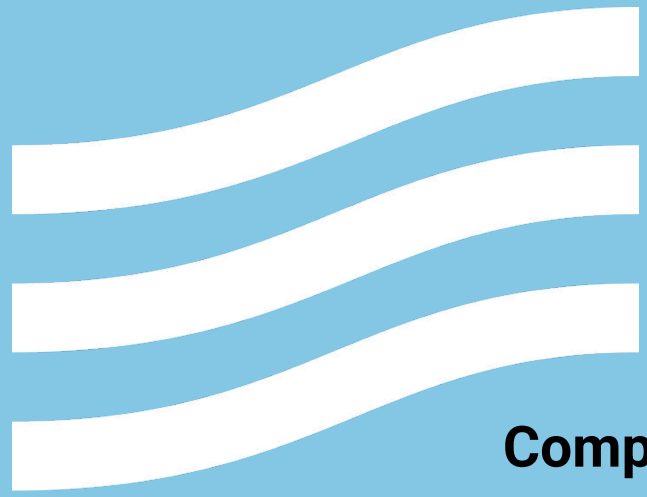


Introdução: Java
Compreender compilação, execução e entrada de dados

Firjan SENAI

- A linguagem Java foi criada em 1992 na Sun Microsystem.
- Em 2008 foi adquirida pela Oracle Corporation.

Características:

- Orientada a Objetos;
- Portabilidade;
- Segurança;
- Linguagem Simples;
- Alta Performance;
- Interpretada;
- Multiplataforma;
- Fortemente Tipada;

Java1: ano 1996 -primeira versão estável da linguagem Java foi o JDK (Java Development Kit) 1.0.2, em janeiro de 1996 com o codinome Oak.

Java2: ano 1998 – Neste versão houve um grande aumento das classes na biblioteca Java (API) entre outras características como: J2SE (Java 2 Standard Edition), J2EE (Java 2 Enterprise Edition) e J2ME (Java 2 Micro Edition).

Java3: ano 2000 – Incorporação do Corba. Inclusão das bibliotecas JNDI, JavaSound entre outros.

Java4: ano 2002 – Inclusao de suporte a IPV6, XML, imagens e outros recursos.

Java5: ano 2004 – Uma das versões mais utilizadas. Inserção de recursos como: Enumeradores, Autoboxing, Generics, for-each entre outros.

Java6: ano 2006 - A partir desta versão, as siglas J2SE, J2EE e J2ME foram substituídas pelas siglas **Java SE**, Java EE e Java ME respectivamente. Esta versão apresenta melhorias na parte de segurança e desempenho da máquina virtual.

Java7 ano 2011 – Algumas características importantes: permite o uso de strings em condições do switch, inferência na criação de objetos com tipos genéricos, uma biblioteca para tratar entrada e saída e melhorias nos streams para XML e Unicode.

Java8 ano 2014 – Melhoria na performance, manipulação de data e expressões como Lamba

Java9 ano 2017 - melhoria de desempenho às aplicações, jshell, api de suporte ao HTTP 2.0 entre outros.

Java10 ano 2018 - Inferência de tipos para variáveis locais, Garbage-Collector Interface entre outras melhorias.

Java11 ano 2018 – Anotações de tipo em expressões lambda, padronizaçao do cliente HTTP

Java12 ano 2019 – Novos métodos String, alterações de expressões no Switch, métodos transform entre outros.

Java 14 ano 2020 - Melhoria na utlização de alguns comandos como o instanceof, switch e outros.

Java 17 ano 2021 - Implementação de classes seladas, atualizações e melhorias na linguagem.

Java 18 ano 2022 - Melhorias de desempenho, estabilidade e segurança. UTF8 por padrão, Simple Web Server entre outros.

Java 21 ano 2023 - oferece milhares de melhorias de desempenho, estabilidade e segurança, incluindo aprimoramentos na plataforma que ajudarão os desenvolvedores a aumentar a produtividade e impulsionar a inovação e o crescimento em suas organizações.

Fonte: https://blogs.oracle.com/oracle-brasil/post/oracle-lanca-java-21

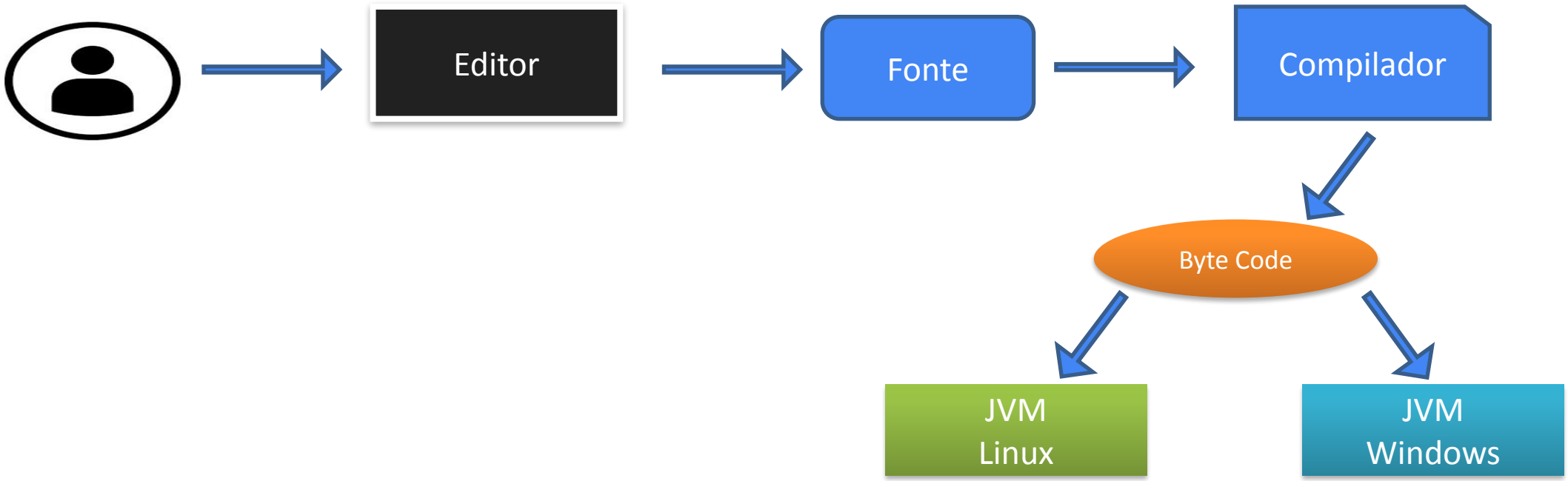
É uma edição ou versão que terá um período prolongado de suporte no caso do Java a cada três anos, recebendo apenas atualizações trimestrais de segurança, estabilidade e desempenho.

As versões LTS atuais do Java são Java 7, Java 8, Java 11,17 e 21.

A Oracle pretende oferecer suporte às versões Java LTS da seguinte forma:

- Java 8 até pelo menos 2030
- Java 11 foi estendido até pelo menos janeiro 2032
- Java 17 até pelo menos 2029
- Java 21 até pelo menos 2031

Um dos recursos do Java é a portabilidade do código gerado. Esta portabilidade é atingida através da utilização de bytecodes. Bytecode é um formato de código intermediário entre o código fonte, o texto que o programador consegue manipular, e o código de máquina, que o computador consegue executar. Na plataforma Java, o bytecode é interpretado por uma máquina virtual Java (JVM). A portabilidade do código Java é obtida à medida que máquinas virtuais Java estão disponíveis para diferentes plataformas. Assim, o código Java que foi compilado em uma máquina pode ser executado em qualquer máquina virtual Java, independentemente de qual seja o sistema operacional ou o processador que executa o código:



Java SE (Standard Edition)

JDK: Java Developer's Kit, conjunto de ferramentas para desenvolvimento;
JRE: Java Runtime Environment, ambiente de interpretação e execução.

Java Open JDK

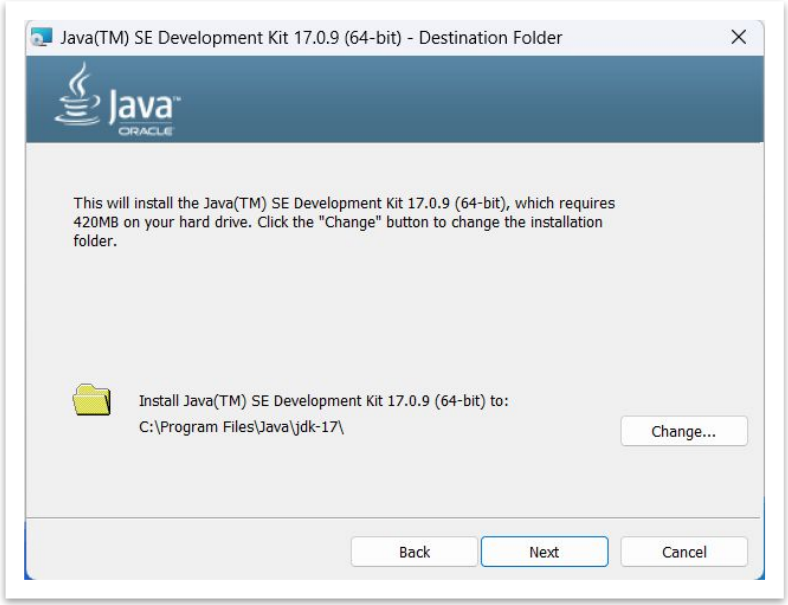
O Java OPEN JDK é a versão free, no entanto, é preciso fazer atualizações sempre que uma nova versão for lançada. Caso não sejam feitas as atualizações, não serão mais feitas correções de bugs e nem instaladas novas funcionalidades que forem lançadas no programa.

Algumas ferramentas do Java JDK

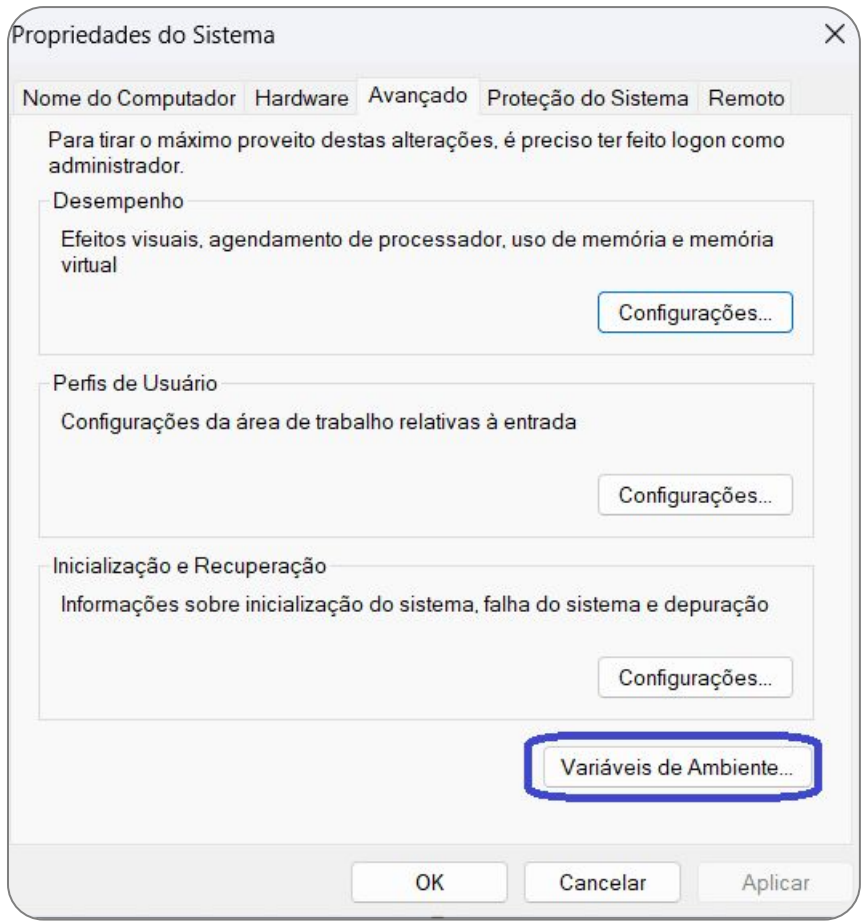
- Compilador Java - javac
- Interpretador de aplicações Java - java
- Um gerador de documentação para programas - javadoc
- O manipulador de arquivos comprimidos - jar

Fazer o download e instalação

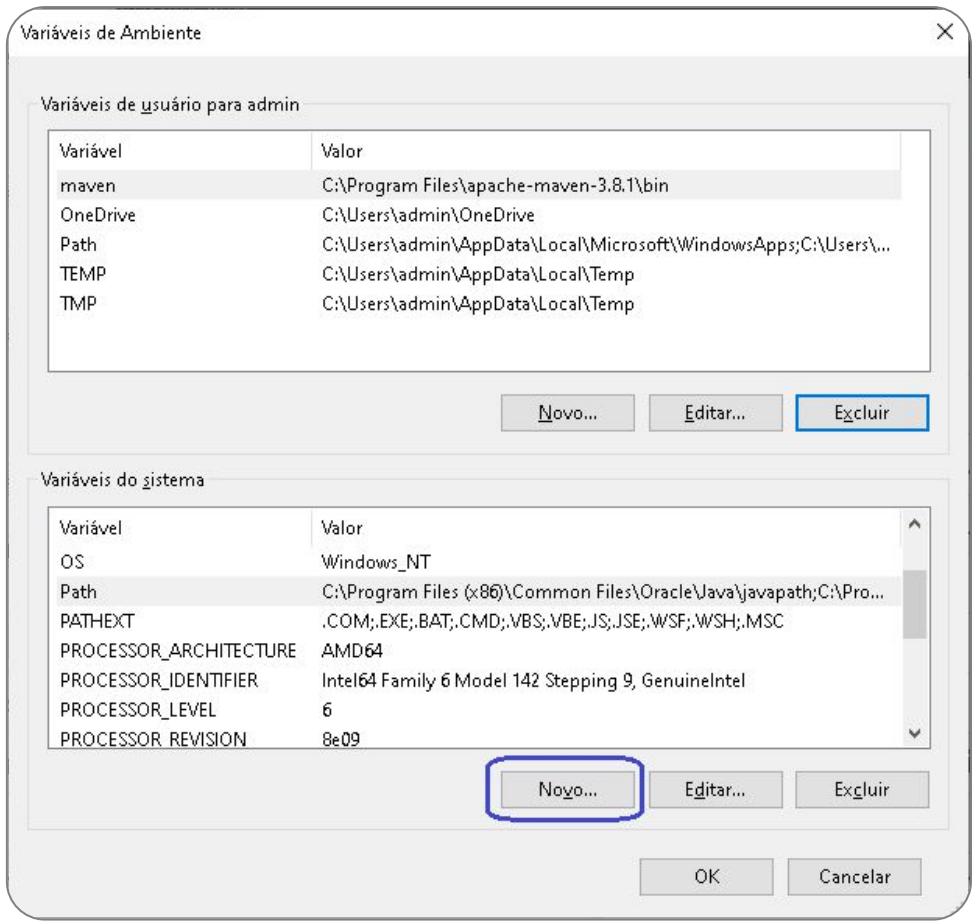
https://www.oracle.com/java/technologies/javase/jdk17-archive-downloads.html



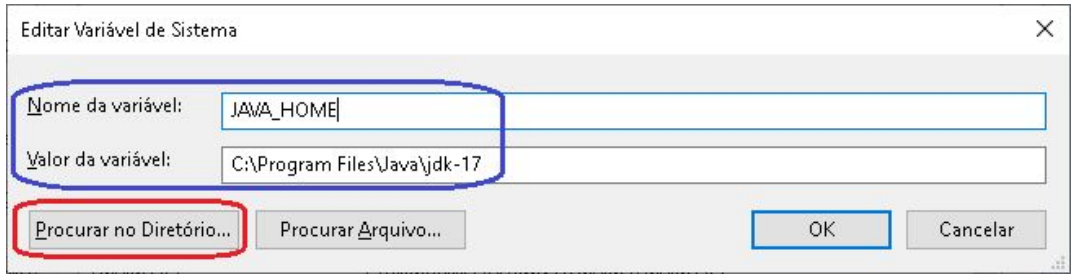
Na pesquisa do Windows digite variáveis e abra a propriedade de sistema

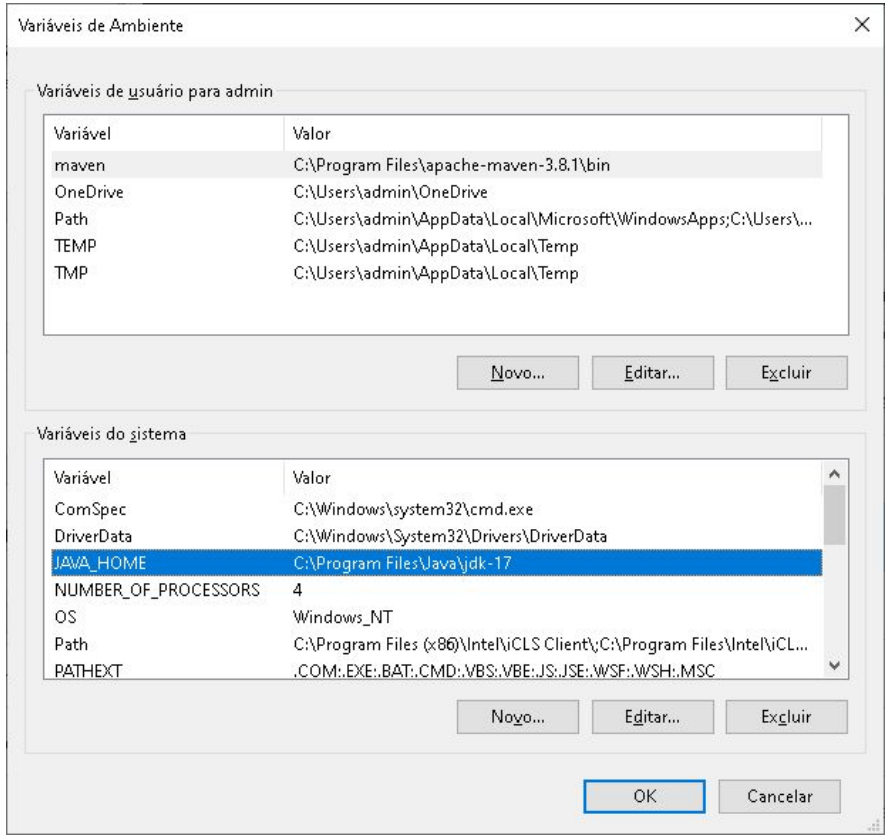


A variável pode ser utilizada para um determinado usuário ou para todos os usuários do sistema. Clique em Novo nas variáveis do sistema



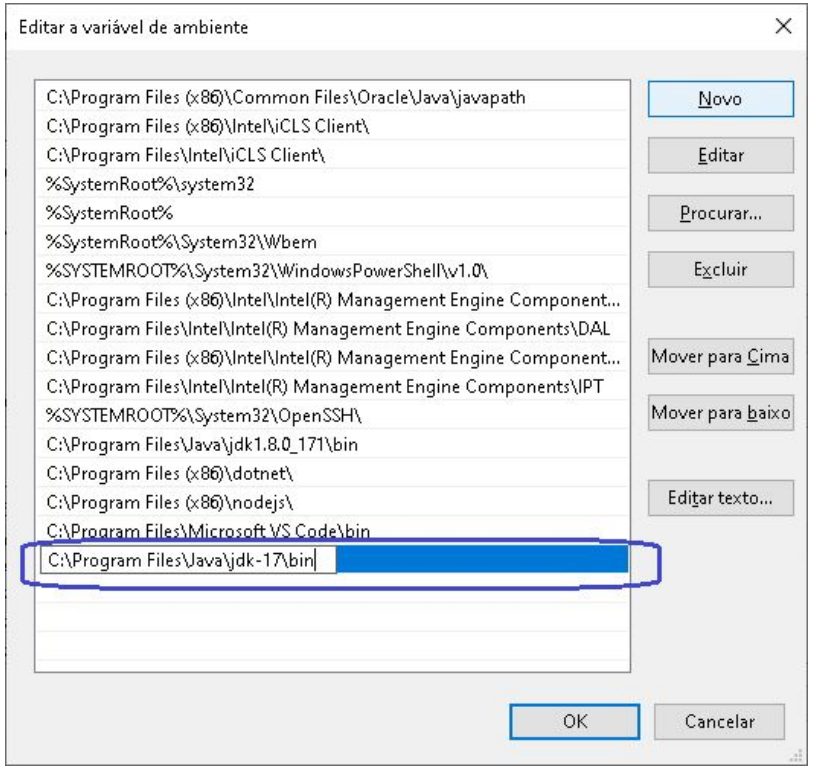
Digitar o nome da variável e procure o diretório de instalação do Java. Finalize clicando em OK





O Path é uma variável de ambiente de um sistema operacional que fornece a uma aplicação uma lista de pastas onde procurar por arquivos executáveis. Na imagem abaixo é exibida a configuração do Path do Java no Windows.

Para configuar, selecione a variável de sistema Path clique em Editar, adicione a pasta **bin** do Java para que o sistema operacional encontre os executáveis do Java. Clique em OK



Verificando a configuração

```
C:\Users\roni_>java --version
java 17.0.7 2023-04-18 LTS
Java(TM) SE Runtime Environment (build 17.0.7+8-LTS-224)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.7+8-LTS-224, mixed mode, sharing)
```

Uma aplicação Java deve ter pelo menos uma classe que contenha um método chamado main(), o qual contém o primeiro código a ser executado para iniciar a aplicação.

Usando um editor de texto inserimos o código e salvamos o arquivo com o nome **Exemplo.java** Conforme exemplo abaixo:

O nome do arquivo deve coincidir com o nome da classe

```
public class Exemplo {
  public static void main(String[] args){
     System.out.println("Olá Mundo");
  }
}
```

Acesse o terminal do Windows ou Linux e execute os comandos abaixo:
O compilador converte arquivos-fonte Java em bytecodes com o comando **javac**

javac Exemplo.java

Como resultado teremos um arquivo bytecode com o mesmo nome do arquivo mas com a extensão **.class**:
O interpretador Java é chamado com o aplicativo java.exe. Ele é usado para interpretar o bytecode arquivo .class

Para execução basta digitar.

java Exemplo

O método main() é a primeira função que será executada no programa. Ele é public o que quer dizer que ele é visível globalmente, void porque não tem retorno, static o que significa que não precisamos criar objetos e também recebe um array de objetos do tipo String.

Quem chama o método main é o inicializador quando interpretamos o bytecode. O único argumento do método main serve para armazenar em cada entrada do array os parâmetros digitados pelo usuário após o nome da classe a ser interpretada.

Vamos alterar nossa classe Exemplo conforme abaixo e compilar e executar passando argumentos

```
public class Exemplo {
  public static void main(String[] args){
     System.out.println("Olá Mundo");
     System.out.println(args[0]);
     System.out.println(args[1]);
     System.out.println(args[2]);
  }
}
```

javac Exemplo.java
java Exemplo Celular TV Geladeira

```
c:\teste>java Exemplo Celular TV Geladeira
Olá Mundo !!
Celular
TV
Geladeira
```

Podemos criar uma variável de ambiente no sistema operacional com o comando:

set Linguagem='Java’

e usar o comando **set** para exibir as variáveis de ambiente do sistema operacional. A variável **USERNAME** guarda o usuário logado no Windows.

```
public class Teste {

	public static void main(String[] args) {
		System.out.println(args[0]);
		System.out.println(args[1]);
	}
}
```

1) Criar uma Exemplo2.java (uma Classe) no vscode com o nome **Exemplo2**. Imprima seu nome em uma linha e sobrenome em outra linha usando o comando “System.out.print()”. Sabendo que os caracteres \n representam quebra de linhas.

2) Utilize os caracteres abaixo no **Exemplo2** no lugar do \n para ver o resultado:

\t
\b
\\
\’
\”

É um ambiente de desenvolvimento integrado, combinando ferramentas, recursos que facilitam o desenvolvimento de aplicações.

Eclipse

- Multiplataforma
- Suporte a Plugins
- Pacotes de desenvolvimento para Java Web e Desktop
- Muito utilizada no mercado

NetBeans

- Possui suporte para criação de interfaces para aplicações web, desktop e mobile.
- Multiplataforma
- Muito utilizada em instituições de ensino

IntelliJ

- Multiplataforma
- Possui um ótimo assistente de código
- Suporte nativo ao Kotlin
- Uso de plugins: É possível desenvolver em diferentes tecnologias com o IntelliJ (Python, Dart, etc) com o uso de plugins;

VsCode

- Simples
- Diversas extensões disponíveis