

# Unidade 0 - Nivelamento - Introdução ao Java



**PUC Minas**

Instituto de Ciências Exatas e Informática  
Departamento de Ciência da Computação

- Introdução
- Primeiros Programas em Java
- *Integrated Development Environments* (IDEs) para Java
- Variáveis e Expressões
- Estruturas de Controle
- *Arrays*
- *String*



- **Introdução**
- Primeiros Programas em Java
- *Integrated Development Environments* (IDEs) para Java
- Variáveis e Expressões
- Estruturas de Controle
- *Arrays*
- *String*



# História

- Desenvolvida pela  na década de 90
- Adquirida pela **ORACLE®** em 2009
- Inicialmente, voltada para microaplicações
- Cresceu com a ascensão da Internet (Java multiplataforma)

# Origem do Nome

- Java é uma ilha produtora de um tipo de café apreciado por programadores americanos



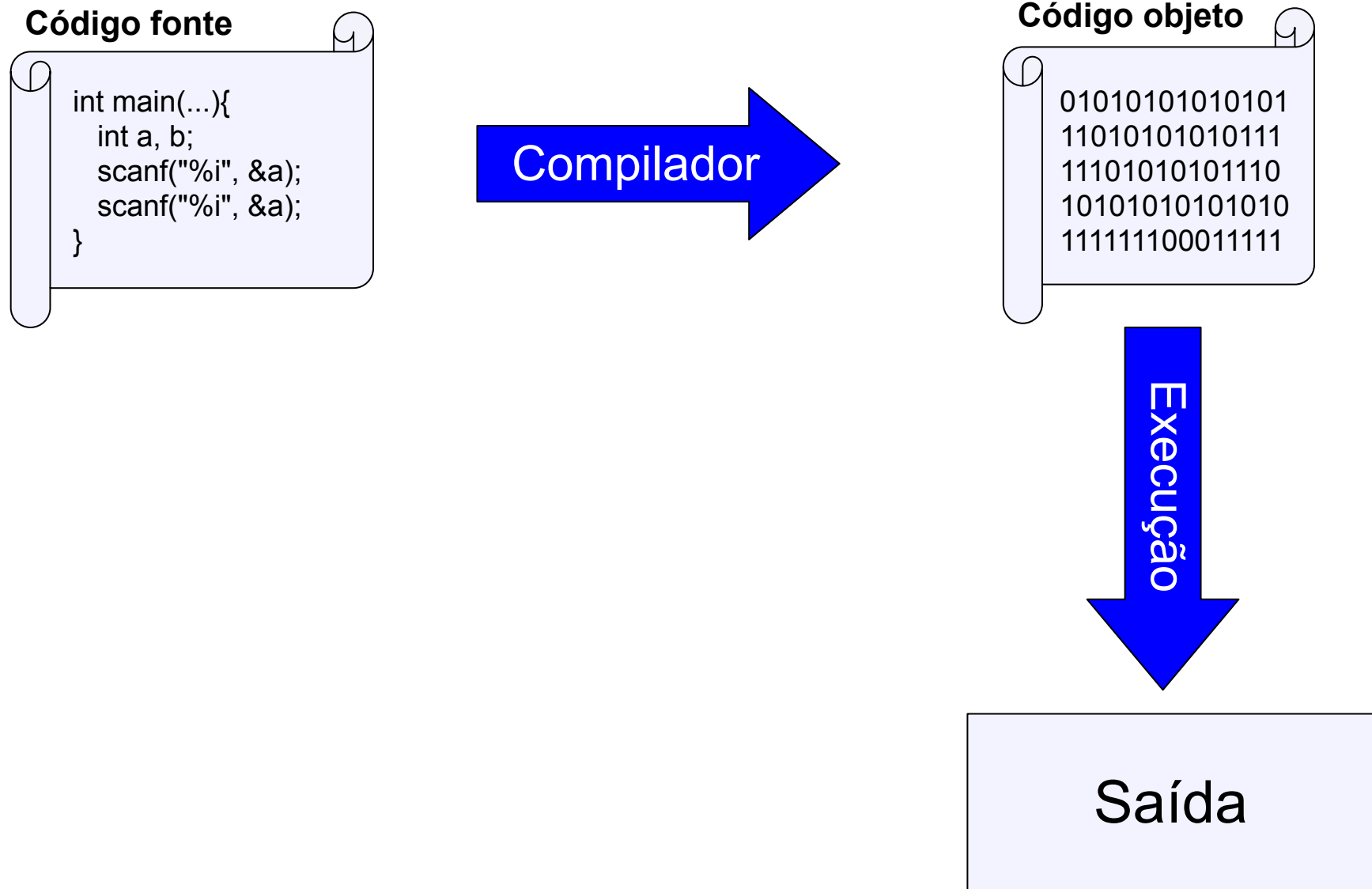
# Características

- Linguagem do tipo *C-like*
- Linguagem Orientada por Objetos
- Aplicações gerais, computação ubíqua, embarcada e web
- Multiplataforma: Escreva uma vez, execute em qualquer lugar

# Características

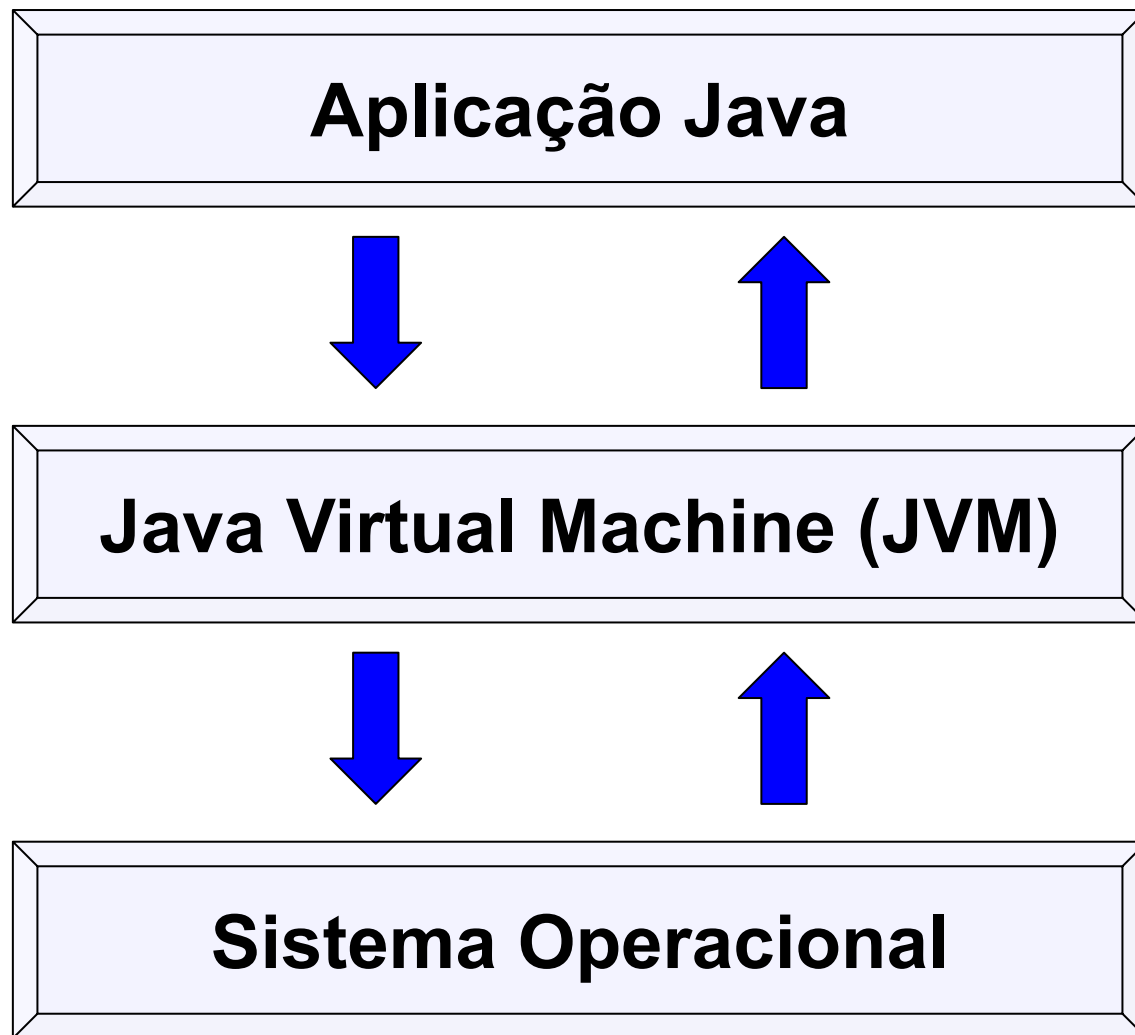
- Segurança: a execução dos códigos é controlada pela JVM
- Projetada para rede: “A rede é o computador”
- Padronização e Internacionalização
- Linguagem simples, poderosa e produtiva

# Compilador Tradicional (C/C++/Pascal)



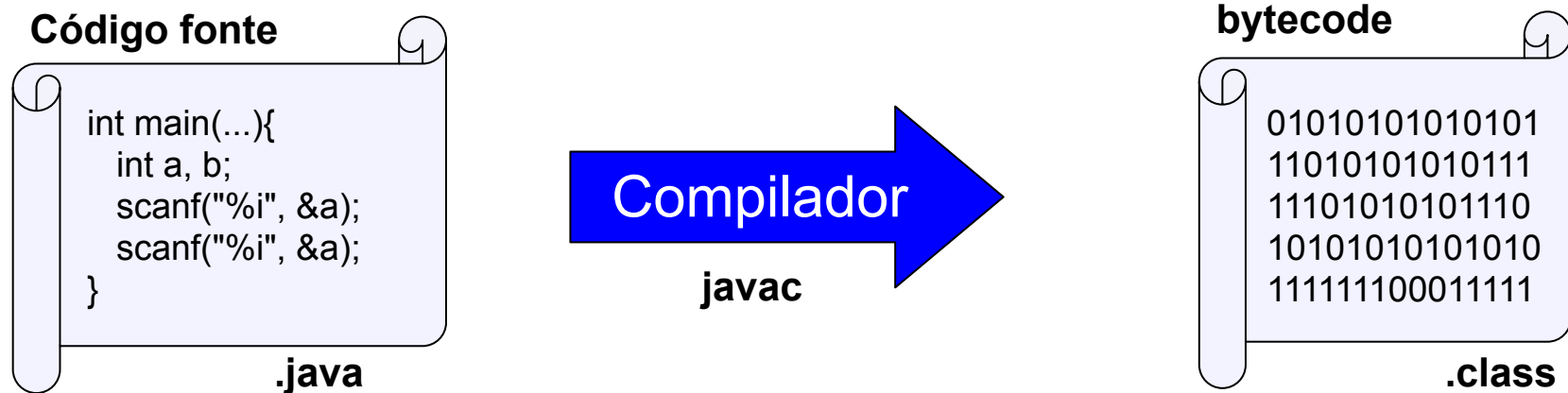


# Java Virtual Machine (JVM)



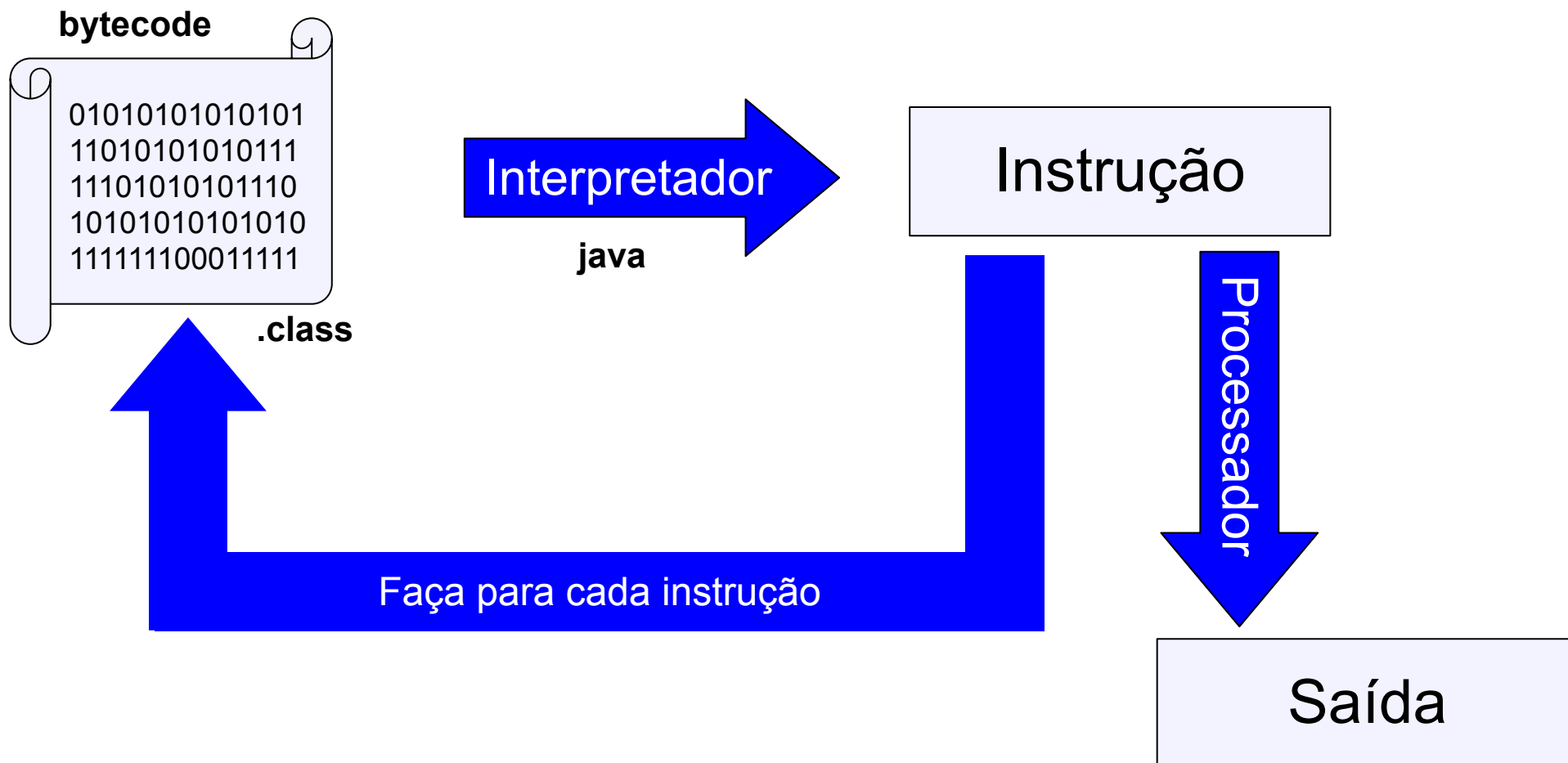
# Compilador e Interpretador Java

- Um programa Java é compilado como *bytecodes* (código objeto para a JVM)



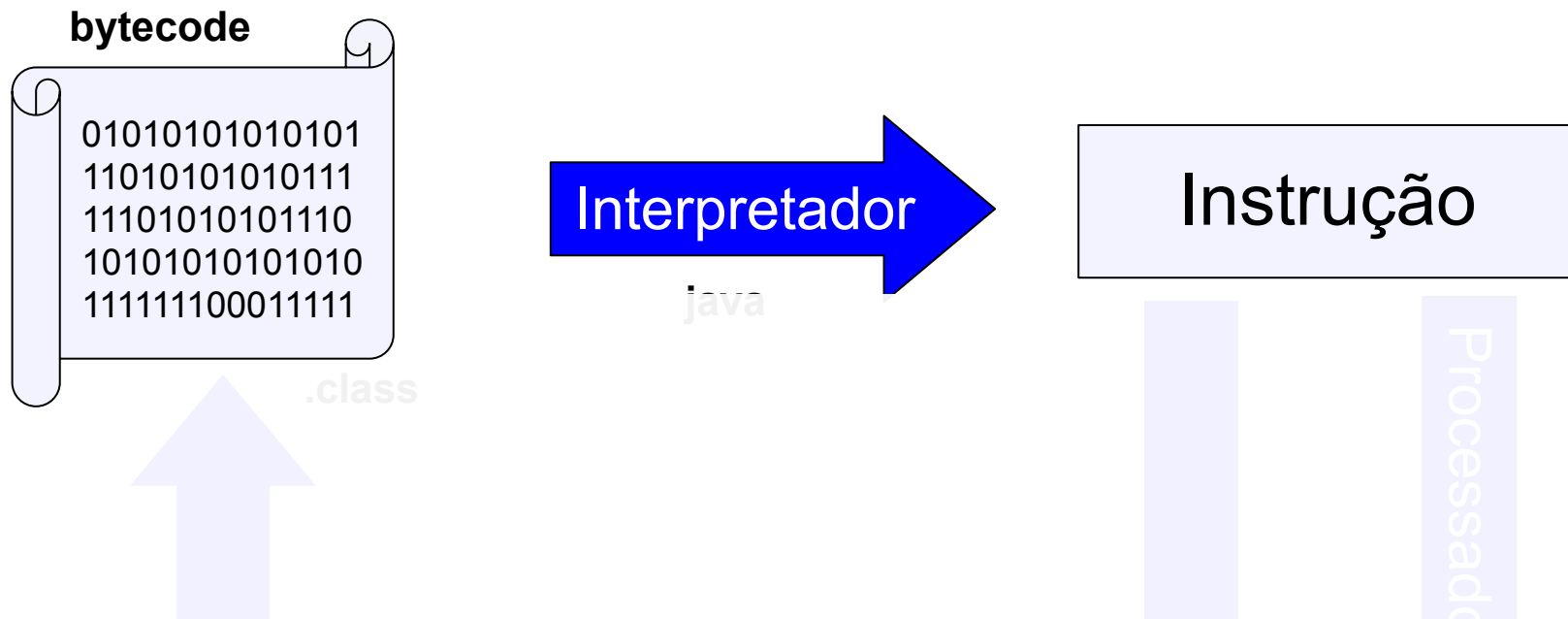
# Compilador e Interpretador Java

- JVM é um processador virtual que interpreta os bytecodes e gera instruções que serão executadas pelo processador



# Compilador e Interpretador Java

- JVM é um processador virtual que interpreta os bytecodes e gera instruções que serão executadas pelo processador



A interpretação faz com que Java seja multiplataforma e não tão rápida

# Algumas Soluções para a Questão Velocidade

- Just In Time *Compiler* (JIT):
  - Traduz os *bytecodes* para instruções de máquina nativas, contudo, perde a multiplataforma e tem aumento no tamanho do código
- JVM em hardware:
  - Por exemplo: picoJava, microJava e UltraJava™

# Como Começar

- Utilize qualquer editor de texto simples
  - Windows (bloco de notas) e Linux (vim ou gEdit)
  - jGrasp
  - Existem IDEs robustas como Eclipse e NetBeans
- *Java Virtual Machine*
  - Baixe no site da Oracle
  - No Linux, durante, a instalação, pode-se selecionar as opções de Java

# Observações para Algumas Linguagens C-like

- Declaração de variáveis, comentários e sintaxe do if, while, do-while, for, switch-case é igual em C, C++, C# e Java
- Em C e C++, chamamos métodos e arrays do Java de funções e vetores, respectivamente

- Introdução
- **Primeiros Programas em Java**
- *Integrated Development Environments* (IDEs) para Java
- Variáveis e Expressões
- Estruturas de Controle
- *Arrays*
- *String*





# Nosso Primeiro Programa

```
class ExemploEscrever1 {  
    public static void main (String args[]){  
        System.out.println("Alo pessoal!!!");  
    }  
}
```

# Nosso Primeiro Programa

O nome do arquivo deve ser  
ExemploEscrever1.java

```
class ExemploEscrever1 {  
    public static void main (String args[]){  
        System.out.println("Alo pessoal!!!");  
    }  
}
```

# Nosso Primeiro Programa

```
class ExemploEscrever1 {  
    public static void main (String args[]){  
        System.out.println("Alo pessoal!!!");  
    }  
}
```

O Java diferencia letras  
maiúsculas e minúsculas

# Nosso Primeiro Programa

- Execute o compilador java para criar os *bytecodes* (arquivo ExemploEscrever1.class)

```
javac ExemploEscrever1.java (na linha de comando)
```

# Nosso Primeiro Programa

- Execute o compilador java para criar os *bytecodes* (arquivo ExemploEscrever1.class)

**javac** ExemploEscrever1.java (na linha de comando)

Se tiver algum erro, o compilador informa cada erro / linha

# Nosso Primeiro Programa

- Execute o interpretador java e veja a mensagem “Alo Pessoal!!!” na tela

**java** ExemploEscrever1 (na linha de comando)

Veja que colocamos apenas  
ExemploEscrever sem .class nem  
.java

# Estrutura dos Programas em Java

```
class NOME_DA_CLASSE {  
  
    public static void main (String args[]){  
        ...  
    }  
}
```

# Estrutura dos Programas em Java

```
class NOME_DA_CLASSE {
```

```
    public static void main
```

Nome do seu arquivo sem a  
extensão .java

```
    }  
}
```



# Estrutura dos Programas em Java

```
class NOME_DA_CLASSE
```

Em nosso exemplo,  
System.out.println("Alo Pessoal!!!");

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

Aqui entra seu código

```
}
```

```
}
```

# Mais Exemplos de Escrita

```
class ExemploEscrever2 {  
    public static void main (String args[]){  
        System.out.println("Alo");  
        System.out.println(" pessoal!!!");  
    }  
}
```

# Mais Exemplos de Escrita

```
class ExemploEscrever3 {  
    public static void main (String args[]){  
        System.out.println("Alo" + " pessoal!!!");  
    }  
}
```

# Mais Exemplos de Escrita

```
class ExemploEscrever4 {  
    public static void main (String args[]){  
        System.out.println("Inteiro: " + 13);  
        System.out.println("Real: " + 13.45);  
    }  
}
```

# Mais Exemplos de Escrita

```
class ExemploEscrever5 {  
    public static void main (String args[]){  
        int n;  
        n = 2 + 3;  
        System.out.println("Inteiro: " + n);  
    }  
}
```

# Mais Exemplos de Escrita

## Atribuição em Java

```
class ExemploEscrever5 {  
    public static void main (String args[]){  
        int n;  
        n = 2 + 3;  
        System.out.println("Inteiro: " + n);  
    }  
}
```

# Mais Exemplos de Escrita

// Exemplo de como somar dois números

```
class ExemploEscrever6 {  
    public static void main (String args[]){  
        double x, y; // números reais de dupla precisão  
        x = 2;  
        y = 3.0; /* iniciando o y, e fazendo y = y+x; */    y = y + x;  
        System.out.println("Valor x+y:" + (x+y));  
    } //fim do método principal  
} /* fim da classe */
```

# Mais Exemplos de Escrita

Existem dois tipos de comentários:  
→ linha (//)  
→ bloco (/\* ... \*/)

// Exemplo de como somar

```
class ExemploEscrever6 {  
    public static void main (String args[]){  
        double x, y; // números reais de dupla precisão  
        x = 2;  
        y = 3.0; /* iniciando o y, e fazendo y = y+x; */    y = y + x;  
        System.out.println("Valor x+y:" + (x+y));  
    } //fim do método principal  
} /* fim da classe */
```



# Exemplo de Leitura com Classe Scanner

- Faça um programa que leia uma *String*, caractere, inteiro e real. Em seguida, seu programa deve imprimir na tela os valores lidos do teclado

Arquivo: ExemploScanner.java

# Exemplo de Leitura com Classe BufferedReader

- Faça um programa que leia uma *String*, caractere, inteiro e real. Em seguida, seu programa deve imprimir na tela os valores lidos do teclado

Arquivo: ExemploBufferedReader.java

- Introdução
- Primeiros Programas em Java
- ***Integrated Development Environments (IDEs) para Java***
- Variáveis e Expressões
- Estruturas de Controle
- *Arrays*
- *String*



- Introdução
- Primeiros Programas em Java
- ***Integrated Development Environments (IDEs) para Java***
- Variáveis e Expressões
- Estruturas de Controle
- *Arrays*
- *String*



# *Integrated Development Environments* para Java

- Existem diversas opções de IDEs como, por exemplo, o jGRASP, Eclipse e Netbeans

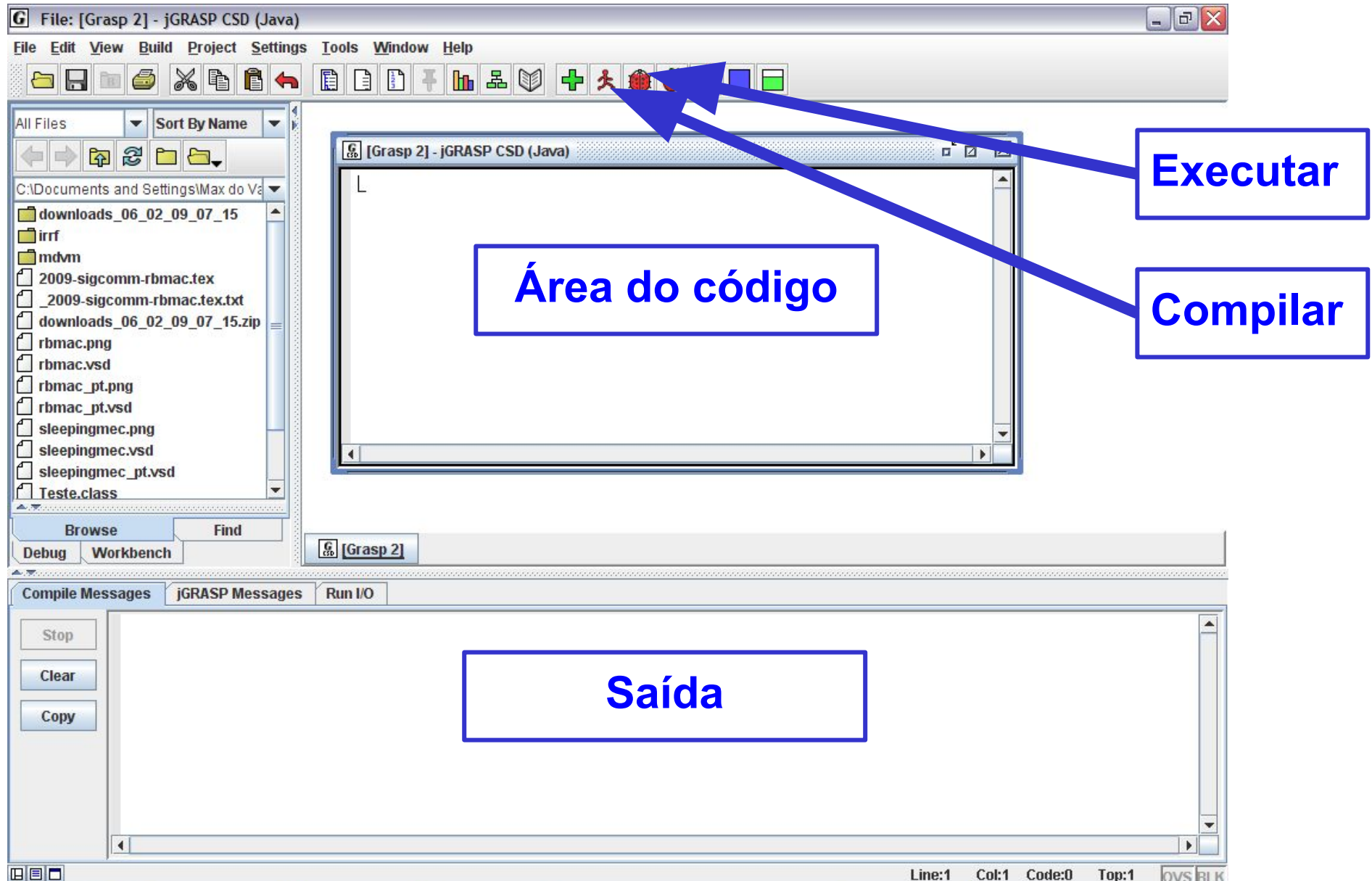


## *Integrated Development Environments* para Java

- O jGRAP é mais simples e tem menos recursos que as demais, verdadeiros ambientes integrados de desenvolvimento
- Antes de instalar qualquer um deles, primeiro, procure (google), baixe e instale o JDK
- Em seguida, para qualquer um deles, procure (google), baixe e instale a IDE

# *Integrated Development Environments para Java*







# *Integrated Development Environments* para Java

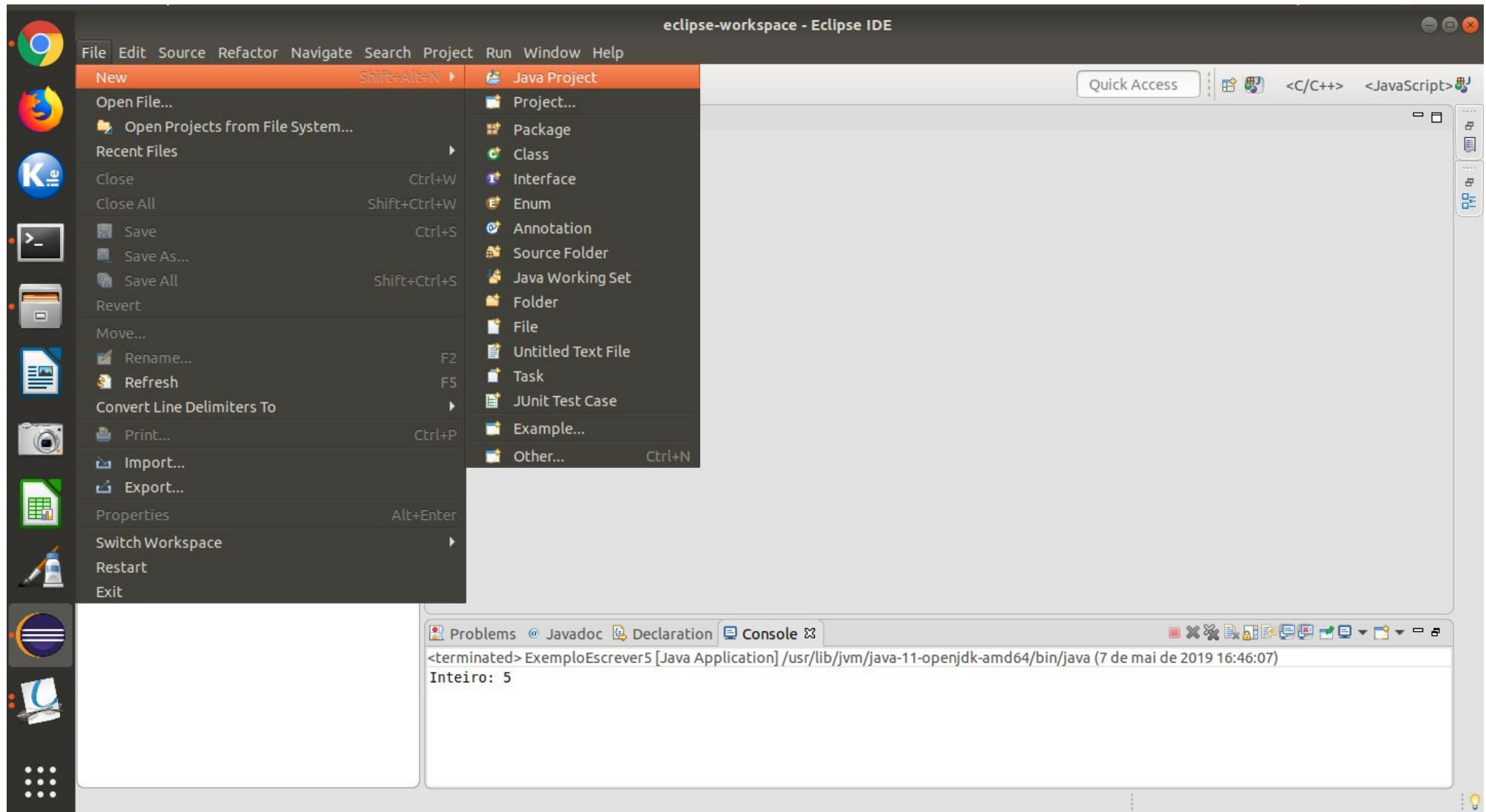


- Primeira tarefa: Criar um novo projeto Java
- Segunda tarefa: Criar uma nova classe
- Terceira tarefa: Implementar, compilar e executar
- Quarta tarefa: Importar uma classe

- **Primeira tarefa: Criar um novo projeto Java**
- Segunda tarefa: Criar uma nova classe
- Terceira tarefa: Implementar, compilar e executar
- Quarta tarefa: Importar uma classe

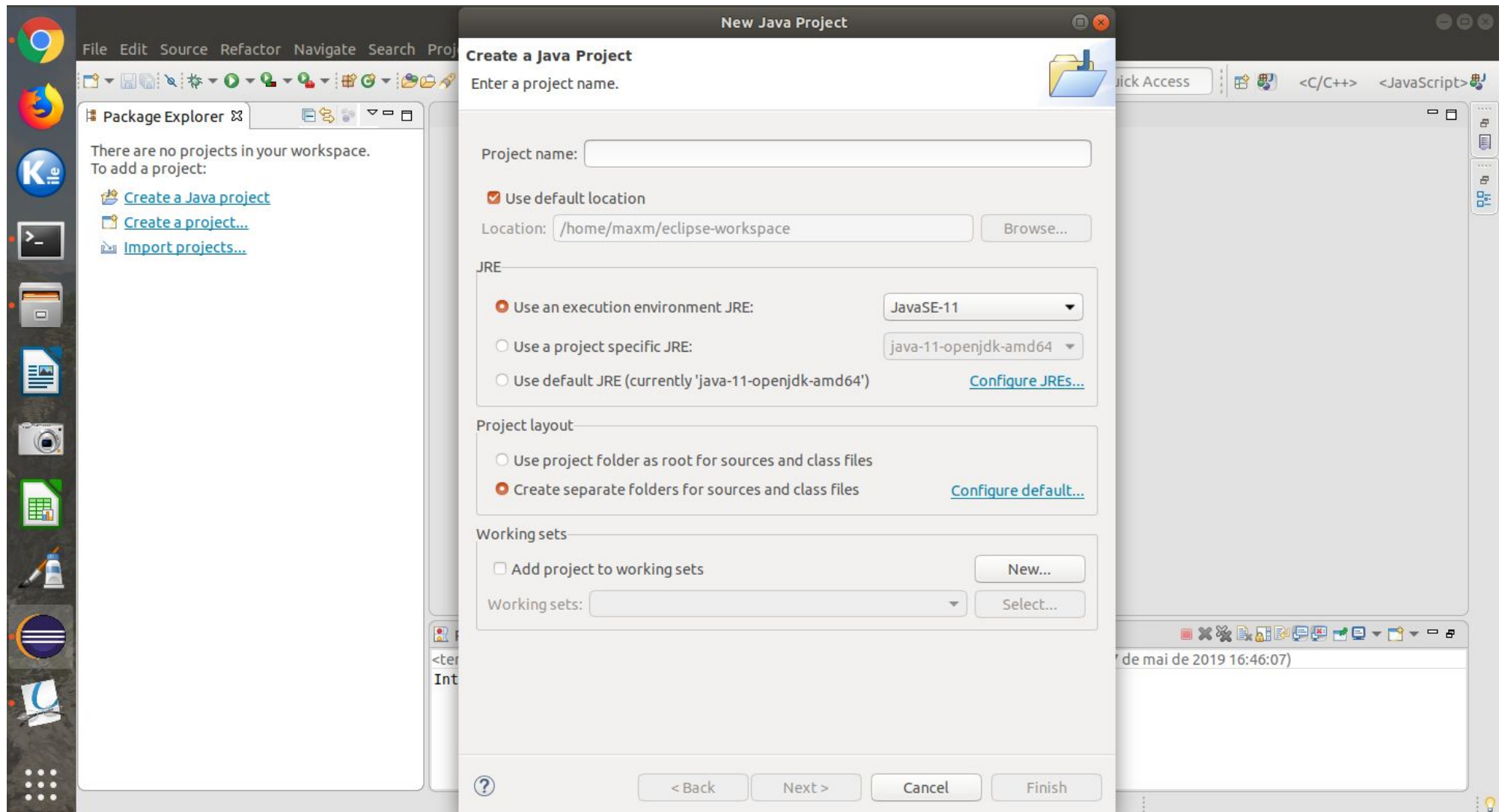
# Tarefa 1: Criar um Novo Projeto Java

- Escolha a opção ***File / New / Java Project***



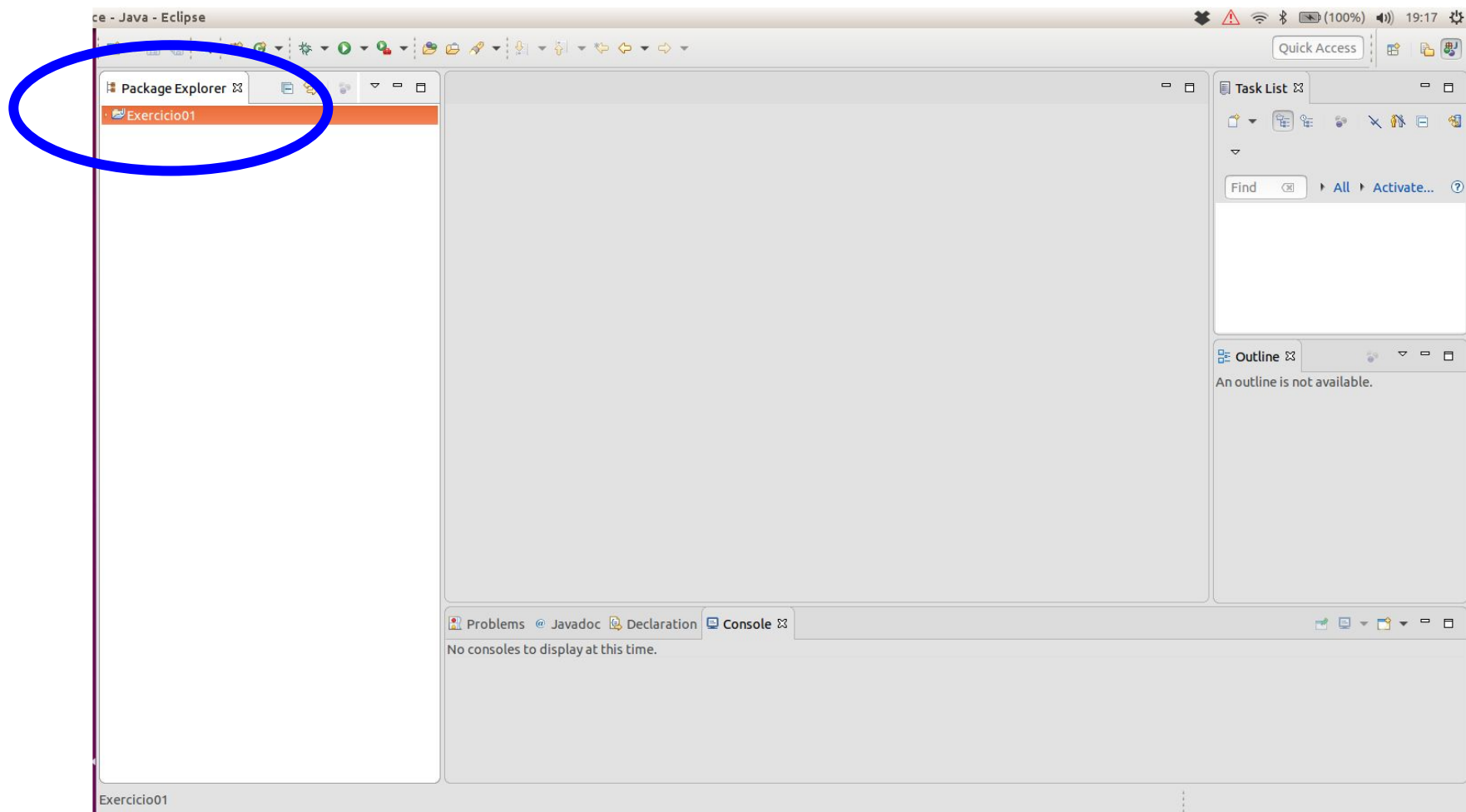
# Tarefa 1: Criar um Novo Projeto Java

- Escolha o ***Project Name*** como Unidade01 e, depois, ***Finish***



# Tarefa 1: Criar um Novo Projeto Java

- O ***Package Explorer*** contém uma pasta com o nome do seu projeto (se não aparecer, vá em ***Windows / Show View***)



# Tarefa 1: Criar um Novo Projeto Java

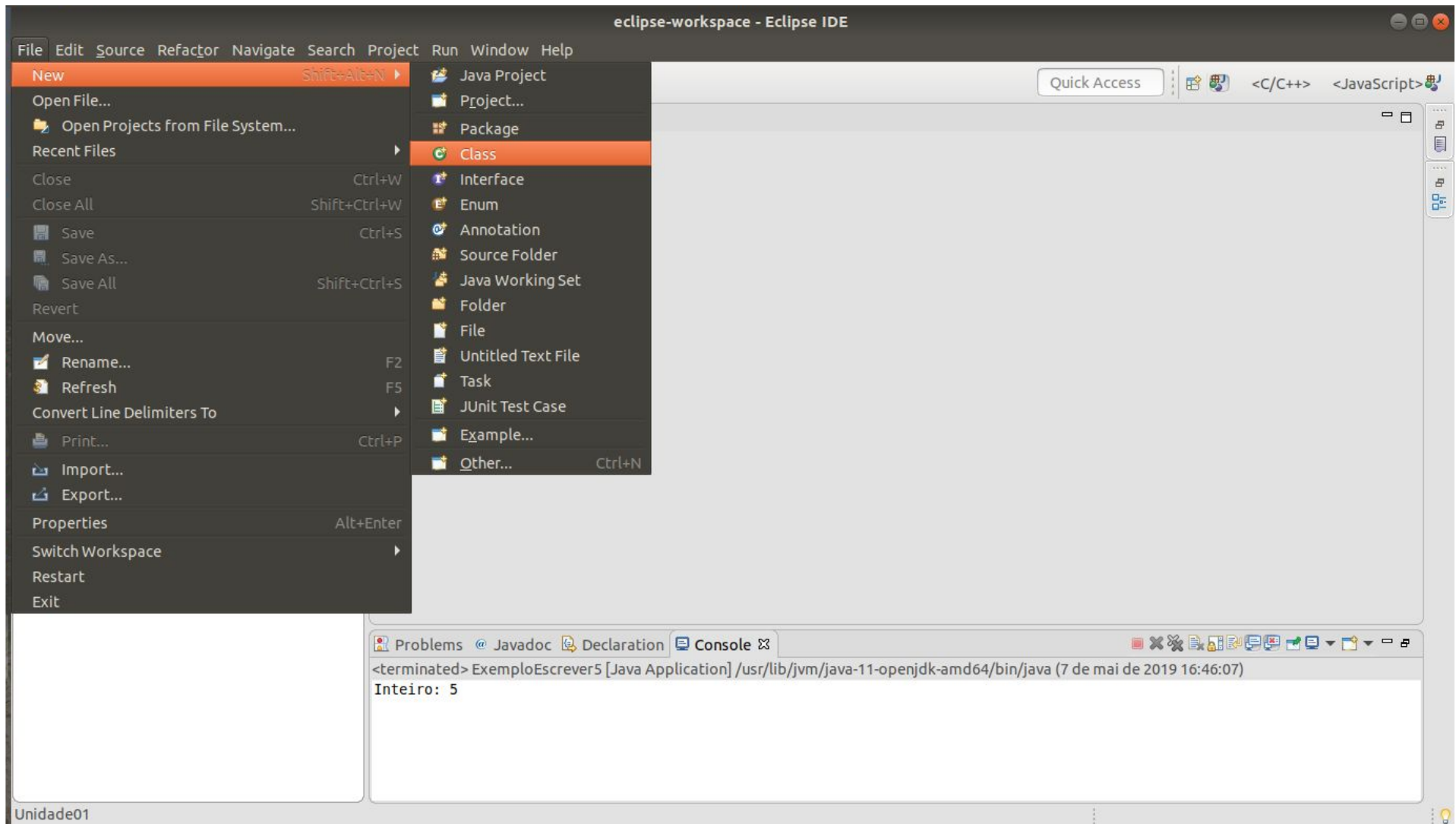
- Abra a pasta ***workspace/NomeProjeto*** e veja que ela tem duas pastas: src (arquivos .java) e bin (arquivos .class)

- Primeira tarefa: Criar um novo projeto Java
- **Segunda tarefa: Criar uma nova classe**
- Terceira tarefa: Implementar, compilar e executar
- Quarta tarefa: Importar uma classe



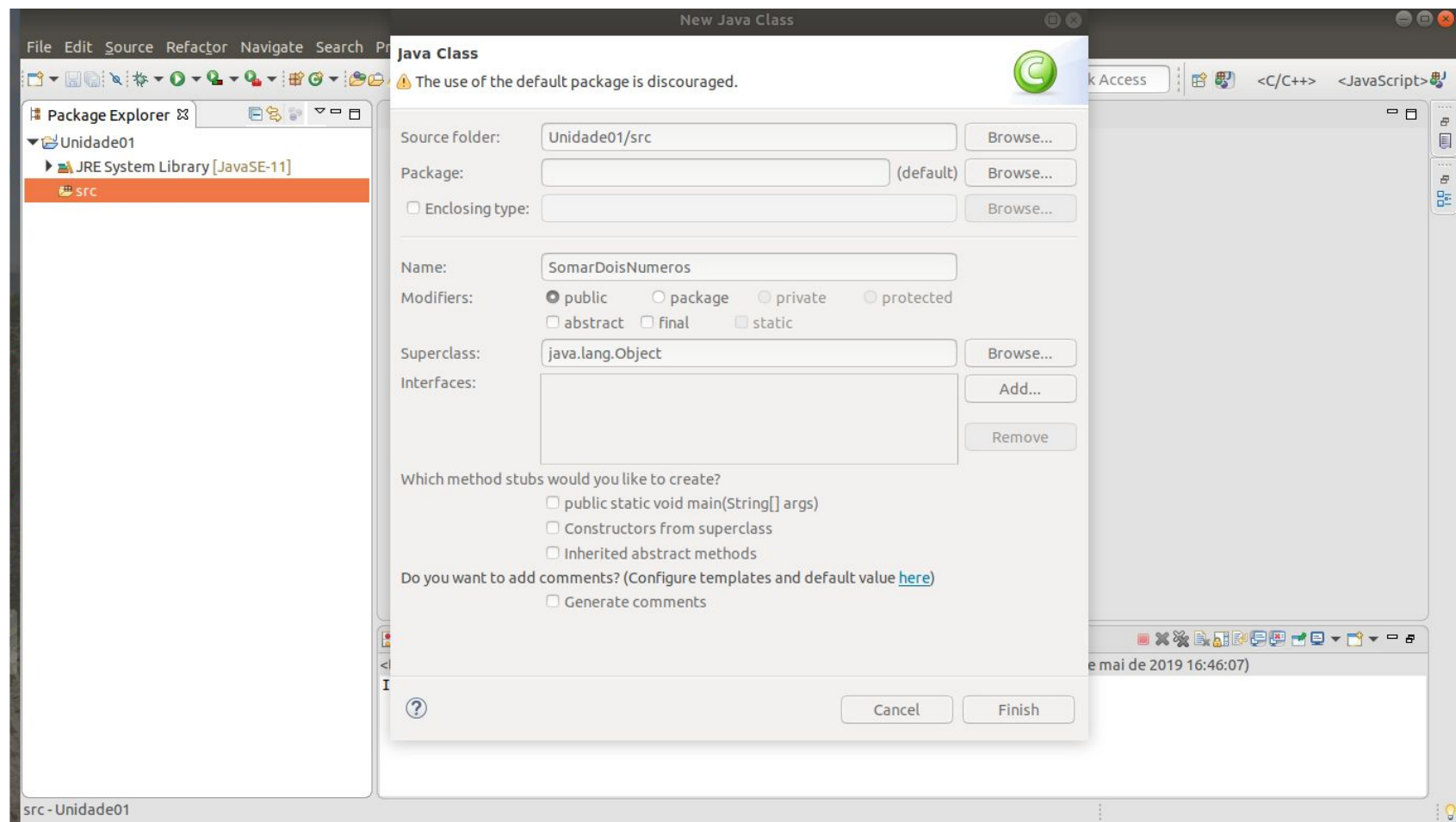
# Tarefa 2: Criar uma Nova Classe

- Escolha a opção ***File / New / Class***



# Tarefa 2: Criar uma Nova Classe

- Escolha o nome de sua classe (e.g., SomarDoisNumero) e pressione o botão ***Finish***



## Tarefa 2: Criar uma Nova Classe

- Na pasta ***workspace/NomeProjeto/src***, temos o arquivo **.java** (e.g., SomarDoisNumeros.java)

- Primeira tarefa: Criar um novo projeto Java
- Segunda tarefa: Criar uma nova classe
- **Terceira tarefa: Implementar, compilar e executar**
- Quarta tarefa: Importar uma classe

## Tarefa 3: Implementar, Compilar e Executar

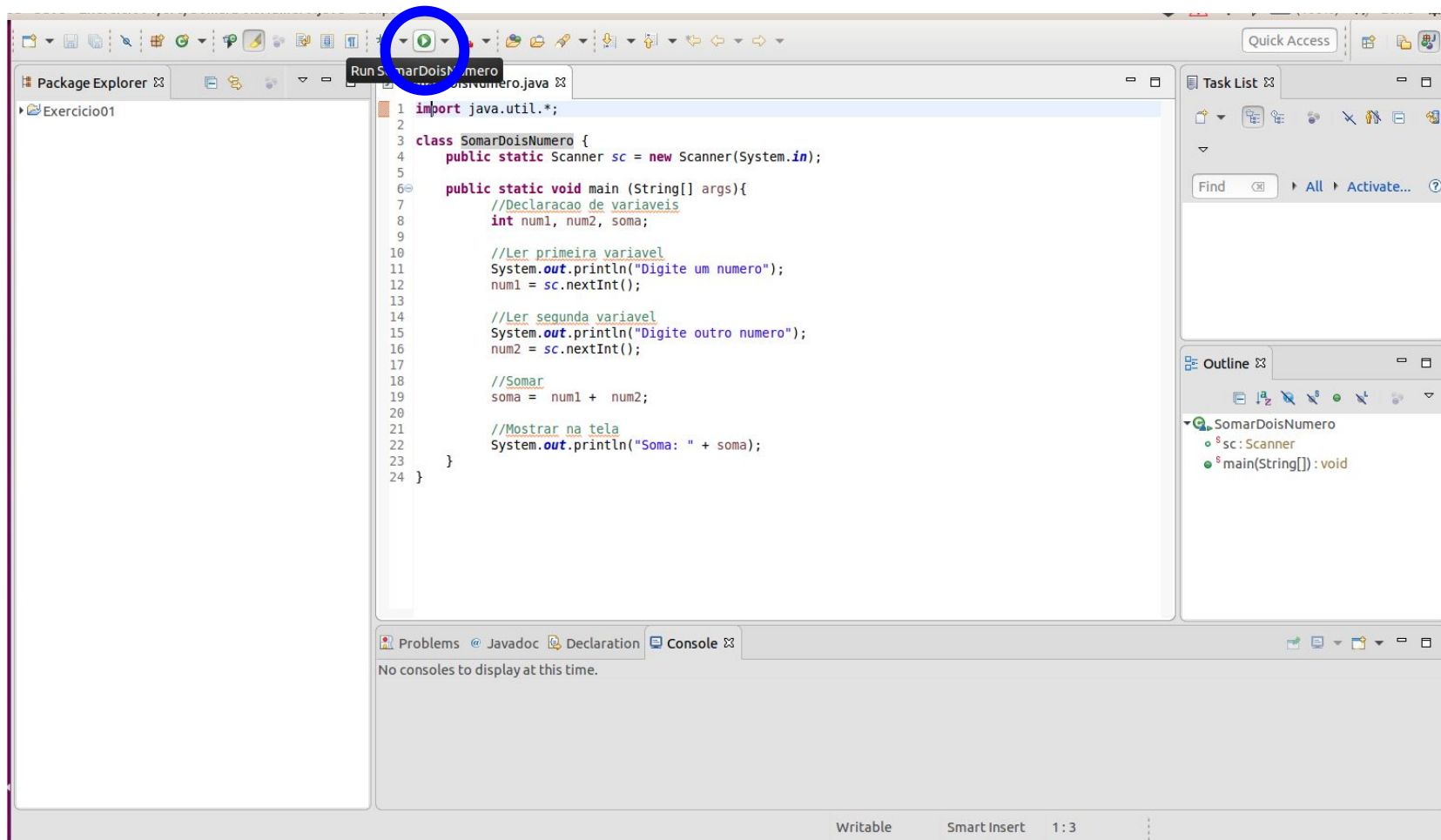
- Implemente um programa para ler dois números e mostrar a soma deles na tela seguindo os passos abaixo:
  - Primeiro, antes do código da classe, insira a linha ***import java.util.\*;*** para importar a classe Scanner
  - Segundo, remova a palavra ***public*** antes de ***class***
  - Terceiro, dentro da classe, insira a linha ***public static Scanner sc = new Scanner(System.in);***
  - Quarto, copie o código da próxima página dentro do método main

# Tarefa 3: Implementar, Compilar e Executar

```
public static void main (String args[]){  
    //Declaracao de variaveis  
    int num1, num2, soma;  
  
    //Leituras  
    System.out.println("Digite um número");  
    num1 = sc.nextInt();  
    System.out.println("Digite outro número");  
    num2 = sc.nextInt();  
  
    //Somar  
    soma = num1 + num2;  
  
    //Mostrar na tela  
    System.out.println("Soma:" + soma);  
}
```

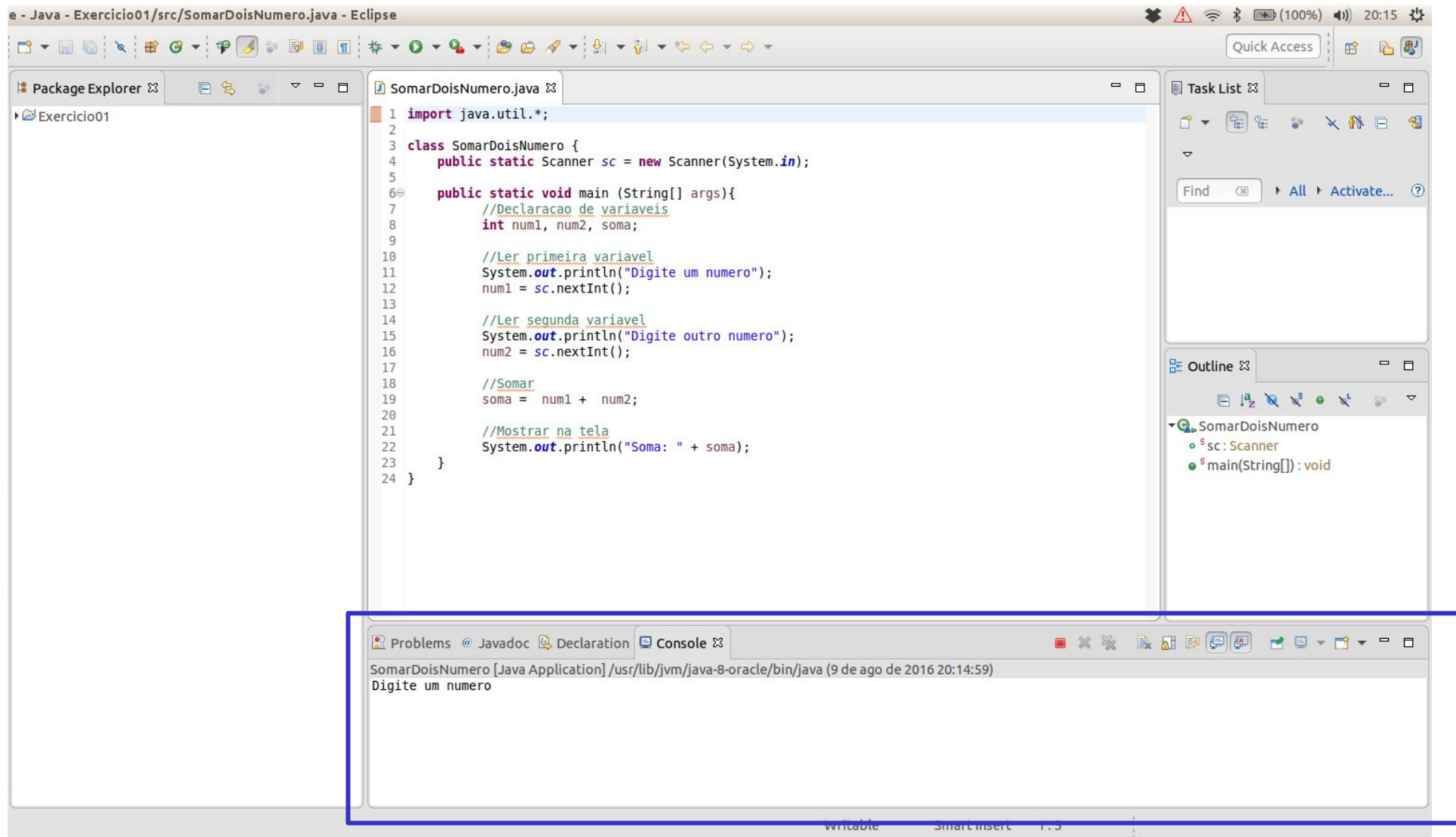
# Tarefa 3: Implementar, Compilar e Executar

- Compile seu programa pressionando o botão **Run** e veja o **SomarDoisNumero.class** em **workspace/NomeProjeto/bin**



# Tarefa 3: Implementar, Compilar e Executar

- Execute seu programa na área de Console





- Primeira tarefa: Criar um novo projeto Java
- Segunda tarefa: Criar uma nova classe
- Terceira tarefa: Implementar, compilar e executar
- **Quarta tarefa: Importar uma classe**

## Tarefa 4: Importar uma Classe

- No ***Package Explorer***, em src, clique com o botão direito do mouse e escolha a opção ***Import***
- Selecione a opção ***File System*** e o botão ***Next***
- Escolha a pasta da classe a ser importada
- Escolha a classe a ser importada e selecione o botão ***Finish***

# *Integrated Development Environments* para Java



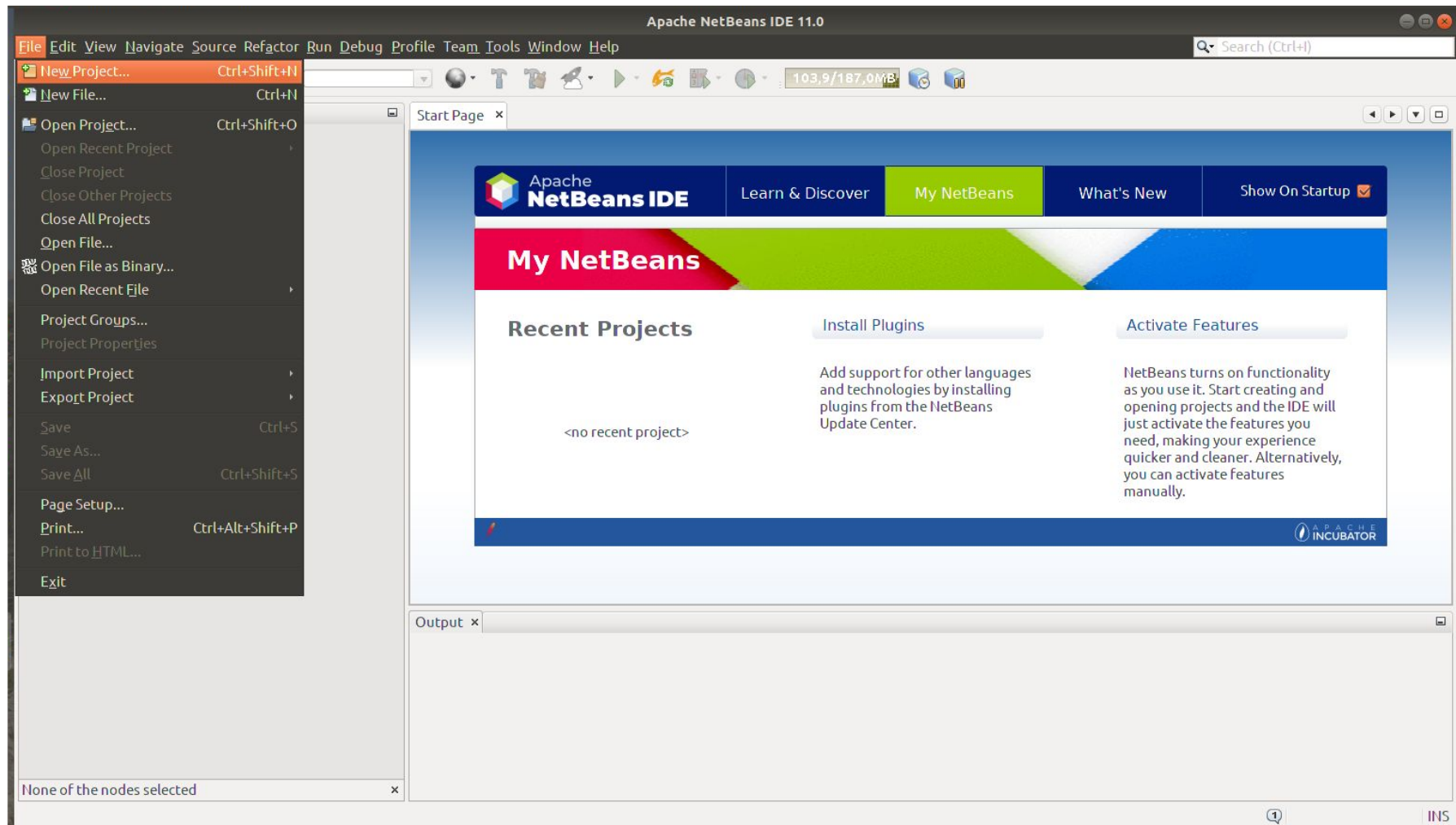
# NetBeans

- Primeira tarefa: Criar um novo projeto Java
- Segunda tarefa: Criar uma nova classe
- Terceira tarefa: Implementar, compilar e executar
- Quarta tarefa: Importar uma classe

- **Primeira tarefa: Criar um novo projeto Java**
- Segunda tarefa: Criar uma nova classe
- Terceira tarefa: Implementar, compilar e executar
- Quarta tarefa: Importar uma classe

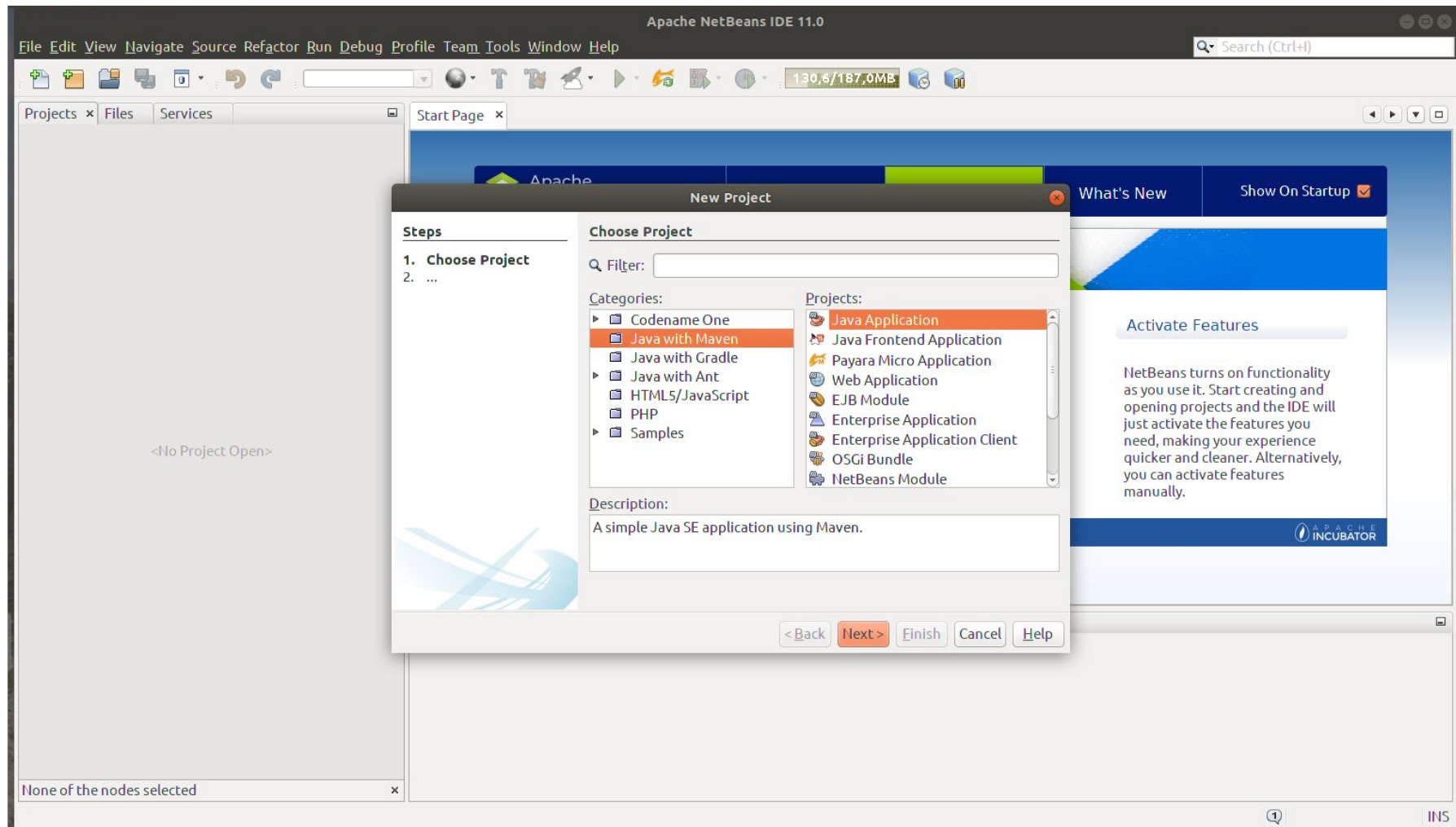
# Tarefa 1: Criar um Novo Projeto Java

- Escolha a opção **File / New Project** ou o botão



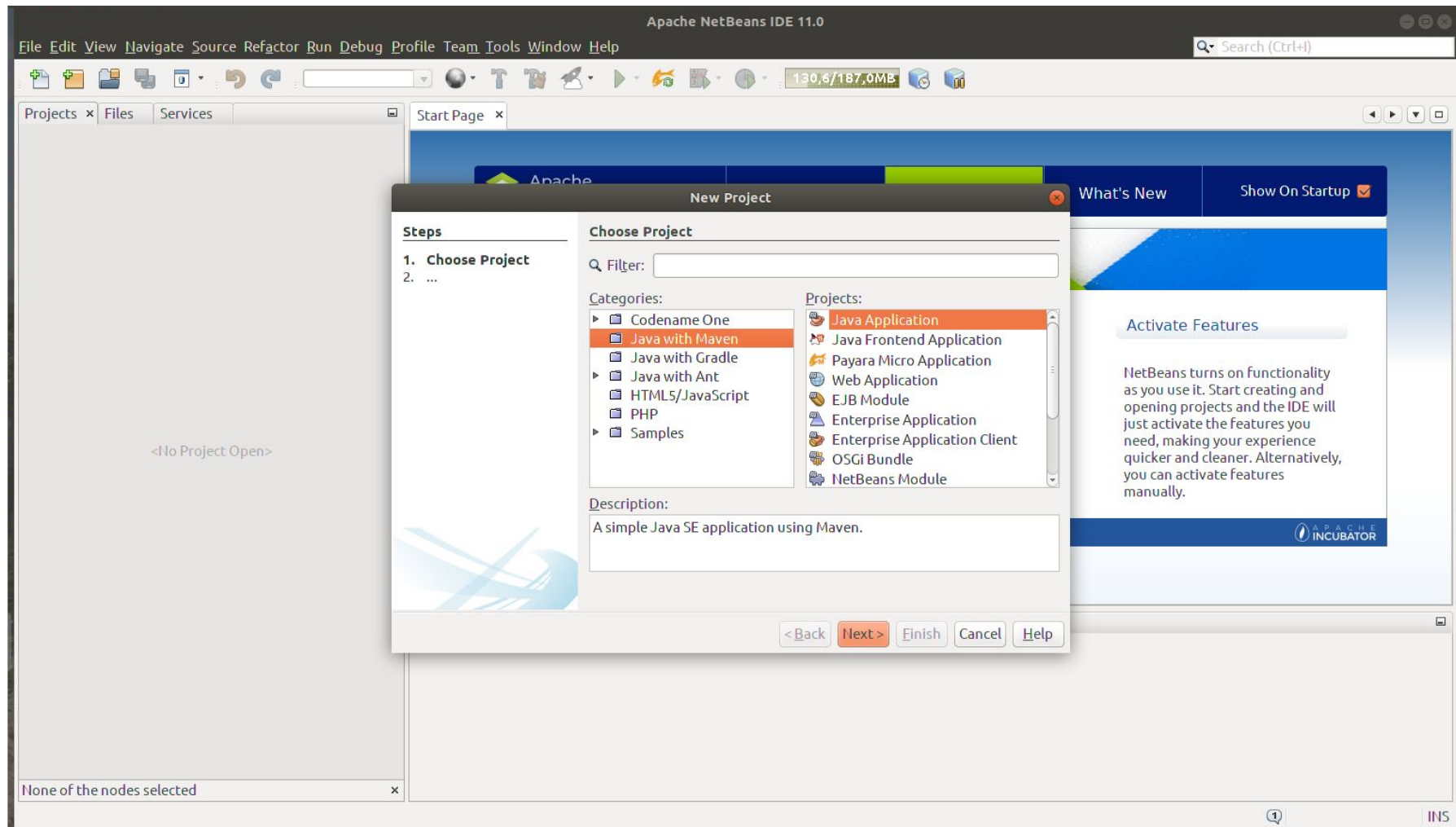
# Tarefa 1: Criar um Novo Projeto Java

- Selecione a opção ***Java Application***



# Tarefa 1: Criar um Novo Projeto Java

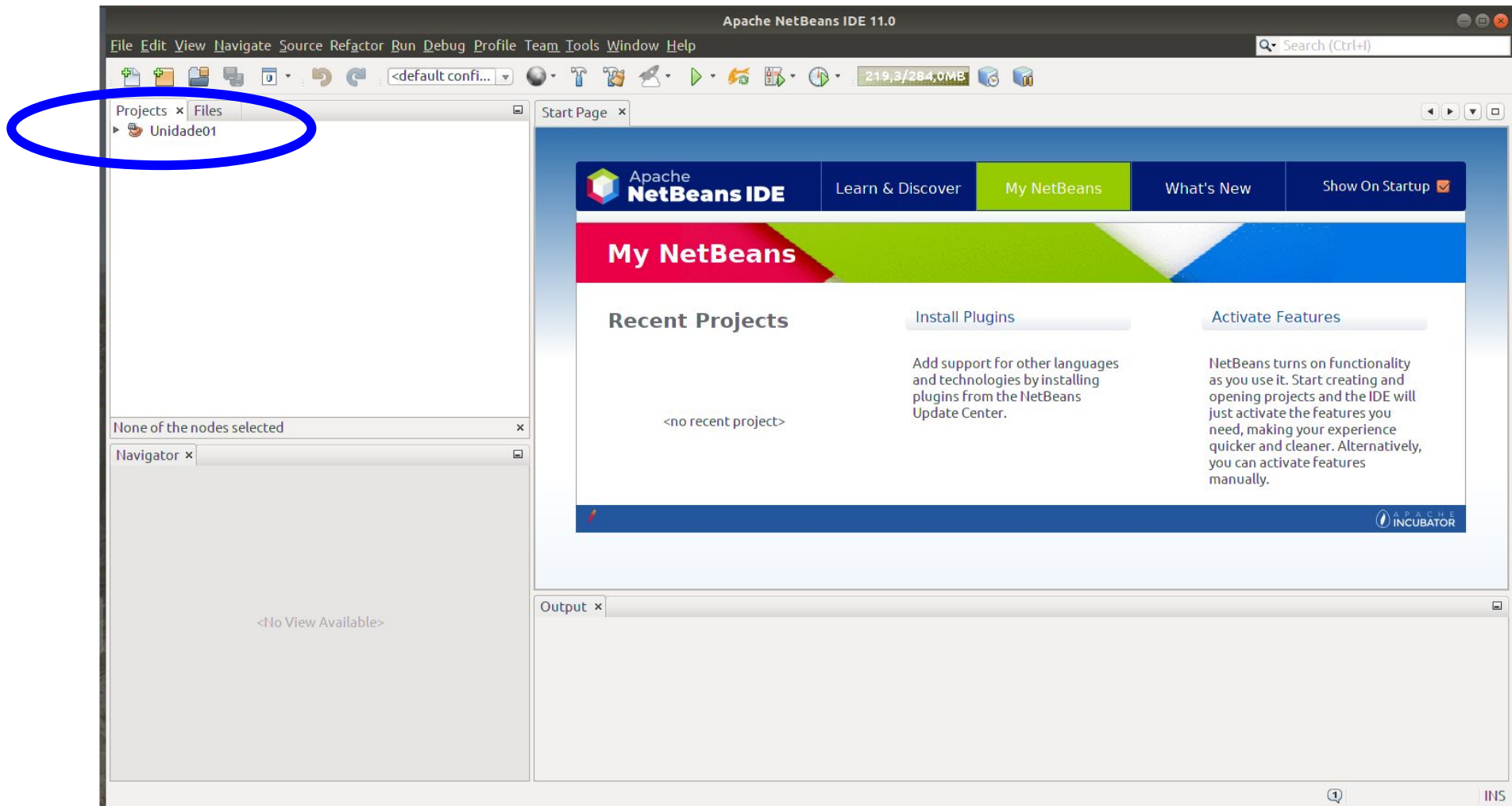
- Escolha o **Project Name** como Unidade01 e, depois, **Finish**





# Tarefa 1: Criar um Novo Projeto Java

- Em **Projects**, temos seu projeto (se não aparecer, Windows)



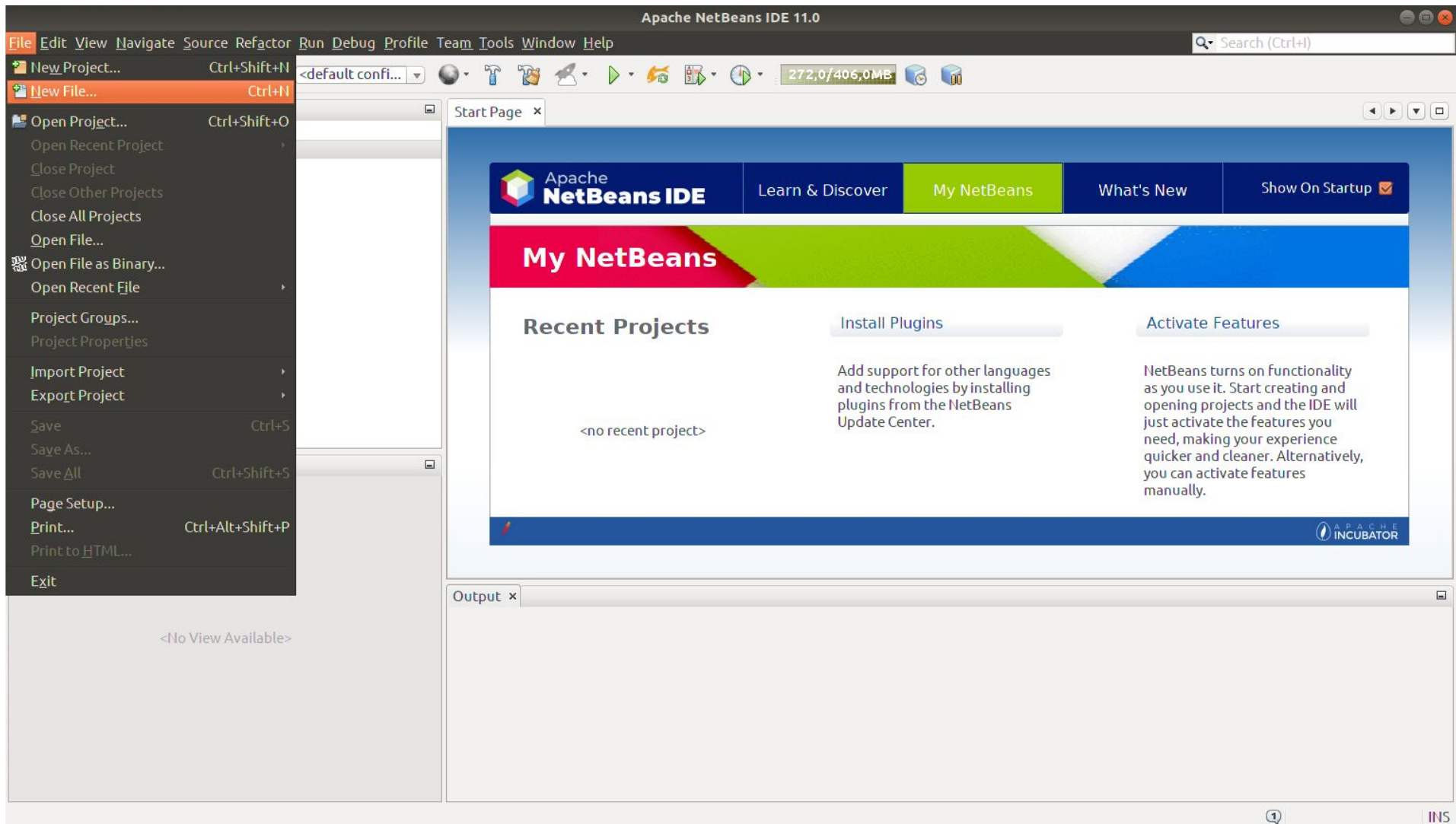
# Tarefa 1: Criar um Novo Projeto Java

- Abra a pasta ***workspace/NomeProjeto*** e veja que ela tem/terá duas pastas:
  - `src/main/java` (arquivos `.java`)
  - `target/classes` (arquivos `.class`)

- Primeira tarefa: Criar um novo projeto Java
- **Segunda tarefa: Criar uma nova classe**
- Terceira tarefa: Implementar, compilar e executar
- Quarta tarefa: Importar uma classe

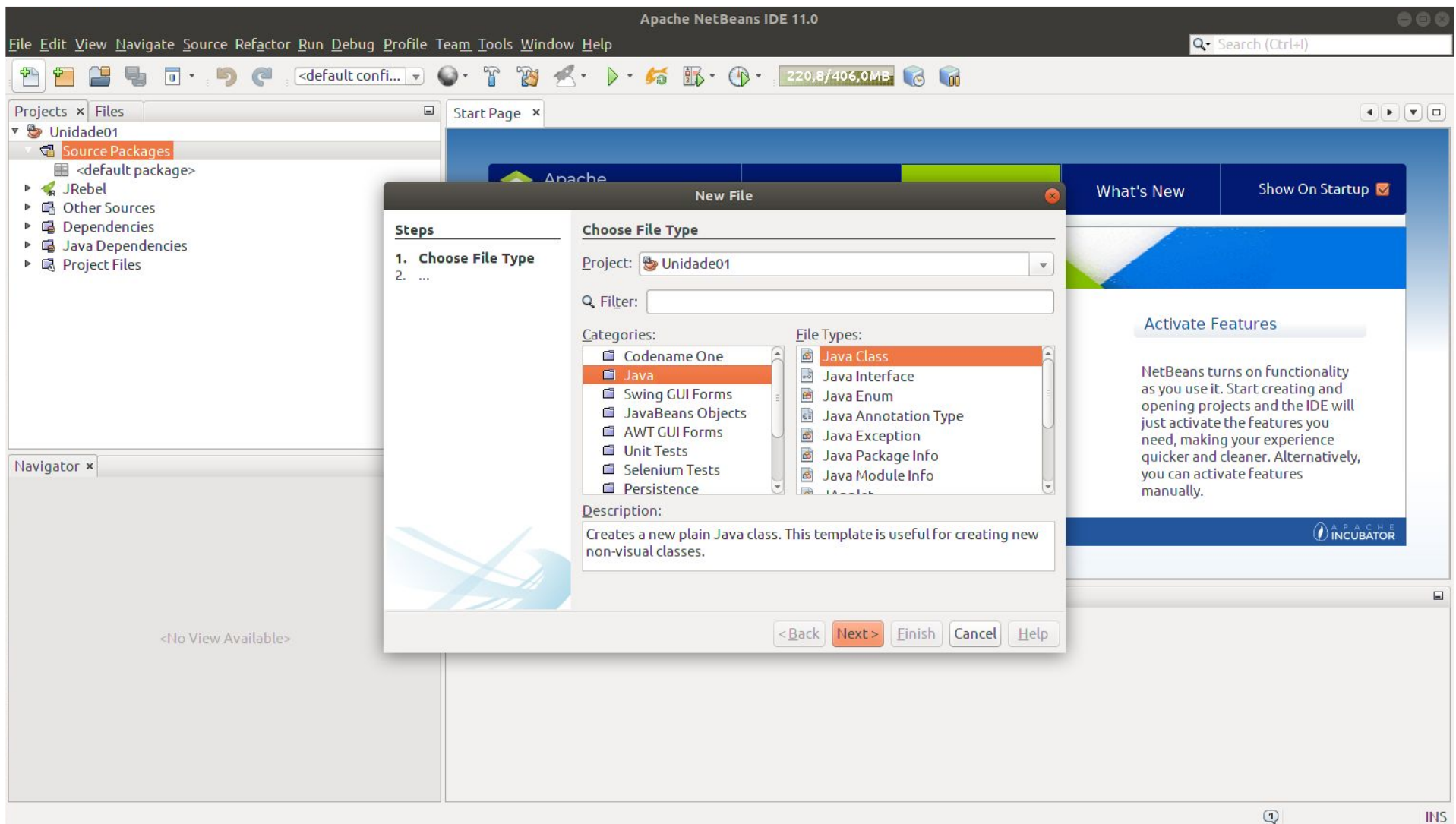
# Tarefa 2: Criar uma Nova Classe

- Escolha a opção **File / New File** ou o botão



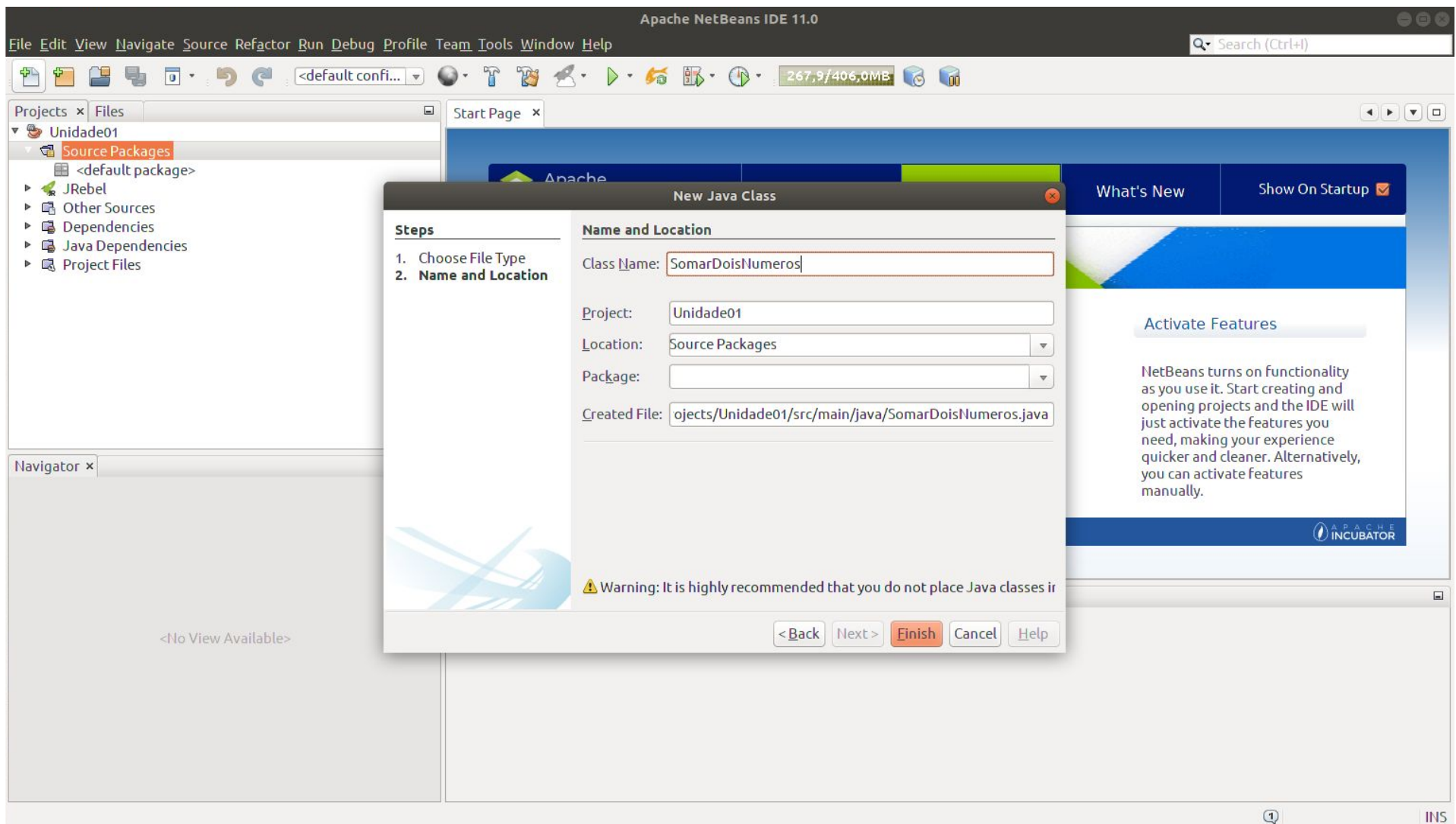
# Tarefa 2: Criar uma Nova Classe

- Escolha a opção **Java / Java Class** e **Next**



# Tarefa 2: Criar uma Nova Classe

- Escolha o **Class Name** (e.g., SomarDoisNumero) e **Finish**



## Tarefa 2: Criar uma Nova Classe

- Na pasta ***workspace/NomeProjeto/src***, temos o arquivo **.java** (e.g., SomarDoisNumeros.java)

- Primeira tarefa: Criar um novo projeto Java
- Segunda tarefa: Criar uma nova classe
- **Terceira tarefa: Implementar, compilar e executar**
- Quarta tarefa: Importar uma classe




## Tarefa 3: Implementar, Compilar e Executar

- Implemente um programa para ler dois números e mostrar a soma deles na tela seguindo os passos abaixo:
  - Primeiro, antes do código da classe, insira a linha ***import java.util.\*;*** para importar a classe Scanner
  - Segundo, remova a palavra ***public*** antes de ***class***
  - Terceiro, dentro da classe, insira a linha ***public static Scanner sc = new Scanner(System.in);***
  - Quarto, copie o código da próxima página dentro do método main

# Tarefa 3: Implementar, Compilar e Executar

```
public static void main (String args[]){  
    //Declaracao de variaveis  
    int num1, num2, soma;  
  
    //Leituras  
    System.out.println("Digite um número");  
    num1 = sc.nextInt();  
    System.out.println("Digite outro número");  
    num2 = sc.nextInt();  
  
    //Somar  
    soma = num1 + num2;  
  
    //Mostrar na tela  
    System.out.println("Soma:" + soma);  
}
```

## Tarefa 3: Implementar, Compilar e Executar

- Salve o código implementado
- Em ***File / Project Properties (Unidade01) / Run / Main Class***, defina nossa classe como a principal do projeto
- Compile e execute seu código em Run (ou no botão )

- Primeira tarefa: Criar um novo projeto Java
- Segunda tarefa: Criar uma nova classe
- Terceira tarefa: Implementar, compilar e executar
- **Quarta tarefa: Importar uma classe**

## Tarefa 4: Importar uma Classe

- Copie os arquivos para src/main/java
- Se necessário, redefina a classe principal em ***File / Project Properties (Unidade01) / Run / Main Class***
- Se necessário, em Projetos, clique em um arquivo e ***Run***
- Se necessário, em Source, ***Organize Imports*** ou ***Scan for External Changes***

- Introdução
- Primeiros Programas em Java
- *Integrated Development Environments* (IDEs) para Java
- **Variáveis e Expressões**
- Estruturas de Controle
- *Arrays*
- *String*



# Definição de Variável

- Local (endereço) na memória principal que armazena um conteúdo



# Regras para Nome de Variáveis

- O 1º caractere será uma letra (em Java são '\_' e '\$' letras)
- O restante do nome pode conter letras ou dígitos
- O Java diferencia letras maiúsculas de minúsculas
- Palavras reservadas e nomes de funções: não permitidos
  - Algumas palavras-reservadas: class, for, while, public, if, else, int, double...



# Regras para Nome de Variáveis

## • Exemplos **corretos**

- Cont
- Teste23
- Sao\_Joao
- \_\_sizeint
- \$cont
- \_if

# Regras para Nome de Variáveis

## • Exemplos **incorretos**

- 1cont
- oi!gente
- Sao..Joao
- \_size-int
- if

# Boa Prática de Programação

As variáveis começam com letras minúsculas e as classes com letras maiúsculas

# Tipos das Variáveis

boolean	true / false	1 bit
char	caractere unicode	16 bits
byte	inteiro	8 bits
short	inteiro	16 bits
int	inteiro	32 bits
long	inteiro	64 bits
float	real	32 bits
double	real	64 bits

# Declaração de Variáveis

tipo da variável

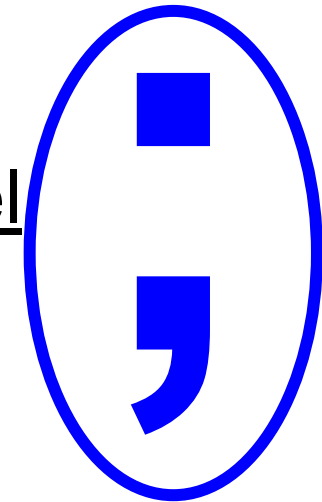
nome da variável



# Declaração de Variáveis

tipo da variável

nome da variável



# Exemplo de Declaração de Variáveis

```
int x;
```

```
double y;
```

```
char c;
```

# Exemplo de Declaração de Variáveis

```
int a, b;
```

```
double x, y, z;
```

```
char c1, c2, c3, c4, c5, c6;
```



# Atribuição

variável



expressão



## Atribuição

variável

=

expressão

;

Não confundir := com = com ==

## Atribuição

variável



expressão



Ver arquivos ExemploAtribuicao1.java,  
ExemploAtribuicao2.java e ExemploAtribuicao3.java

# Expressões

- Compostas por operandos, operadores e pontuações
  - Operandos são variáveis ou constantes (e.g., a, x, 2)
  - Operadores (e.g., +, -, %) e pontuações (e.g., (,),{,},“,”) são símbolos

# Expressões

- Compostas por operandos, operadores e pontuações
  - Operandos são variáveis ou constantes (e.g., a, x, 2)
  - Operadores (e.g., +, -, %) e pontuações (e.g., (,),{,},“,”) são símbolos

Respeitamos:

- prioridade de operadores
- prioridade de pontuação

# Expressões

- Retornam um valor como, por exemplo:
  - $(5 + 4)$  retorna 9
  - $((5+4) == 9)$  retorna true
  - $(9 \& 5)$  retorna 1

# Principais Operadores Unários

- Menos unário ou negação ( **-** ): muda o sinal de um valor

```
int x = 32, y = 0;
```

```
x = -x;
```

```
y = -y;
```

```
System.out.println(x); //Imprime -32
```

```
System.out.println(y); //Imprime 0
```

# Principais Operadores Unários

- Pré ou pós incremento ( **++** ): incrementa um valor

```
int x = 5, y = 10;
```

```
x++; //Equivalente a x = x + 1
```

```
++y; //Equivalente a y = y + 1
```

```
System.out.println(x); //Imprime 6
```

```
System.out.println(y); //Imprime 11
```

```
System.out.println(x++); //Imprime 6
```

```
System.out.println(++y); //Imprime 12
```

```
System.out.println(x); //Imprime 7
```

```
System.out.println(y); //Imprime 12
```



# Principais Operadores Unários

- Pré ou pós decremento ( **--** ): decrementa um valor

```
int x = 5, y = 10;
```

```
x--;           //Equivalente a x = x - 1
```

```
--y;          //Equivalente a y = y - 1
```

```
System.out.println(x);    //Imprime 4
```

```
System.out.println(y);    //Imprime 9
```

```
System.out.println(x--);  //Imprime 4
```

```
System.out.println(--y);   //Imprime 8
```

```
System.out.println(x);    //Imprime 3
```

```
System.out.println(y);    //Imprime 8
```

# Principais Operadores Binários

• **+**, **-**, **\***, **/** e **%**

```
int x = 32, y = 3, z;
```

```
z = x + y; // z recebe 35
```

```
z = x - y; // z recebe 29
```

```
z = x * y; // z recebe 96
```

```
z = x / y; // z recebe 10 (se x ou y fosse real seria 10,67)
```

```
z = x % y; // z recebe 2 (a operação de mod só existe para inteiros)
```

# Operadores para Manipulação de Bits

• <<, >>, ^, ~, & e |

```
int z;  
z = 9 << 1;    //z recebe 18  
z = 9 >> 1;    //z recebe 4  
z = 9 ^ 2;     //z recebe 11 (xor)  
z = ~128;      //z recebe -129 (inversão de zeros e uns)  
z = 9 & 5;     //z recebe 1 (and bit-a-bit)  
z = 9 | 5;     //z recebe 13 (or bit-a-bit)
```

- Introdução
- Primeiros Programas em Java
- *Integrated Development Environments* (IDEs) para Java
- Variáveis e Expressões
- **Estruturas de Controle**
- *Arrays*
- *String*



# Comandos

- Estruturas condicionais:
  - **if-else**
  - Operador ternário
  - switch-case
- Estruturas de repetição:
  - **while**
  - do-while
  - **for**
- Tratamento de exceções

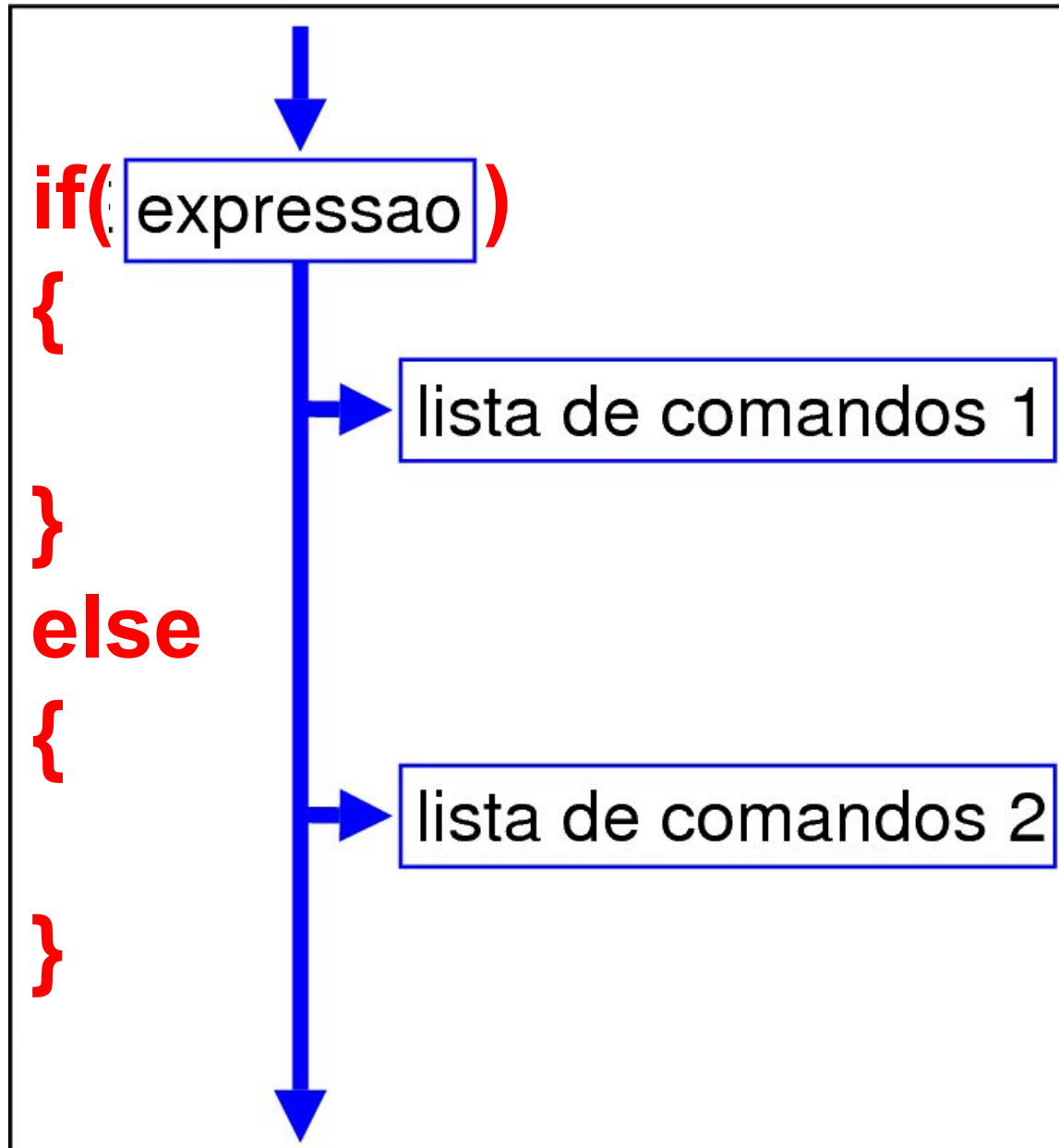


# Comandos

- Estruturas condicionais:
  - **if-else**
  - Operador ternário
  - switch-case
- Estruturas de repetição:
  - while
  - do-while
  - for
- Tratamento de exceções



## Comando if-else



# Exemplo do Comando if-else

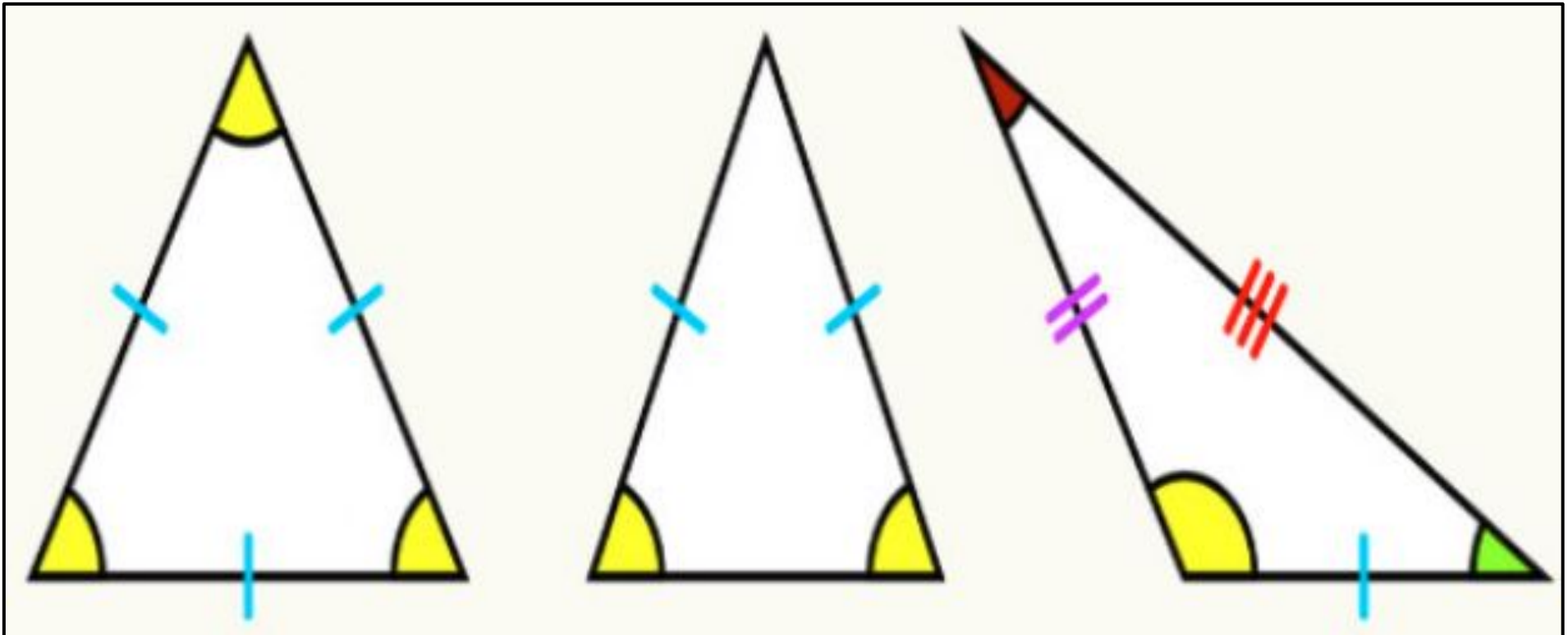
- Faça um programa que leia a nota de um aluno e escreva na tela:  
“Parabéns, muito bom” (se a nota  $\geq 80$ ); “Parabéns, aprovado” (se a nota  $\geq 70$  && nota  $< 80$ ); e, caso contrário, “Infelizmente, reprovado”

Arquivo: Exemplo1f.java



# Exercício do Comando if-else

- Faça um programa que leia três números reais representando os lados de um triângulo e informe se seu triângulo é Equilátero, Isósceles ou Escaleno



# Mais Exercícios do Comando if-else

- Leia 3 números inteiros, selecione o menor e o maior e imprima os seus respectivos valores na tela.
- Leia 10 números inteiros, selecione o maior e imprima seu valor na tela.
- Leia dois números. Se um deles for maior que 45, realize a soma dos mesmos. Caso contrário, se os dois forem maior que 20, realize a subtração do maior pelo menor, senão, se um deles for menor do que 10 e o outro diferente de 0 realize a divisão do primeiro pelo segundo. Finalmente, se nenhum dos casos solicitados for válido, mostre seu nome na tela.
- Seja uma partida de futebol, leia os números de gols do mandante e do visitante e imprima quem foi o vencedor ou se foi empate.



# Mais Exercícios do Comando if-else

- O banco do Zé abriu uma linha de crédito para os seus clientes. O valor máximo da prestação não poderá ultrapassar 40% do salário bruto. Fazer um algoritmo que permita entrar com o salário bruto e o valor da prestação e informar se o empréstimo será concedido.
- Leia dois números reais e imprima a raiz cúbica do menor e o logaritmo do menor considerando o maior como a base desse logaritmo.
- Leia 10 números inteiros, selecione o menor e o maior e imprima os seus respectivos valores na tela.

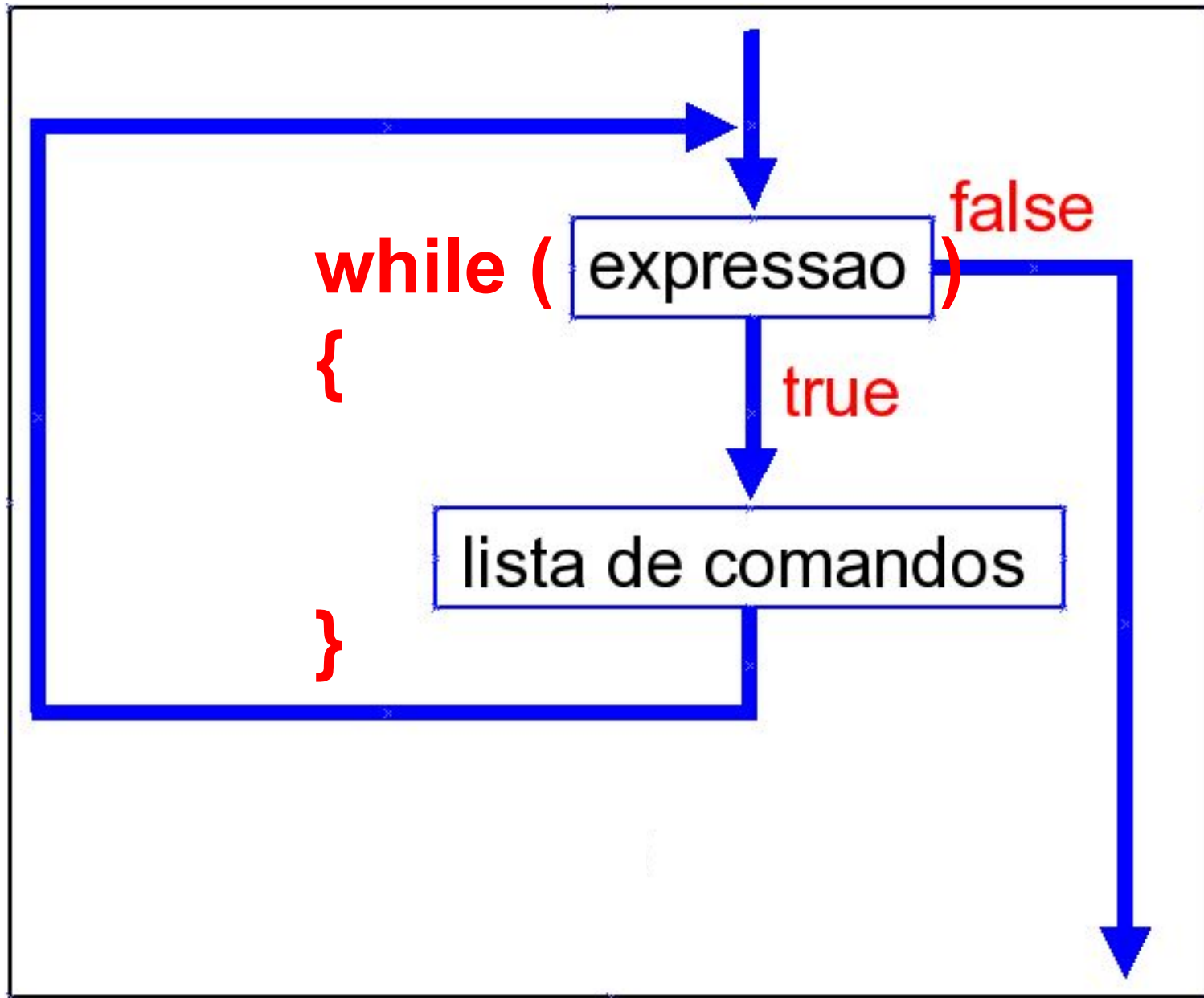


# Comandos

- Estruturas condicionais:
  - if-else
  - Operador ternário
  - switch-case
- Estruturas de repetição:
  - **while**
  - do-while
  - for
- Tratamento de exceções



## Comando while



# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;  
  
while (num < 4){  
    System.out.println(num++);  
    num += 2;  
}
```

# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;  
  
while (num < 4){  
    System.out.println(num++);  
    num += 2;  
}
```

TELA

## QUADRO DE MEMÓRIA

QUADRO DE MEMÓRIA	
num	

# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;
```

```
while (num < 4){  
    System.out.println(num++);  
    num += 2;  
}
```

TELA

## QUADRO DE MEMÓRIA

num	0
-----	---



# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;
```

```
while (num < 4){ true
```

```
    System.out.println(num++);  
    num += 2;
```

```
}
```

TELA

## QUADRO DE MEMÓRIA

num	0
-----	---

# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;  
  
while (num < 4){  
    System.out.println(num++);  
    num += 2;  
}
```

TELA

0

## QUADRO DE MEMÓRIA

num	0
-----	---

# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;  
  
while (num < 4){  
    System.out.println(num++);  
    num += 2;  
}
```

TELA

0

## QUADRO DE MEMÓRIA

num	0 1
-----	-----

# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;  
  
while (num < 4){  
    System.out.println(num++);  
    num += 2;  
}
```

TELA

0

## QUADRO DE MEMÓRIA

num	0 1 3
-----	-------

# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;
```

```
while (num < 4){ true
```

```
    System.out.println(num++);  
    num += 2;
```

```
}
```

TELA

0

## QUADRO DE MEMÓRIA

num	0 1 3
-----	-------

# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;

while (num < 4){
    System.out.println(num++);
    num += 2;
}
```

TELA

0  
3

## QUADRO DE MEMÓRIA

num	0 1 3
-----	-------

# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;

while (num < 4){
    System.out.println(num++);
    num += 2;
}
```

TELA

0  
3

## QUADRO DE MEMÓRIA

num	0 1 3 4
-----	---------

# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;

while (num < 4){
    System.out.println(num++);
    num += 2;
}
```

TELA

0

3

## QUADRO DE MEMÓRIA

num	0 1 3 4 6
-----	-----------



# Exemplo do Comando while

- Faça o quadro de memória e mostre a saída na tela para o código abaixo

```
int num = 0;  
while (num < 4){  
    System.out.println(num +);  
    num += 2;  
}
```

**false**



TELA
0
3

## QUADRO DE MEMÓRIA

QUADRO DE MEMÓRIA	
num	0 1 3 4 6

# Exemplo do Comando while

- Faça um programa que mostre na tela os 10 primeiros números pares

Arquivo: ExemploWhile01.java

# Exemplo do Comando if-else

- Faça um programa que mostre na tela o logaritmo na base 10 dos números 1 à 10

Arquivo: ExemploWhile02.java

# Exercício do Comando while

- Faça um programa que leia a nota de 5 alunos e mostre na tela a média das mesmas

Arquivo: ExercicioWhile.java

# Mais Exercícios do Comando while

- Faça um programa que mostre os 10 primeiros números inteiros positivos.
- Faça um programa que leia um número inteiro N e mostre na tela os N primeiros números inteiros ímpares.
- Faça um programa que leia um número inteiro N e mostre na tela os N primeiros números da sequência 1, 5, 12, 16, 23, 27 34.



## Mais Exercícios do Comando while

- Faça um programa que leia um número inteiro N indicando a nota máxima em uma prova P. Em seguida, leia a nota de 20 alunos (entre 0 e N) e mostre na tela: (i) a média da turma, (ii) o número de alunos cuja nota foi menor que a média da Universidade (suponha 60%) e (iii) a quantidade de alunos com conceito A (mais de 90%).
- Faça um programa que leia um número inteiro n e mostre na tela o n-ésimo termo da sequência de Fibonacci



# Comandos

- Estruturas condicionais:
  - if-else
  - Operador ternário
  - switch-case
- Estruturas de repetição:
  - while
  - do-while
  - **for**
- Tratamento de exceções



# Comando for

- Similar ao comando enquanto, contudo, ele permite: pré-comandos (início) e pós-comandos (incremento)

```
início;  
  
while (expressão) {  
    lista de comandos;  
    incremento;  
}
```

```
for (início; expressão; incremento;) {  
    lista de comandos;  
}
```



# Comando for

- Similar ao comando enquanto, contudo, ele permite: pré-comandos (início) e pós-comandos (incremento)

```
int num = 0;

while (num < 4) {
    System.out.println(num++);
    num += 2;
}
```

```
for (int num = 0; num < 4; num += 2) {
    System.out.println(num++);
}
```


```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```



```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```

```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```

## Comando for

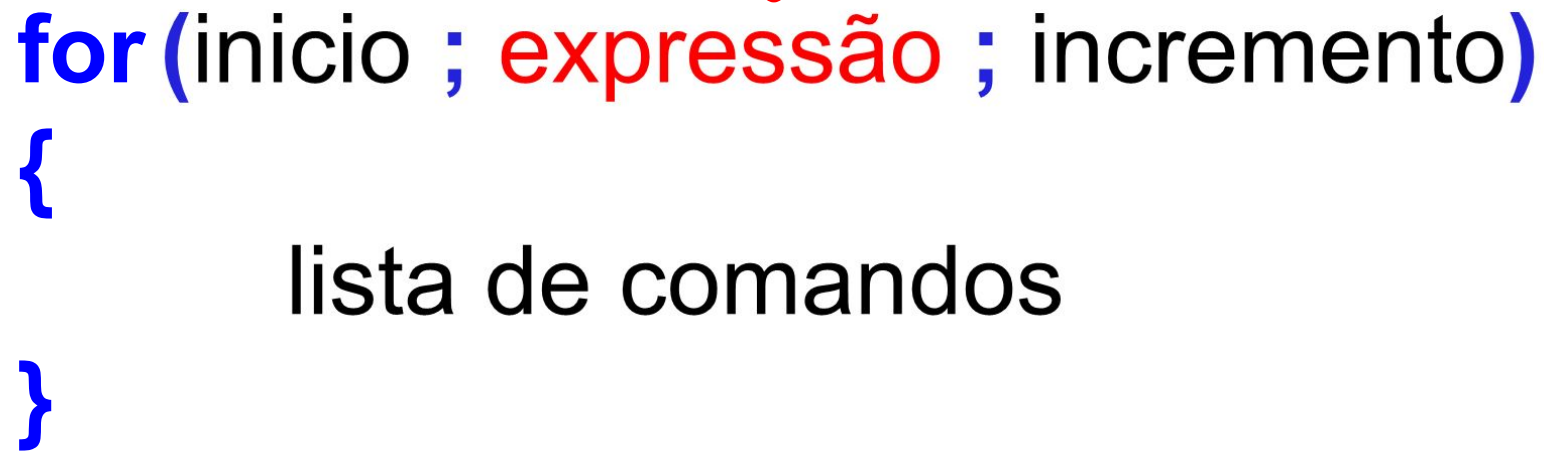


```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```

The diagram illustrates the execution flow of a for loop. A red arrow points upwards to the 'início' (start) value, another red arrow points to the right along the top of the loop header, and a third red arrow points downwards from the end of the header, indicating the start of the loop body.

```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```

## Comando for



A diagram illustrating the Java for loop syntax. The code is shown as `for (início ; expressão ; incremento)` followed by a block of code in curly braces, with the text "lista de comandos" centered inside. The word "expressão" is highlighted in red. Above the semicolon after "expressão", there are three red circles of increasing size, with a red thought bubble containing the text "true or false?" pointing to the second circle.

```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```


```
for (início ; expressão ; incremento)
{
    ↓ true
    lista de comandos
}
```



```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```


## Comando for

```
for (início ; expressão ; incremento)  
{  
    lista de comandos  
}
```



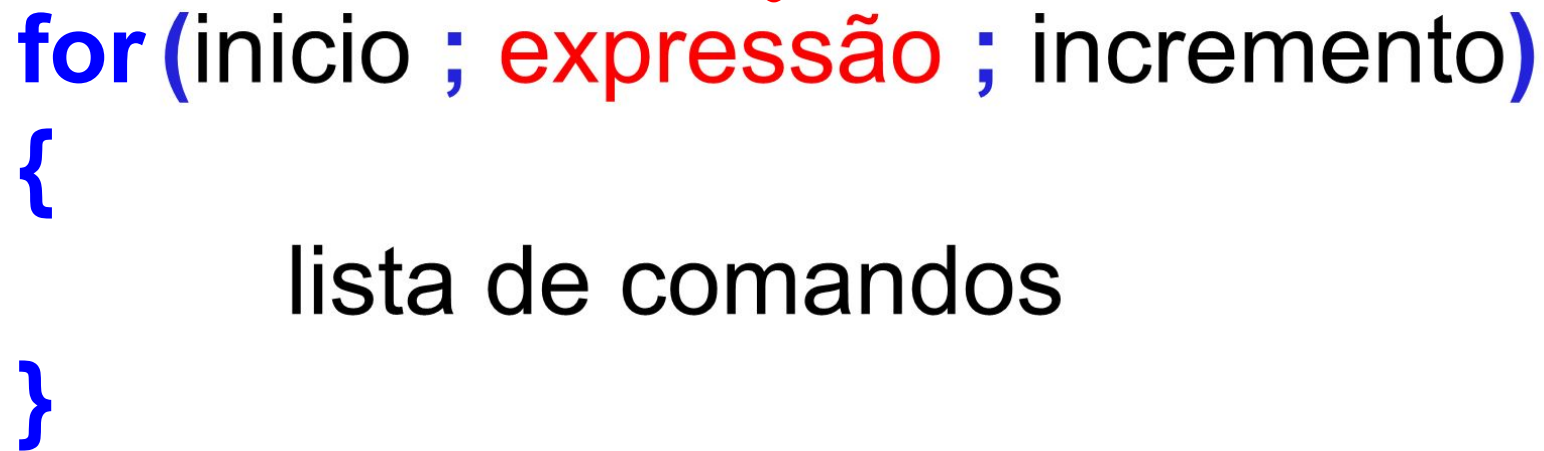
```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```

## Comando for



```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```

## Comando for



A diagram illustrating the Java for loop syntax. The code is shown as `for (início ; expressão ; incremento)` followed by a block of code enclosed in curly braces, with the text "lista de comandos" centered inside. The word "expressão" is highlighted in red. Above the semicolon after "expressão", there are three red circles of increasing size, with a red thought bubble containing the text "true or false?" pointing to the second circle.


```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```

```
for (início ; expressão ; incremento)
{
    ↓ true
    lista de comandos
}
```

```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```

## Comando for


```
for (início ; expressão ; incremento)  
{  
    lista de comandos  
}
```





```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```


## Comando for



```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```

## Comando for

```
for (início ; expressão ; incremento)
{
    lista de comandos
}
```



false

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; i <= 3 ; i++ ){  
    System.out.println(i);  
}
```

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; i <= 3 ; i++ ){  
    System.out.println(i);  
}
```

TELA

QUADRO DE MEMÓRIA

i	
---	--

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for (int i = 1 ; i <= 3 ; i++ ){  
    System.out.println(i);  
}
```

TELA

QUADRO DE MEMÓRIA

i	1
---	---

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; true i <= 3 ; i++ ) {  
    System.out.println(i);  
}
```

TELA

QUADRO DE MEMÓRIA

i	1
---	---

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; i <= 3 ; i++ ){  
    System.out.println(i);  
}
```

TELA

1

QUADRO DE MEMÓRIA

i	1
---	---



# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; i <= 3 ; i++ ){  
    System.out.println(i);  
}
```

TELA

1

QUADRO DE MEMÓRIA

i
1 2

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; true i <= 3 ; i++ ) {  
    System.out.println(i);  
}
```

TELA

1

QUADRO DE MEMÓRIA

i	1	2
---	---	---

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; i <= 3 ; i++ ){  
    System.out.println(i);  
}
```

TELA

1  
2

QUADRO DE MEMÓRIA

i	1 2
---	-----

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; i <= 3 ; i++ ){  
    System.out.println(i);  
}
```

TELA

1  
2

QUADRO DE MEMÓRIA

i	1	2	3
---	---	---	---

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; true i <= 3 ; i++ ) {  
    System.out.println(i);  
}
```

TELA

1  
2

QUADRO DE MEMÓRIA

i	1	2	3
---	---	---	---

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; i <= 3 ; i++ ){  
    System.out.println(i);  
}
```

TELA

1  
2  
3

QUADRO DE MEMÓRIA

i	1	2	3
---	---	---	---

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; i <= 3 ; i++ ){  
    System.out.println(i);  
}
```

TELA

1  
2  
3

QUADRO DE MEMÓRIA


i	1	2	3	4
---	---	---	---	---

# Exemplo do Comando for

- Como imprimir os 3 primeiros números a partir do 1?

```
for ( int i = 1 ; i <= 3 ; i++ ) {  
    System.out.println(i);  
}
```

**false**



TELA

1  
2  
3

QUADRO DE MEMÓRIA

i	1	2	3	4
---	---	---	---	---



# Exemplo do Comando for

- Faça um programa que leia a nota de 5 alunos e mostre na tela a média das mesmas usando o comando for

Arquivo: ExemploFor.java

# Exemplo do Comando for

- Faça um programa que leia a nota de 5 alunos e mostre na tela a média das notas cujo valor é maior ou igual a oitenta

Arquivo: ExemploForIf.java

- Introdução
- Primeiros Programas em Java
- *Integrated Development Environments* (IDEs) para Java
- Variáveis e Expressões
- Estruturas de Controle
- **Arrays**
- *String*



# Motivação

- **Exercício:** Faça um programa que leia  $n$  números inteiros, calcule a média desses valores e mostre aqueles que forem maiores que a média

# Motivação

- **Exercício:** Faça um programa que leia  $n$  números inteiros, calcule a média desses valores e mostre aqueles que forem maiores que a média
  - Como criamos  $n$  variáveis ( $n$  é lido do teclado)?

# Motivação

- **Exercício:** Faça um programa que leia  $n$  números inteiros, calcule a média desses valores e mostre aqueles que forem maiores que a média
  - Se fizermos a leitura de todos os valores e adicionarmos cada valor em uma nova variável, perderemos os valores lidos e não poderemos mostrar aqueles que forem maiores que a média

# Motivação

- **Exercício:** Faça um programa que leia  $n$  números inteiros, calcule a média desses valores e mostre aqueles que forem maiores que a média

```
int n, valor, media = 0;
```

```
n = sc.nextInt();
```

```
for (int i = 0; i < n; i++){  
    valor = sc.nextInt();  
    media += valor;  
}
```

```
media /= n;
```

```
//Como mostrar os elementos maiores que a média?
```

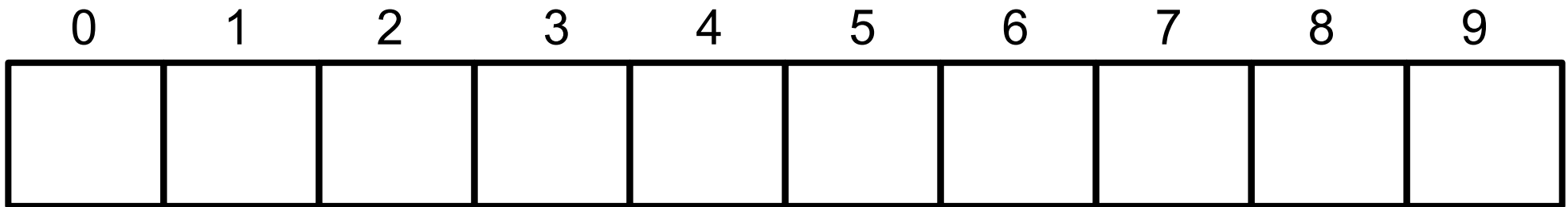
```
//E agora José?
```

- O uso de *arrays* permite a manipulação de uma grande massa de dados a partir de pequenos trechos de código



# Definição

- Conjunto de variáveis que pode ser endereçada por um índice



- Posições válidas de 0 a  $(n-1)$
- Cuidado: Frequentemente, os programadores erram e acessam posições negativas ou maiores que  $(n-1)$

# Declaração de Vetor ou Matriz

- Acrescentamos um par de colchetes ou antes ou depois do nome da variável, por exemplo:

```
int ind[ ];  
double [ ]nota;  
double matriz[ ][ ], matriz3D[ ][ ][ ];
```

# Declaração Múltipla

- **tipo** variável1[tamanho1], variável2[tamanho2], ..., variávelN[tamanhoN];

# Tamanho do Vetor ou Matriz

- Utilizamos o operador new conforme mostrado nos exemplos abaixo:

```
ind = new int[10];  
nota = new double[70];  
matriz = new double[10][20];  
matriz3D = new double[10][20][30];
```

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];

vet[0] = 5;
vet[1] = 9;
vet[2] = 312;

vet[0] = vet[1] + vet[2];
vet[1]--;

int posicao = 0;
vet[posicao] = 3;

for (int i = 0; i < 3; i++){
    vet[i] = sc.nextInt();
}

for (int i = 0; i < 3; i++){
    System.out.println(vet[i]);
}
```

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

Quantas posições  
nosso *arrays* possui?

Quais são as posições  
disponíveis?

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

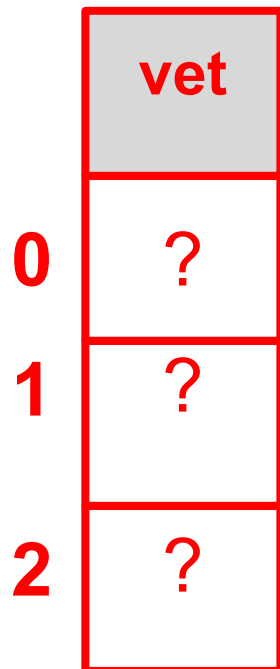
```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```



Na verdade, em Java, o valor inicial das posições de um *array* de inteiros é zero.

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

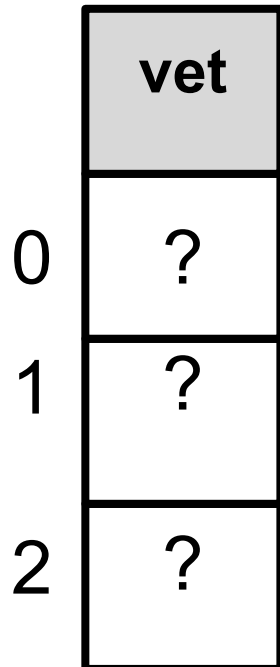
```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

O que acontece se  
acessarmos uma  
posição negativa ou  
maior que n-1?





# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

	vet
0	5
1	?
2	?

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

vet	
0	5
1	9
2	?

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

	vet
0	5
1	9
2	312

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

	vet
0	321
1	9
2	312

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

	vet
0	321
1	8
2	312

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

	vet
0	3
1	8
2	312

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){  
    vet[i] = sc.nextInt();  
}
```

```
for (int i = 0; i < 3; i++){  
    System.out.println(vet[i]);  
}
```

Cuidado: i é diferente  
de vet[i]

	vet
0	3
1	8
2	312

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){  
    vet[i] = sc.nextInt();  
}
```

```
for (int i = 0; i < 3; i++){  
    System.out.println(vet[i]);  
}
```

	vet
0	3
1	8
2	312

i
0



# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

true or false?

	vet
0	3
1	8
2	312

i
0

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i])
```

```
}
```

vet	
0	5
1	8
2	312

i
0

Usuário digita 5

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++) {
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++) {
```

```
    System.out.println(vet[i]);
```

```
}
```

vet	
0	5
1	8
2	312

i
1

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

true or false?

	vet
0	5
1	8
2	312

i
1

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i])
```

```
}
```

vet	
0	5
1	10
2	312

i
1

Usuário digita 10

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++) {
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++) {
```

```
    System.out.println(vet[i]);
```

```
}
```

vet	
0	5
1	10
2	312

i
2

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

true or false?

	vet
0	5
1	10
2	312

i
2

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i])
```

```
}
```

vet	
0	5
1	10
2	15

i
2

Usuário digita 15



# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++) {
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++) {
```

```
    System.out.println(vet[i]);
```

```
}
```

vet	
0	5
1	10
2	15

i
3

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

true or false?

vet	
0	5
1	10
2	15

i
3

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

vet	
0	5
1	10
2	15

i
0

TELA

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

true or false?

vet	
0	5
1	10
2	15

i
0

TELA

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

vet	
0	5
1	10
2	15

i
0

TELA
5

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

vet	
0	5
1	10
2	15

i
1

TELA
5

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

true or false?

vet	
0	5
1	10
2	15

i
1

TELA
5

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

vet	
0	5
1	10
2	15

i
1

TELA
5 10



# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

	vet
0	5
1	10
2	15

i
2

TELA
5 10

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

true or false?

vet	
0	5
1	10
2	15

i
2

TELA
5 10

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

	vet
0	5
1	10
2	15

i
2

TELA
5
10
15

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

vet	
0	5
1	10
2	15

i
3

TELA
5
10
15

# Acesso ao Conteúdo de uma Posição

```
int vet[ ] = new int [3];
```

```
vet[0] = 5;
```

```
vet[1] = 9;
```

```
vet[2] = 312;
```

```
vet[0] = vet[1] + vet[2];
```

```
vet[1]--;
```

```
int posicao = 0;
```

```
vet[posicao] = 3;
```

```
for (int i = 0; i < 3; i++){
```

```
    vet[i] = sc.nextInt();
```

```
}
```

```
for (int i = 0; i < 3; i++){
```

```
    System.out.println(vet[i]);
```

```
}
```

true or false?

vet	
0	5
1	10
2	15

i
3

TELA

5  
10  
15

# Index Out of Bounds

- Acesso a posições maiores que  $(n-1)$

```
int n = 10;  
int vet1[] = new int[n];  
...  
for (int i = n; i >= 0; i--){  
    vet1[i] = sc.nextInt();  
}
```

# Index Out of Bounds

- Acesso a posições negativas

```
int n = 10;  
int vet1[] = new int[n];  
...  
for (int i = n-1; i >= 0; i--){  
    vet1[i-1] = sc.nextInt();  
}
```

# Exemplo sobre *Arrays*

- Faça um programa que leia a nota e o nome de 5 alunos e mostre na tela o nome daqueles que ficaram acima da média do grupo

Arquivo: ExemploArray.java



# Exemplo sobre *Arrays*

- Faça um programa que leia  $n$  números e os armazene em um *array*
- ... e, em seguida, mostre cada número na tela

# Exemplo sobre *Arrays*

- Faça um programa que leia  $n$  números e os armazene em um *array*

```
int n = 10;  
int vet[ ] = new int[n];  
  
for (int i = 0; i < n; i++) {  
    vet[i] = sc.nextInt();  
}
```

- ... e, em seguida, mostre cada número na tela

```
for (int i = 0; i < n; i++) {  
    System.out.println("vet[" + i + "] = " + vet[i]);  
}
```

# Mais Exercícios sobre *Arrays*

- Faça um programa que leia  $n$  números e mostre quais deles são maiores que a média
- Faça um programa que leia  $n$  números e mostre a soma do  $i$ -ésimo com o  $(2*i+1)$ -ésimo termo até que  $(2*i+1) < n$



# Exemplos

- Faça um programa que leia os elementos de um array de tamanho  $n$  e mostre o maior elemento do array

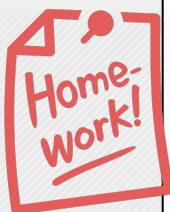
```
int n = sc.nextInt(), maior;
int vet[] = new int[n];

for (int i = 0; i < n; i++){
    vet[i] = sc.nextInt();
}

maior = vet[0];

for (int i = 1; i < n; i++){
    if (vet[i] > maior){
        maior = vet[i];
    }
}

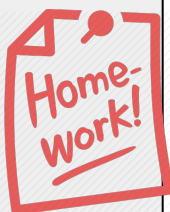
System.out.println("Maior :" + maior);
```



# Exemplos

- Faça um programa que leia os elementos de um array de tamanho  $n$  e mostre o maior e o menor elementos do array

```
int n = sc.nextInt(), maior, menor;
int vet[] = new int[n];
for (int i = 0; i < n; i++){
    vet[i] = sc.nextInt();
}
menor = maior = vet[0];
for (int i = 1; i < n; i++){
    if (vet[i] > maior){
        maior = vet[i];
    } else if (vet[i] < menor){
        menor = vet[i];
    }
}
System.out.println("Maior: " + maior + "\nMenor: " + menor);
```



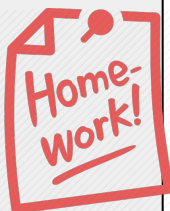
# Exemplos

- Faça um programa que leia dois *arrays* e mostre na tela seus elementos intercalados

```
int tam1, tam2, tamMaior; ler tam1, tam2;
if (tam1 > tam2) { tamMaior = tam1; } else { tamMaior = tam2; }

int vet1[ ] = new int[tam1]; int vet2[ ] = new int[tam2];
//ler vet1, vet2;

for (int i = 0; i < tamMaior; i++){
    if(i < tam1){
        System.out.println(vet1[i]);
    }
    if(i < tam2){
        System.out.println(vet2[i]);
    }
}
```



## Exemplos

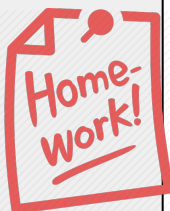
- Faça um programa que leia um array, encontre e mostre na tela os maior e o menor elementos

```
int n = sc.nextInt(), maior, menor;
int vet1[ ] = new int[n];

//ler vet1

if (vet[0] > vet[1]){ maior = vet[0]; menor = vet[1]; }
else                { maior = vet[1]; menor = vet[0]; }

for (int i = 2; i < n; i++){
    if (vet[i] > maior){
        maior = vet[i];
    } else if(vet[i] < menor){
        menor = vet[i];
    }
}
System.out.println("Maior: " + maior + "\nMenor: " + menor);
```



# Mais Exercícios sobre *Arrays*

- Faça um programa que leia os elementos de um array de tamanho  $n$  e mostre a **posição** do menor elemento do array
- Faça um programa que receba um array e ordene os elementos desse array
- Faça um programa que leia dois arrays contendo números inteiros e mostre a união e a intercessão entre os elementos desses arrays





# Mais Exercícios sobre *Arrays*

- Declare um vetor com contendo os elementos 10, 5, 8, 2 e 8. Em seguida, mostre os elementos contidos no *array*
- Faça um programa para ler a nota de cinco alunos, calcular e mostrar: a soma e a média das mesmas e a menor nota
- Faça um programa para ler N números inteiros, calcular a média deles e mostrar aqueles que forem maiores que a média



# Mais Exercícios sobre *Arrays*

- Faça um programa para ler um número inteiro  $N$  e  $N$  elementos de um *array*. Em seguida, se  $N$  for par mostrar na tela a soma do 1o e 2o elemento, 3o e 4o, ...
- Faça um programa para ler um número inteiro  $N$  e  $N$  elementos de um *array*. Em seguida, encontre a posição do menor elemento
- Faça um programa para ler um número inteiro  $N$  e  $N$  elementos de um *array*. Em seguida, calcular e mostrar o número de elementos pares e o de divisíveis por três.



# Mais Exercícios sobre *Arrays*

- Faça um programa que leia N números inteiros e mostre na tela a soma do primeiro e do último, a do segundo e do penúltimo, a do terceiro e do antepenúltimo, ...
- Faça um programa que leia N números inteiros e mostre na tela a soma daqueles que forem múltiplos de três



# Mais Exercícios sobre *Arrays*

- Faça um programa para ler dois vetores contendo N elementos cada. Em seguida, mostre os elementos de forma intercalada
- Faça um programa para ler um vetor contendo N elementos e outro contendo M elementos. Em seguida, mostre os elementos de forma intercalada
- Faça um programa para ler um vetor contendo N elementos inseridos em uma ordem qualquer e mostre os elementos de forma ordenada



# Mais Exercícios sobre Matrizes

- Faça um programa que leia os elementos de uma matriz com L linhas e C colunas e mostre na tela os elementos da matriz em formato de grid
- Faça um programa que leia os elementos de uma matriz com L linhas e C colunas e mostre na tela os elementos da matriz Transposta
- Faça um programa que leia duas matrizes com os mesmos números de linhas e colunas, faça a soma das mesmas e imprima na tela a matriz resultante



# Mais Exercícios sobre Matrizes

- Faça um programa que leia os elementos de uma matriz quadrada com  $N$  linhas e  $N$  colunas e mostre as diagonais principal e secundária
- Faça um programa que leia os elementos de uma matriz e mostre a média dos elementos
- Faça um programa que leia os elementos de uma matriz e mostre a média dos elementos de cada linha
- Faça um programa que leia os elementos de uma matriz e mostre a média dos elementos de cada coluna



- Introdução
- Primeiros Programas em Java
- *Integrated Development Environments* (IDEs) para Java
- Variáveis e Expressões
- Estruturas de Controle
- *Arrays*
- ***String***



- São vetores de caracteres cujas posições são acessadas usando o método *char charAt(posição)*
- Seus tamanhos são acessíveis usando o método *int length()*
- Não existe o método *char charSet(posição)*
- Possui outros métodos



# Exercício com Strings

- Faça um programa que leia uma *String* e um caractere e mostre na tela quantas vezes esse caractere aparece na *String*

Arquivo: ExemploString.java

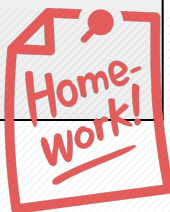
# Mais Exercícios sobre Strings

- Faça um programa que leia uma string e mostre se ela é composta apenas por dígitos

```
String str = sc.nextLine();
boolean resp = true;

for (int i = 0; i < str.length(); i++){
    if ((str.charAt(i) >= '0' && str.charAt(i) <= '9') == false){
        resp = false;
        i = str.length();
    }
}

if (resp == true) {
    System.out.println("Parabéns");
} else {
    System.out.println("Não número");
}
```



# Mais Exercícios sobre Strings

- Faça um programa que leia uma string e a converta para número inteiro

```
String str = sc.nextLine();
int num = 0, tmp;

for (int i = 0; i < str.length(); i++){
    tmp = (int)(str.charAt(i) - 48);
    tmp *= (int)Math.pow(10, str.length() - i - 1);
    num += tmp;
}
System.out.println(num);
```



# Mais Exercícios sobre Strings

- Faça um programa que leia uma string e mostre se ela é um palíndromo

```
String str = sc.nextLine(); // "ROMA E AMOR", "ABCDDCBA", "ABXDDCBA"
boolean resp = true;

for (int i = 0; i < str.length()/2; i++){
    if (str.charAt(i) != str.charAt(str.length() - i - 1)){
        resp = false;
        i = str.length();
    }
}
System.out.println(resp);
```



# Mais Exercícios sobre Strings

- Faça um programa para ler uma palavra. Em seguida, mostre o número de caracteres da mesma e seu número de caracteres maiúsculos
- Faça um programa para ler uma palavra. Em seguida, mostre a primeira ocorrência da letra A
- Faça um programa para ler uma palavra. Em seguida, mostre os números de caracteres, letras, não letras, vogais e consoantes.

