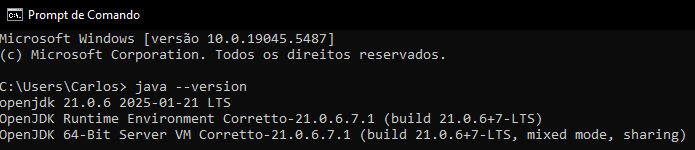
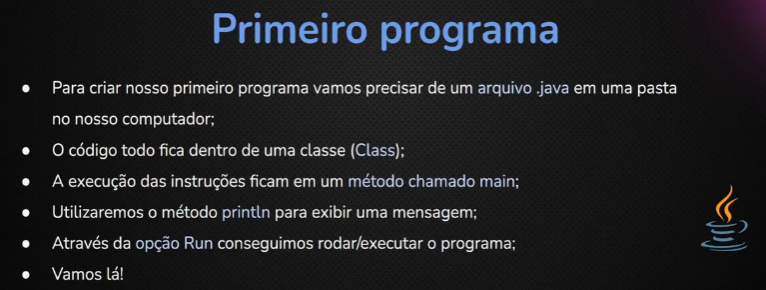


**Instalando o JDK**

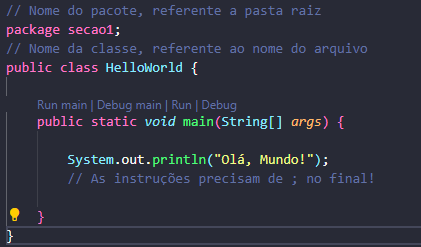


(comando para checar versão)

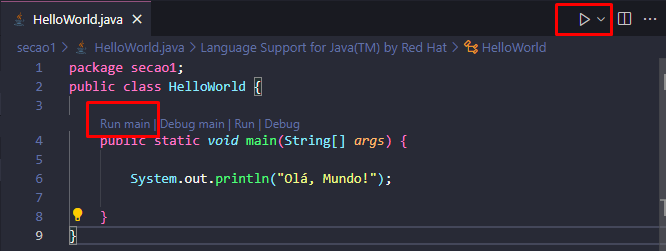


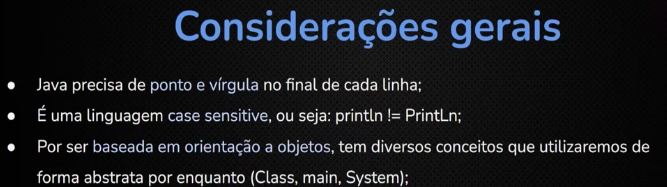
**Primeiro Programa**

A principio vamos ignorar as **class e** o método **main**, uma vez que estamos começando e estes são conceitos aplicados a orientação a objetosou **poo**

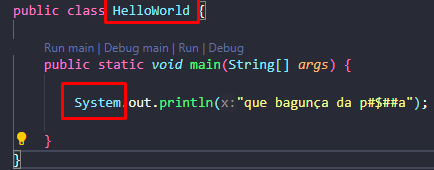


Como mencionado previamente as instruções serão mais detalhadas depois, estas são as formas de rodar o código (botão direito no arquivo e **run java** também funciona)

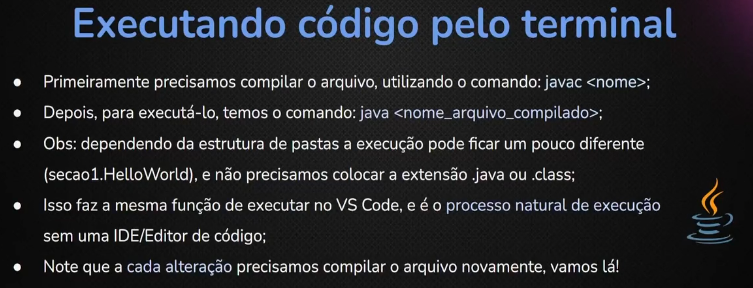




**Considerações Gerais**



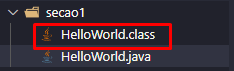
Note como a classe **System** possui a primeira letra maiúscula tal como o nome de nossa classe **HelloWorld**



**Executando código pelo terminal**



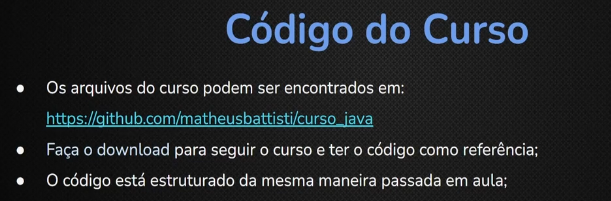
O comando gera um arquivo **.class** com o mesmo nome da classe compilada!



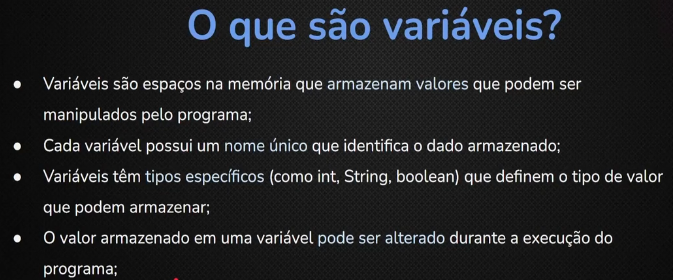
Com esse comando podemos rodar o arquivo .class, note que tivemos de subir um diretório acima (..)





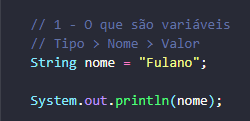


[Aqui](https://github.com/matheusbattisti/curso_java)

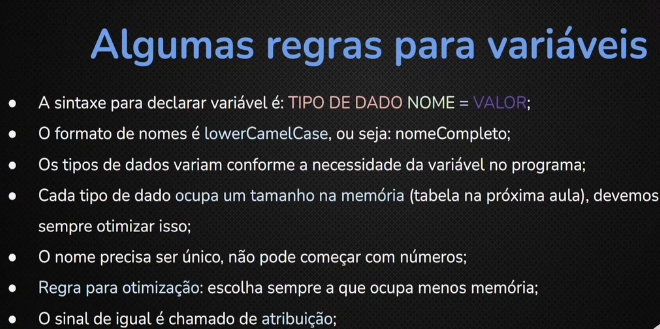




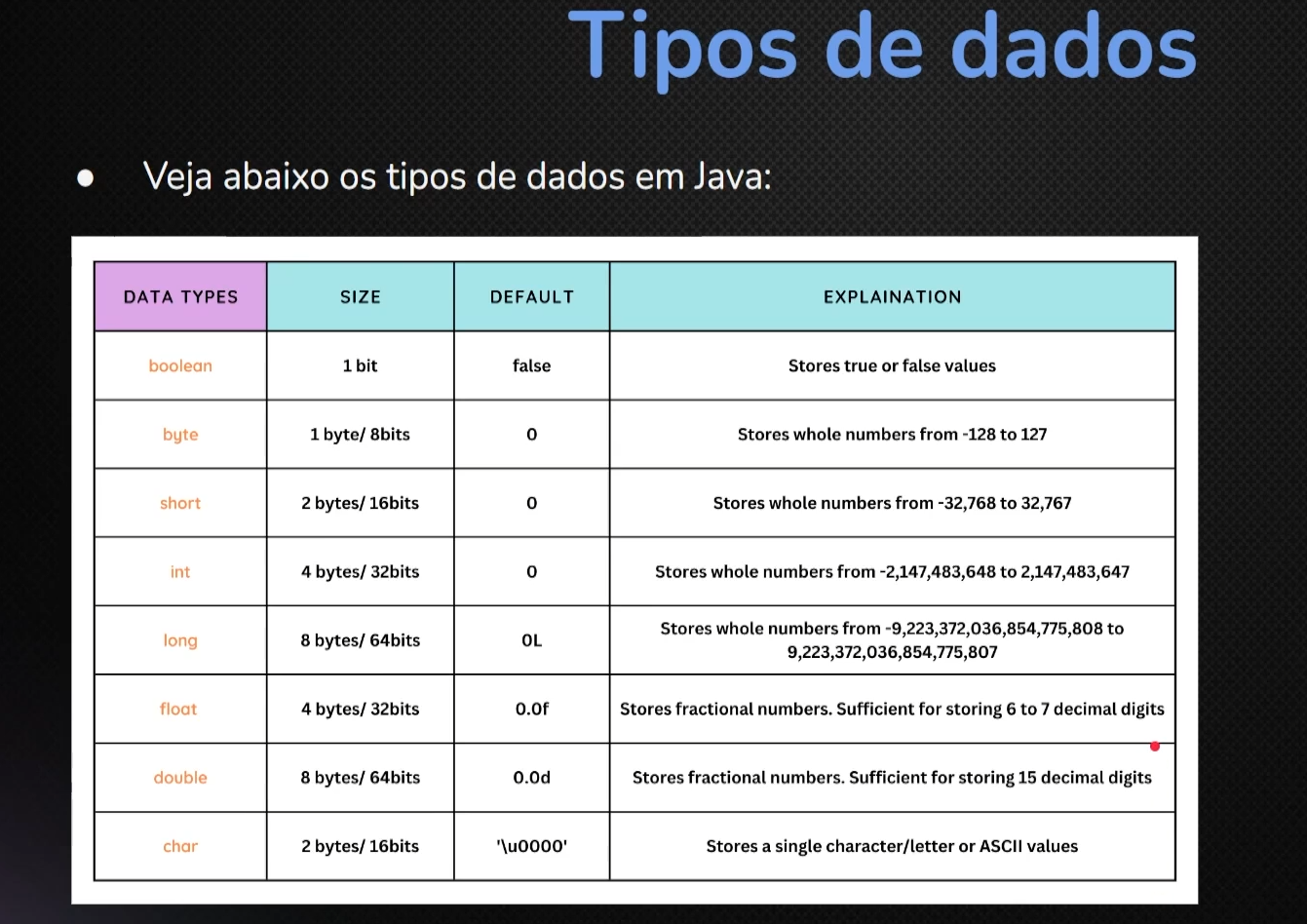
**Variáveis**





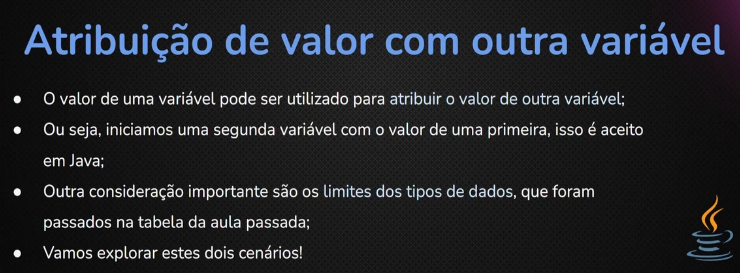


**Algumas regras para variáveis**

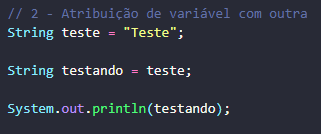


**Tipos de Dados (Primitivos)**

Esses são apenas os tipos primitivos, strings e arrays serão vistos depois!



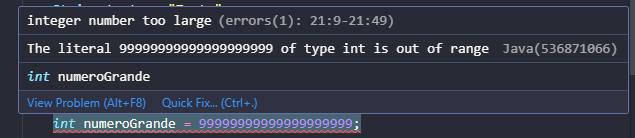
**Atribuição de valor com outra variável**





E

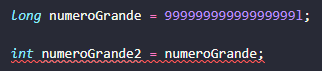




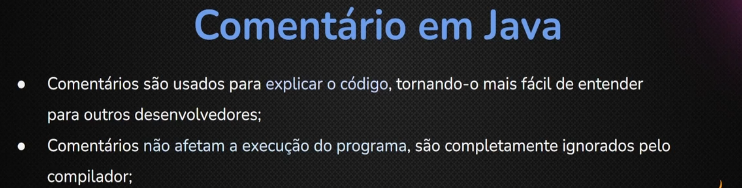
Esse valor ultrapassa o limite de **int** visto na tabela anteriormente

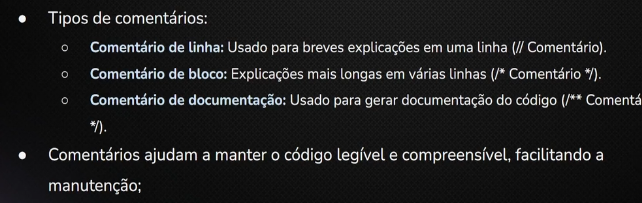


Long aceitaria esse tipo de número (com ajustes rs) note que precisamos por um **‘L’** no final do número!

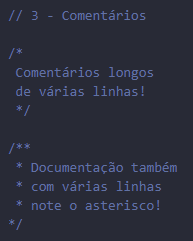


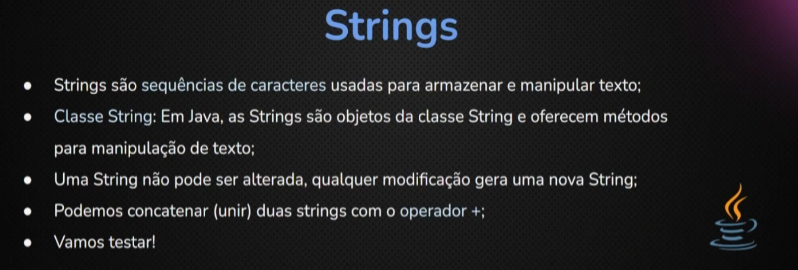
O Java não permite a atribuição de um variável a outra quando ela foge do escopo de suas funcionalidades, **int** não poderia ser capaz de imprimir um número tão grande!



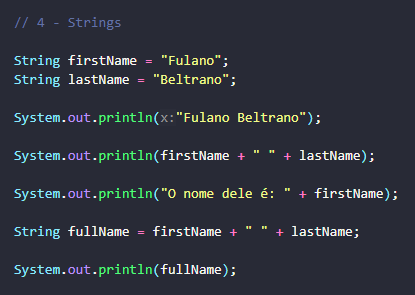


**Comentário em java**

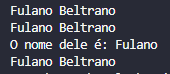


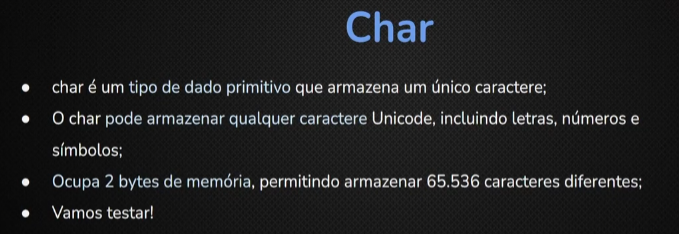


**Strings**

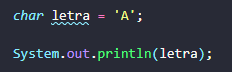


Perceba o uso de concatenação, usando + para unir strings vazias com outras strings ou variáveis!

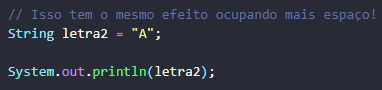




**Char**



Note que **char** usa aspas simples **‘’** ao contrário das aspas duplas **“”** que são reservadas para as strings no Java!

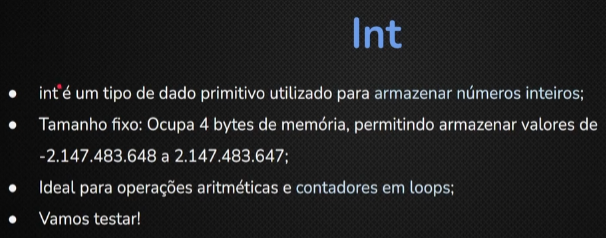


Lembre-se que é importante decidir com qual tipo de dados iremos trabalhar para tornar o código mais eficiente!

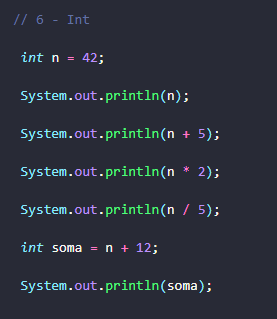


E é claro, apenas um único caractere pode ser exibido!

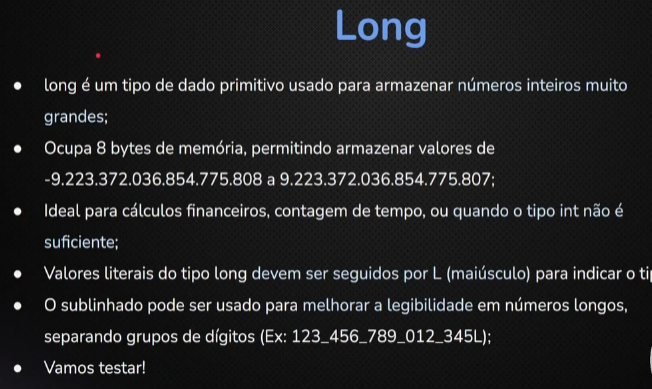




**Int**

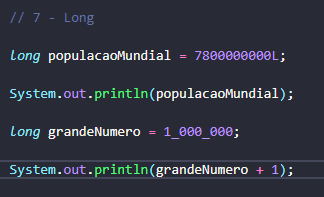


Note que a divisão resultou em 8 ao invés de 8,4, isso ocorre porque **int** não considera os números decimais!



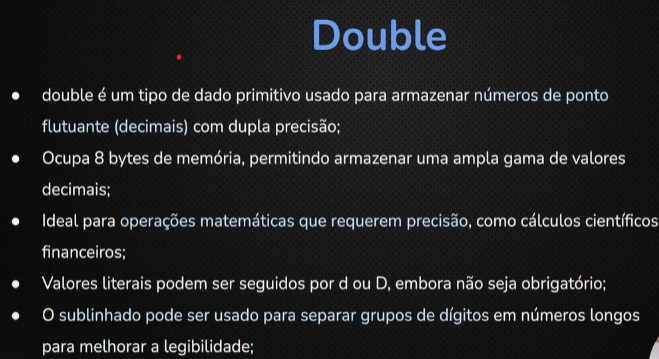
**Long**

Já vimos o long anteriormente na ‘’atribuição de valor a outra variável”

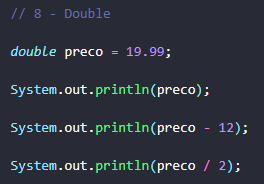


Note que os sublinhados não são impressos, mas ajudam os desenvolvedores a lerem grandes números!



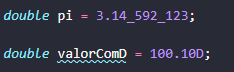


**Double**





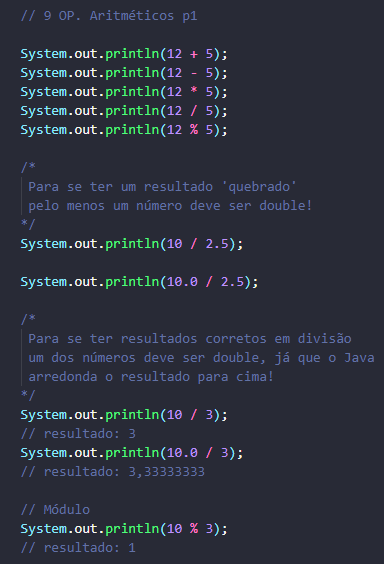
Com o **double** podemos trabalhar com números ‘quebrados’, note apenas que usando ponto . e não virgula , para separar as casas decimais

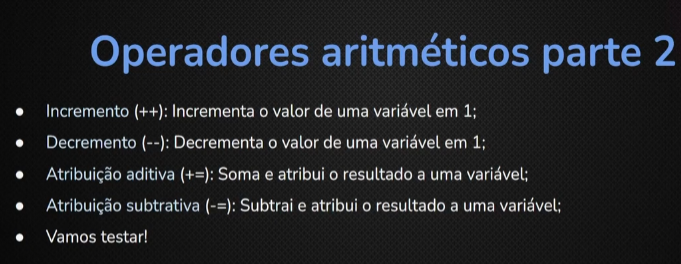


Perceba a regra opcional de \_ para separar grandes números e o D no final dos numerais!

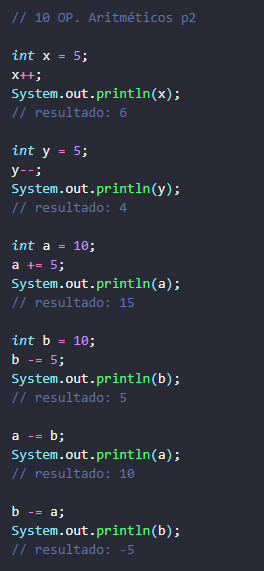


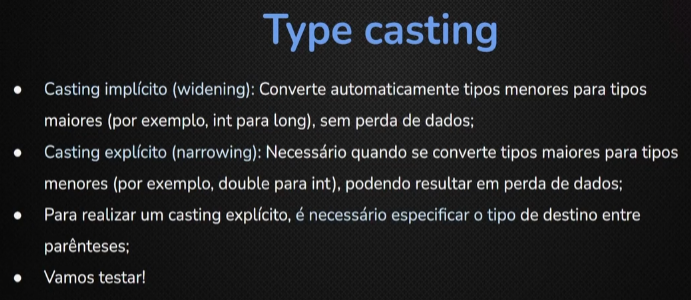
**Operadores Aritméticos pt1**

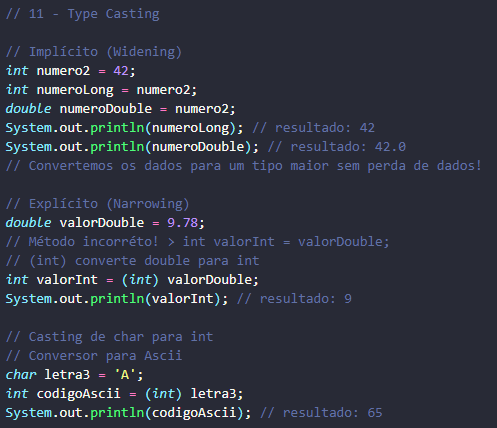




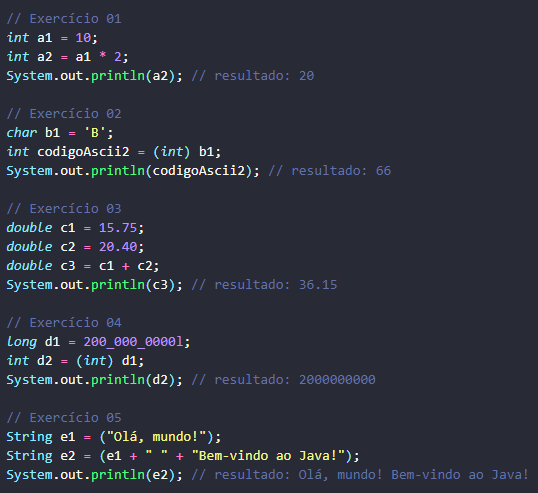
**Operadores Aritméticos pt2**

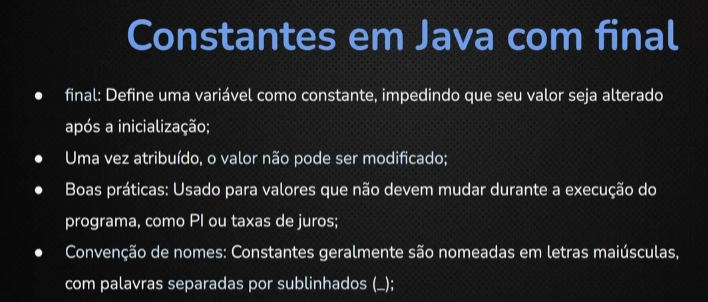


**Type Casting (**mudança de tipo de dado**)**

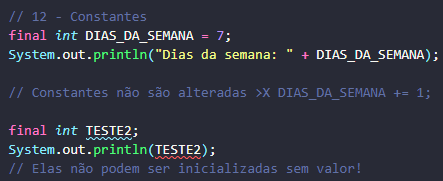


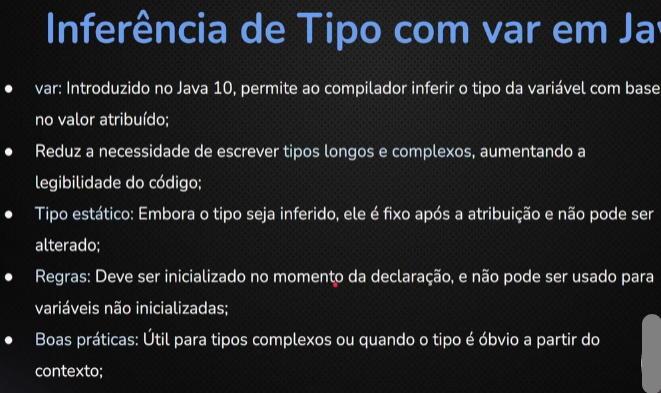
**Exercícios**



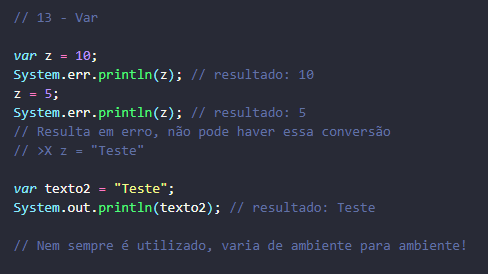


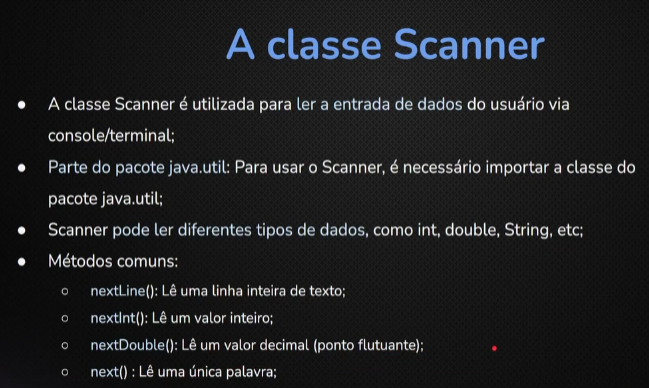
**Constantes em java com final**





**Inferência de tipo com var em java**

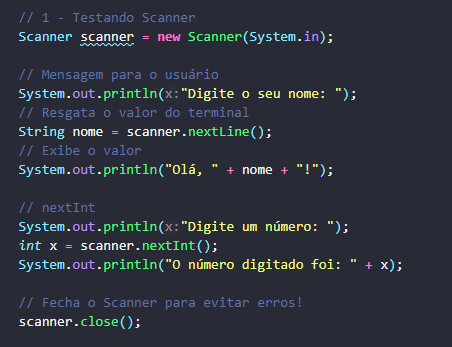




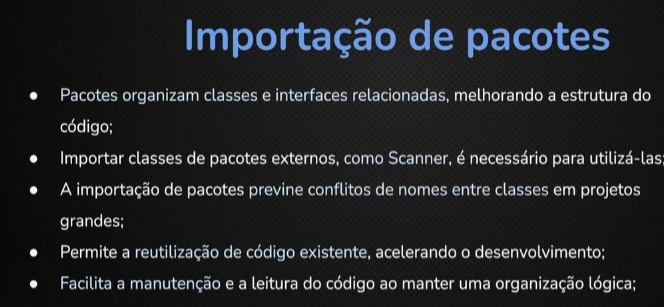
**A classe scanner**



Por vezes a importação é automática assim que você declara o uso de Scanner (Ao tentar importar Scanner sem usá-lo no código resultava no apagamento da linha de importação)

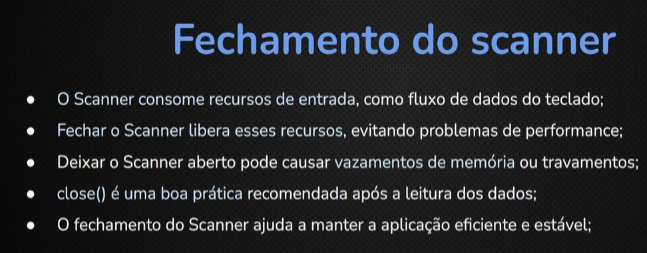


(Veremos o Scanner.close() com mais atenção depois!)



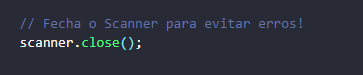
**Importação de pacotes**

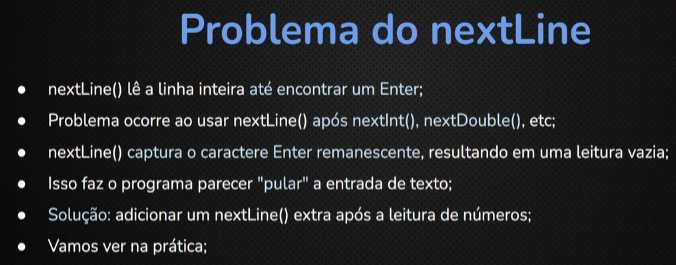
Veremos isso mais pra frente (tenha como referência o projeto de RPG, que continha várias funções de várias classes diferentes!



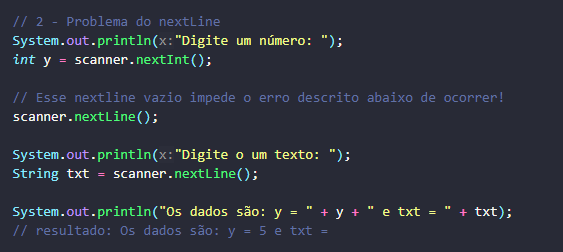
**Fechamento do scanner**

Coloque no final do código!





**Problema do nextLine**



scanner.nextLine() deve vir sempre após recebermos um número!