



# Introdução a Linguagem Java

---

Prof<sup>a</sup>. Rachel Reis  
rachel.@inf.ufpr.br



# Linguagem Java

---

- Origem

- Linguagem desenvolvida pela *Sun Microsystems*
- Sintaxe similar ao C++

- Principais características

- Orientada a objetos
- Gerência automática de memória
- Portabilidade





# Plataforma Java

---

- Plataforma
  - Ambiente de *hardware* e/ou *software* no qual um programa é executado
- A plataforma Java é somente de *software*
  - Esta plataforma de execução funciona sobre outras plataformas de *hardware* e *software*.

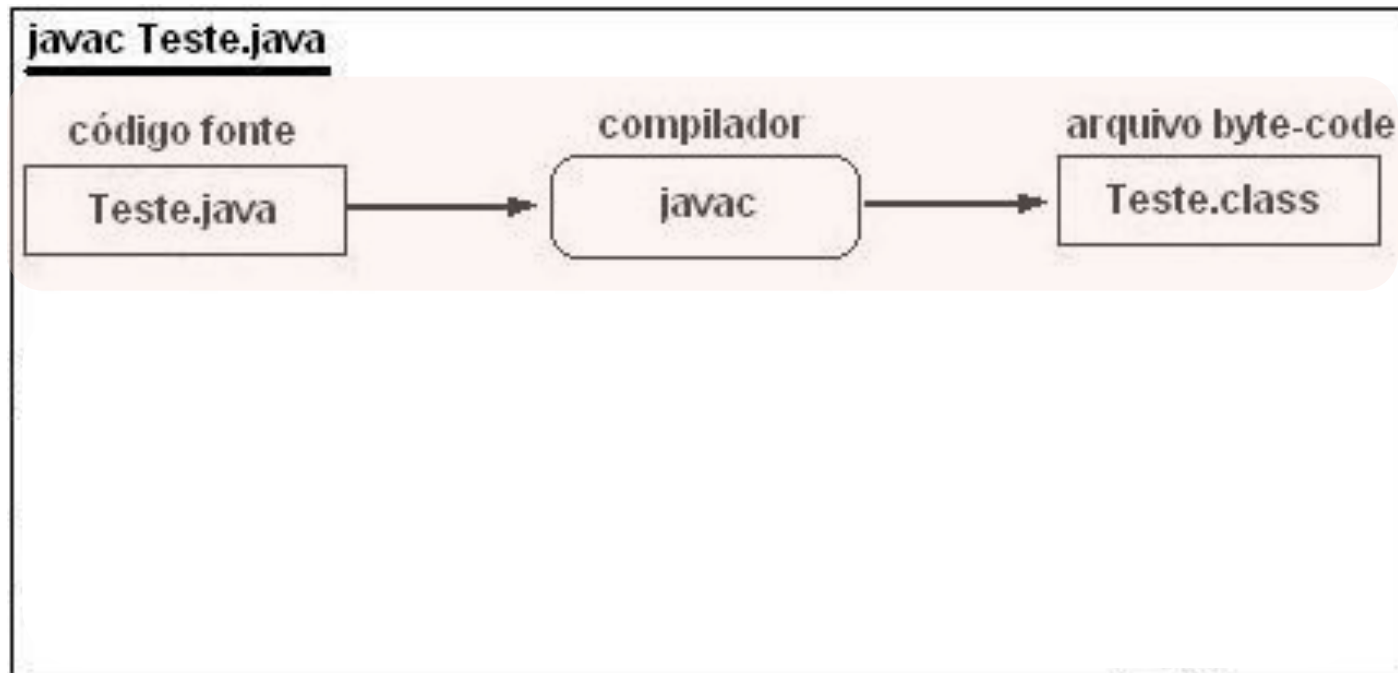


# Plataforma Java

---

- Composta por dois ambientes:
  - Ambiente de desenvolvimento – **JDK** (Java Development Kit)
  - Ambiente de execução – **JRE** (*Java Runtime Environment*)

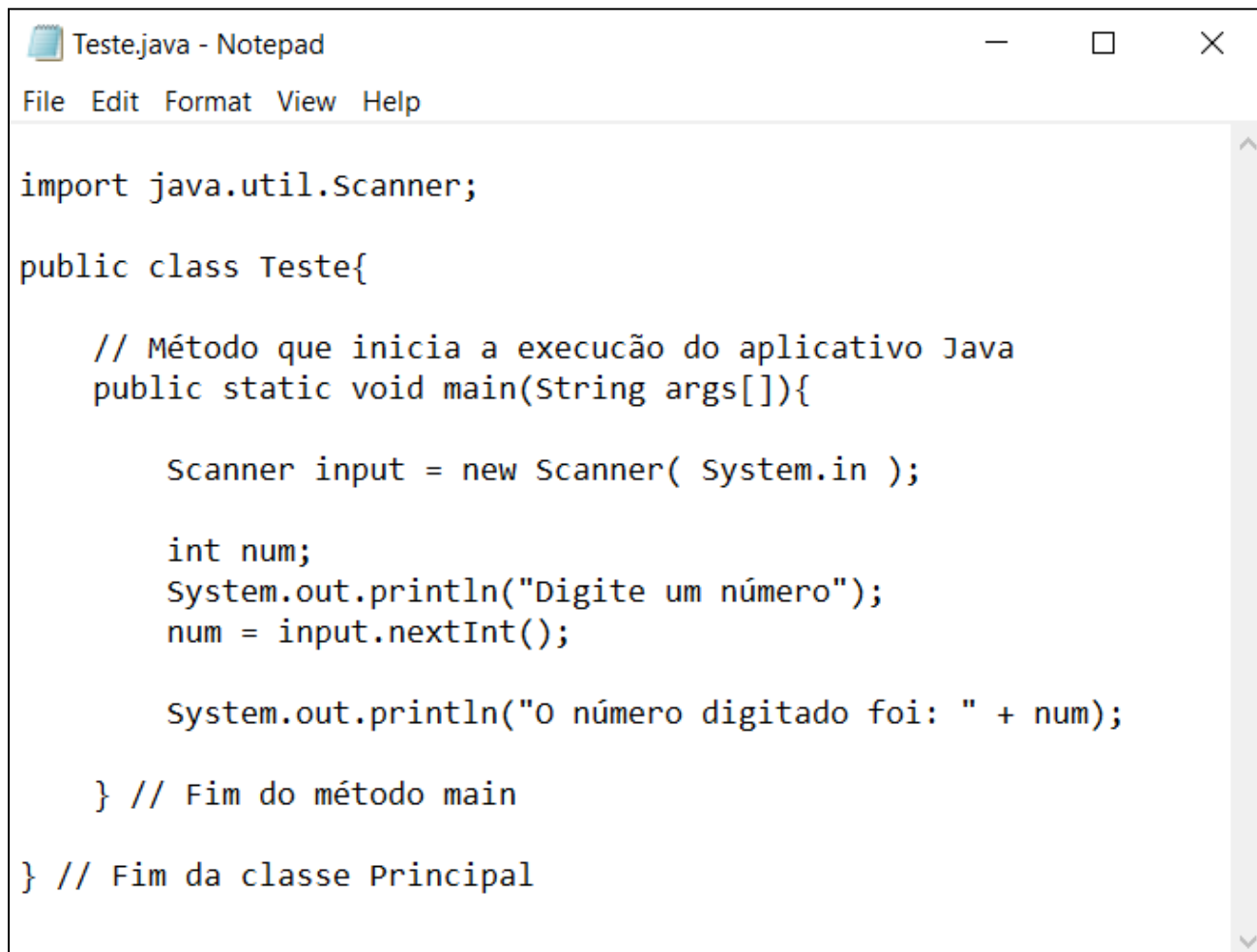
# Ambiente de Desenvolvimento - JDK





# Exemplo de código fonte

## Teste.java



```
Teste.java - Notepad
File Edit Format View Help

import java.util.Scanner;

public class Teste{

    // Método que inicia a execução do aplicativo Java
    public static void main(String args[]){

        Scanner input = new Scanner( System.in );

        int num;
        System.out.println("Digite um número");
        num = input.nextInt();

        System.out.println("O número digitado foi: " + num);

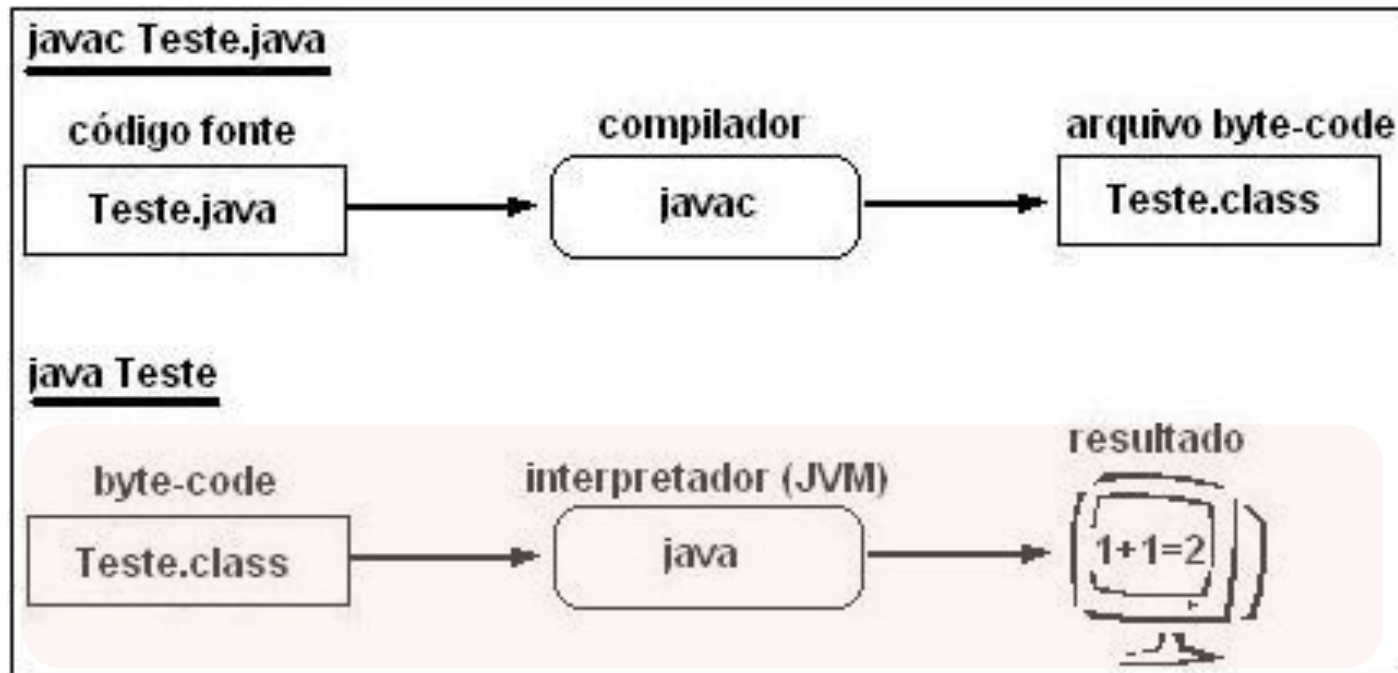
    } // Fim do método main
} // Fim da classe Principal
```

```

Teste.class - Notepad
File Edit Format View Help
Ëpº% 7 F
[] []
! "" #
$ %
& * + , <init> ()V []Code []LineNumberTable []LocalVariableTable []this []LTeste;[]
main []([Ljava/lang/String;)V []args [][Ljava/lang/String;[] input []Ljava/util/Scanner;[] num[]
[]I[]
SourceFile []Teste.java^ ^
[] []java/util/Scanner -^ . /^^ 0^ 1 2 []Digite um nÃºmero 3^ 4 5^ 6 7 []BootstrapMethods[]
8[] 9^ : ;[] Teste[] []java/lang/Object []java/lang/System in[] []Ljava/io/InputStream;[] []
(Ljava/io/InputStream;)V []out []Ljava/io/PrintStream;[] []java/io/PrintStream []println[] []
(Ljava/lang/String;)V []nextInt[] ()I
< =[] []O nÃºmero digitado foi: [] []makeConcatWithConstants [](I)Ljava/lang/String;[] >^ : B[]
$java/lang/invoke/StringConcatFactory[] D[] []Lookup[] ^InnerClasses[] ~
(Ljava/lang/invoke/MethodHandles
$Lookup;Ljava/lang/String;Ljava/lang/invoke/MethodType;Ljava/lang/String;
[Ljava/lang/Object;)]Ljava/lang/invoke/CallSite;[] E[] %java/lang/invoke/MethodHandles
$Lookup[] java/lang/invoke/MethodHandles !
[] [] ^
[] [] / [] [] []*. []± [] [] [] [] ^ [] [] [] [] s [] [] %» Y² []· []L² [][]
[]+[] []=² []º []± [] [] [] [] ^ []
[] [] $ [] [] [] % [] [] [] [] [] [] []
[] [] [] [] A
[] ? C @ [] ' [] [] ( [] )

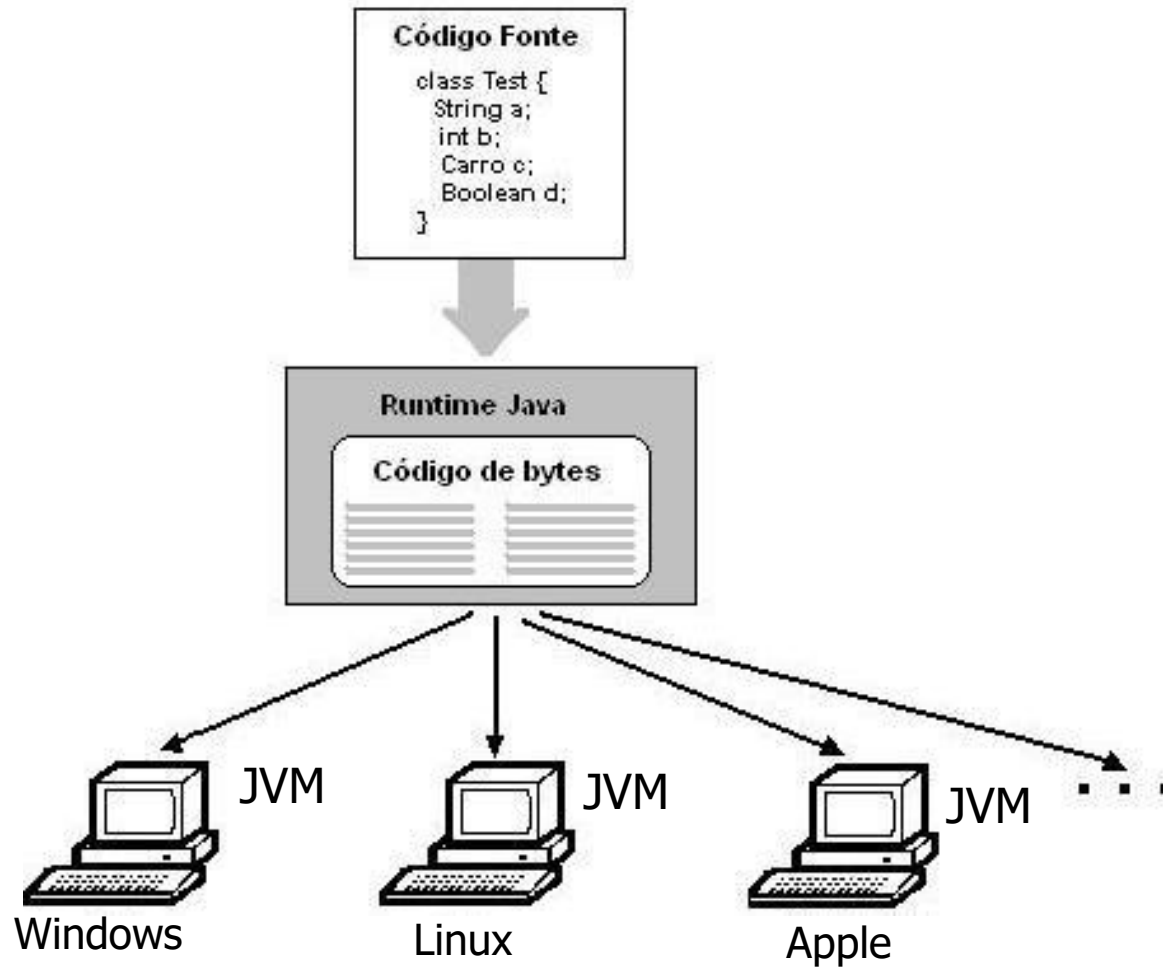
```

# Ambiente de Execução - JRE





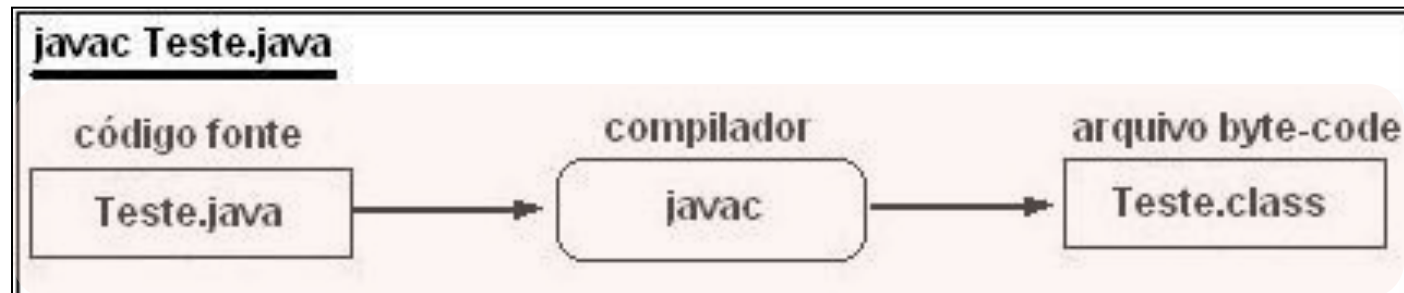
# Ambientes Desenvolvimento e Execução



# Resumo JDK, JRE, JVM

- **JDK** (ambiente de desenvolvimento java):
  - é necessário para desenvolver softwares Java.
  - Inclui a **JRE** e ferramentas de programação, como:
    - **javac** - compilador
    - **jar** - empacotador
    - **javadoc** - ferramenta para geração de documentação

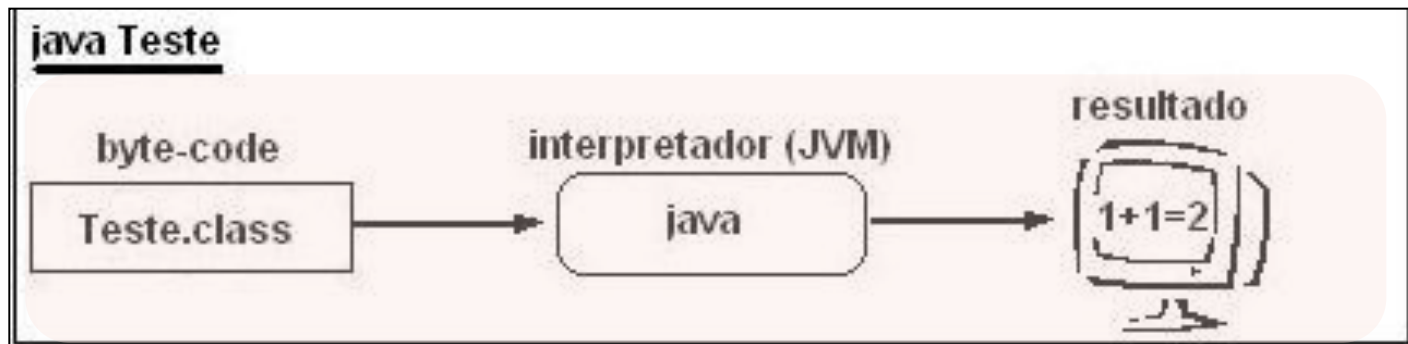
Exemplo:



# Resumo JVM, JRE, JDK

- **JRE** (ambiente de execução java):
  - Consiste na **JVM** e na Java **API**, que são as bibliotecas de classes presentes em todos os ambientes de produção Java
- **JVM** (*Java Virtual Machine*):
  - Máquina virtual que interpreta e executa o código Java compilado (*bytecode*).

Exemplo:



# Plataformas de Desenvolvimento

- Java conta com três plataformas de desenvolvimento conhecidas:

- **JSE,**
- **JEE,**
- **JME.**



- Cada um dessas plataformas possui suas bibliotecas (APIs) específicas.



# Plataformas Java

## JSE

Soluções para  
escritório

- Kit básico
- Aplicações independentes
- Aplicações desktop, notebook

## JEE

Soluções para  
empresas

- Aplicações empresariais
- Comércio eletrônico
- Aplicações web

## JME

Soluções para  
dispositivos móveis

- Aplicações móveis ou portáteis
- Celulares, palmtops, pagers



# Programa Principal

---

- Todo programa Java consiste de pelo menos uma declaração de classe definida pelo programador.

```
public class BemVindo{  
    . . .  
}
```

- A palavra *class* introduz uma classe e deve ser seguida pelo nome da classe.



# Programa Principal

---

- Regras para nomes de classes
  - Primeiro caractere: letra, underscore ( \_ ), cifrão (\$)
  - Após o primeiro caractere: qualquer letra ou número
  - Não é permitido espaço em branco e operadores
  - Não é permitido palavras reservadas
- Convenção de código:
  - Nomes de classe iniciam com uma letra maiúscula e apresentam a letra inicial de cada palavra que eles incluem em maiúscula (por exemplo, SampleClassName).



# Programa Principal

---

- Método main(): ponto de partida de um aplicativo Java.

```
public class BemVindo{  
    // Método principal  
    public static void main(String[] args) {  
  
        ...  
    }  
}
```





# Instrução de Saída

---

- System.out: objeto de saída padrão.

```
public class BemVindo
{
    public static void main(String args[])
    {
        System.out.println("Oi! Seja bem vindo!");
    }
}
```



# Instruções de Saída

---

- Qual a diferença entre as instruções de saída abaixo?
  - `System.out.print(...);`
  - `System.out.println(...);`
  - `System.out.printf(...);`



# IDE

---

- Existem vários ambientes integrados de desenvolvimento para Java

- **Replit**
- NetBeans
- Eclipse
- BlueJ
- VS Code





# Comandos javac e java

---

- Comando incluídos no JDK
  - javac
    - **compila** o código Java sem o uso de IDE
  - java
    - **executa** o código Java sem o uso de IDE



# Prompt de Comandos

- Abra o "Prompt de comandos" e vá para o diretório em que se encontra o arquivo BemVindo.java

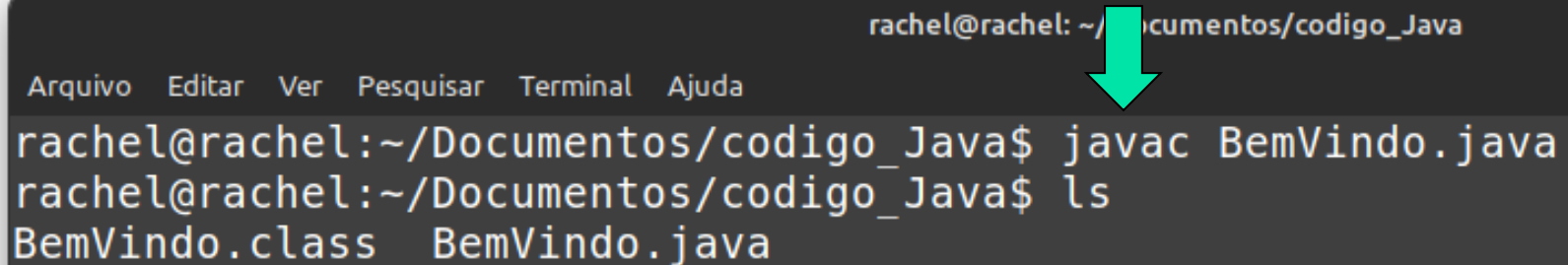
```
rachel@rachel: ~/Documentos/codigo_Java
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
rachel@rachel:~/Documentos/codigo_Java$ ls
BemVindo.java
↑
```



# Prompt de Comandos

---

- Para compilar o programa digite:  
`javac BemVindo.java` (será criado o .class)



```
rachel@rachel: ~/Documentos/codigo_Java
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
rachel@rachel:~/Documentos/codigo_Java$ javac BemVindo.java
rachel@rachel:~/Documentos/codigo_Java$ ls
BemVindo.class  BemVindo.java
```



# Prompt de Comandos

---

- Para executar o programa digite:  
**java BemVindo**

```
rachel@rachel: ~/Documentos/codigo_Java
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
rachel@rachel:~/Documentos/codigo_Java$ java BemVindo
Oi! Seja bem vindo!
```





# Tipos de comentário em Java

---

- Comentários em Java

- Bloco:

- ```
/*
```

- Comentários com mais de uma linha**

- Comentários com mais de uma linha**

- ```
*/
```

- Linha:

- ```
//
```

**Comentário de uma linha**





# Exemplo 1

---

- Escreva um programa m Java para ler um número inteiro digitado pelo usuário. Em seguida, exiba o número na tela.

- Exemplo:

- Entrada

Digite um numero: 45

- Saída

Numero digitado: 45

```
import java.util.Scanner;

public class Principal{

    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in) ;
        int num;
        System.out.print("Digite um numero: ");

        num = input.nextInt() ;

        System.out.println("Numero digitado: " + num) ;

        input.close() ;
    }
}
```

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

```
/*
```

- O **import** é usado para o compilador localizar uma classe.
- No exemplo acima, o comando faz o import da classe Scanner para que ela seja usada no programa.

```
*/
```

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

```
public class Principal{
```

```
    public static void main(String[] args)
    {
```

```
        /*
```

A classe Principal contém o método main() que é responsável por iniciar o programa Java.

```
        */
```

```
    }
```

```
}
```

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

```
public class Principal{
```

```
    public static void main(String[] args)
    {
```

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

```
        /*
```

A declaração acima **cria o objeto input** para ler os dados fornecidos pelo usuário.

```
        */
```

```
    }
```

```
}
```

```
import java.util.Scanner;

public class Principal{

    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in) ;
        int num;

        /*
            A variável num será usada para armazenar um
            número inteiro.
        */

    }
}
```

```
import java.util.Scanner;

public class Principal{

    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in) ;
        int num;
        System.out.print("Digite um numero: ");

        /*
            A instrução acima exibe uma mensagem na tela.
        */

    }
}
```

```
import java.util.Scanner;

public class Principal{

    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in) ;
        int num;
        System.out.print("Digite um numero: ");

        num = input.nextInt() ;

        /*
        O método nextInt() lê o primeiro número
        digitado pelo usuário e atribui a variável num.
        */
    }
```



```
import java.util.Scanner;

public class Principal{

    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in) ;
        int num;
        System.out.print("Digite um numero: ");

        num = input.nextInt() ;

        System.out.println("Numero digitado: " + num) ;

        /*
            A instrução acima exibe uma mensagem na tela.
        */
    }
```

```
import java.util.Scanner;

public class Principal{

    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in) ;
        int num;
        System.out.print("Digite um numero: ");

        num = input.nextInt();

        System.out.println("Numero digitado: " + num);

        input.close();
    }
}
```

/\*

O método **close()** fecha o  
objeto input.

\*/



# Tipos de Dados em Java

---

- Tipo lógico: *boolean*
- Tipo textual: *char* e *String*
- Tipo ponto flutuante: *float* ou *double*
- Tipo inteiro: *byte*, *short*, *int* e *long*

|         |              |                            |
|---------|--------------|----------------------------|
| 8 bits  | <i>byte</i>  | $-2^7 \dots 2^7 - 1$       |
| 16 bits | <i>short</i> | $-2^{15} \dots 2^{15} - 1$ |
| 32 bits | <i>int</i>   | $-2^{31} \dots 2^{31} - 1$ |
| 64 bits | <i>long</i>  | $-2^{63} \dots 2^{63} - 1$ |



# Pesquisar

---

- Instrução para ler um número inteiro (usando Scanner):

```
int num;
```

```
num = input.nextInt() ;
```

- Pesquise as instruções para ler:
  - Um dado do tipo float
  - Um dado do tipo char
  - Um dado do tipo String



## Exemplo 2

---

- Escreva um programa em Java que leia dois números digitados pelo usuário. Em seguida, imprima os números em ordem crescente.

- Exemplo:

- Entrada

Digite um numero: 10

Digite outro numero: 5

- Saída

5 10

```
import java.util.Scanner;

public class Principal{

    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        int num1, num2;

        System.out.print("Digite um numero: ");
        num1 = teclado.nextInt();
        System.out.print("Digite outro numero: ");
        num2 = teclado.nextInt();
        if(num1 < num2){
            System.out.println(num1 + " " + num2);
        }
        else{
            System.out.println(num2 + " " + num1);
        }
        teclado.close();
    }
}
```



## Exemplo 3

---

- Escreva um programa em Java que imprima a tabuada de multiplicação de um número inteiro no intervalo [1,10].

- Exemplo:

- Entrada

Digite um numero: 7

- Saída

$$7 * 1 = 7$$

$$7 * 2 = 14$$

...

$$7 * 10 = 70$$

```
import java.util.Scanner;

public class Principal{

    public static void main(String[] args){
        Scanner teclado = new Scanner(System.in);
        int num, cont, mult;

        System.out.print("Digite um numero: ");
        num = teclado.nextInt();

        cont = 1;
        while(cont <= 10){
            mult = num * cont;
            System.out.println(num+ "*" +cont+"="+mult);
            cont++;
        }
        teclado.close();
    }
}
```





## While → For

---

- Reescreva o código abaixo usando a estrutura de repetição for.

```
cont = 1;
while(cont <= 10) {
    mult = num * cont;
    System.out.println(num+"*"+cont+"="+mult) ;
    cont++;
}
```



# Variáveis final

---

- A palavra-chave final serve para especificar que uma variável não é modificável e que qualquer tentativa de modificar é um erro.

```
public class Principal{  
    public static void main(String args[]){  
        final String msg = "Bem vindo!";  
    }  
}
```



# Variáveis final

---

- O conteúdo de uma variável final pode ser atribuído depois de sua criação.
- Importante: só é possível atribuir valor a uma variável final uma única vez.

```
public class Principal{  
    public static void main(String args[]){  
        final String msg;  
        msg = "Bem vindo!";  
    }  
}
```



# Praticar...

---

- Construa um programa em Java que leia um número inteiro e diga se ele é par ou ímpar.
- Faça um programa em Java que leia números inteiros enquanto não for digitado o número -1, e calcule e imprima a soma destes números
- Faça um programa em Java que calcule e imprima a soma dos 10 primeiros múltiplos de 3.



# Referências

---

- Deitel, P. J.; Deitel, H. M. (2017). Java como programar. 10a edição. São Paulo: Pearson Prentice Hall.
- Barnes, D. J. (2009). Programação orientada a objetos com Java: uma introdução prática usando o BlueJ (4. ed.). São Paulo, SP: Prentice Hall.
- Boratti, I. C. (2007). Programação orientada a objetos em Java. Florianópolis, SC: Visual Books.