

FUNDAMENTOS DE JAVA
EXERCÍCIOS

SENAI

Exercício 10: Diferença Entre Tipos de Dados

Declare uma variável do tipo short e atribua a ela um valor pequeno. Declare uma variável do tipo int e atribua a ela o valor da variável short (casting implícito). Depois, declare uma variável do tipo byte e atribua a ela o valor da variável short usando casting explícito. Exiba todos os valores no console.


SENAI
DEPARTAMENTO REGIONAL
DE SÃO PAULO
www.sp.senai.br