

Curso

# Algoritmos & Programação em JAVA

Atualizado até o Java 21 & Eclipse 2023-09



**Prof. Msc. Antonio B. C. Sampaio Jr**  
ENGENHEIRO DE SOFTWARE & PROFESSOR

@abctreinamentos  
@amazoncodebr

[www.abctreinamentos.com.br](http://www.abctreinamentos.com.br)  
[www.amazoncode.com.br](http://www.amazoncode.com.br)



# CONTEÚDO PROGRAMÁTICO



- UNIDADE 1 – INTRODUÇÃO
- UNIDADE 2 – CONTRUÇÃO DE ALGORITMOS
- UNIDADE 3 – ESTRUTURAS DE SELEÇÃO
  - Primeiro Projeto JAVA no Eclipse
  - Fluxos de Execução
    - Seleção Simples
    - Correção Raiz Quadrada [NOVO]
    - Seleção Composta

# CONTEÚDO PROGRAMÁTICO



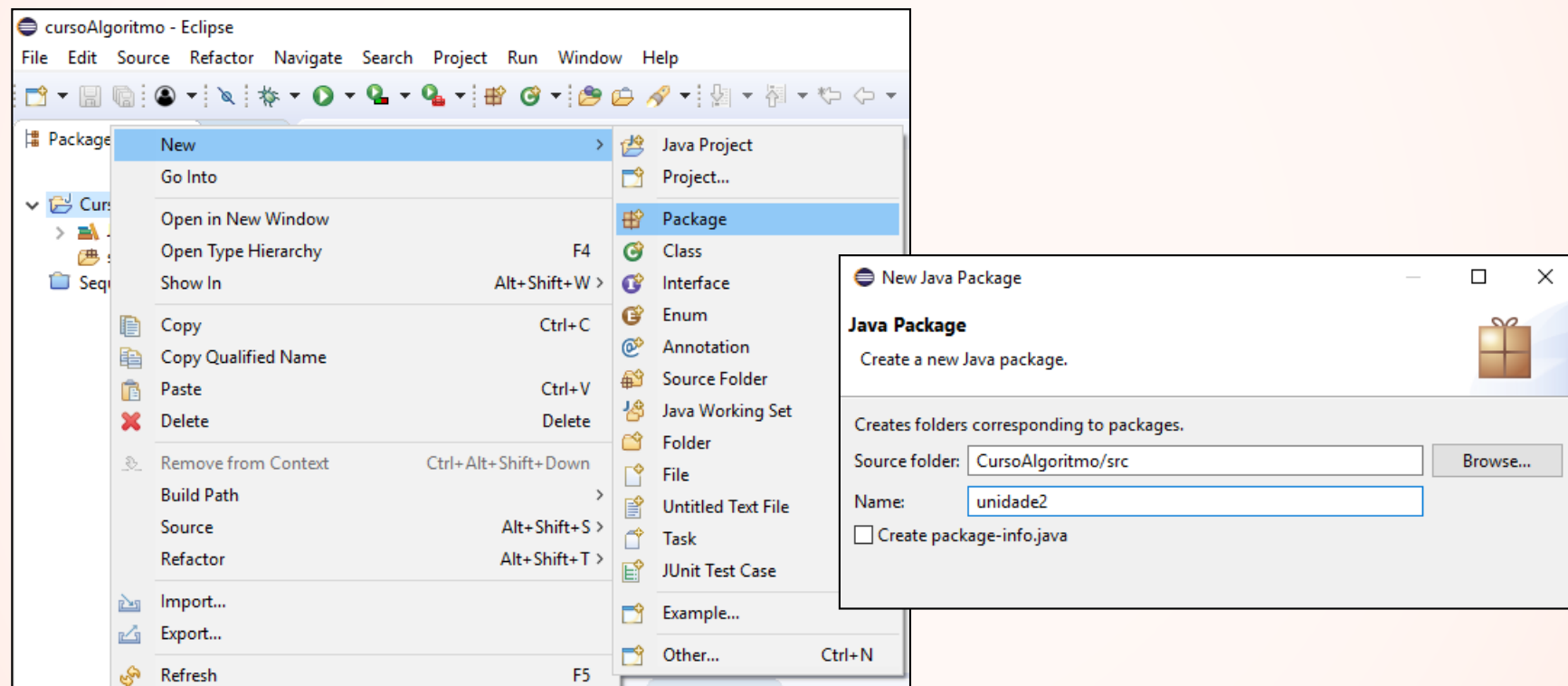
- UNIDADE 3 – ESTRUTURAS DE SELEÇÃO (Continuação)
  - Fluxos de Execução
    - Seleção Encadeada
    - Seleção de Múltipla Escolha
    - Switch Expressions [NOVO]

## UNIDADE 3

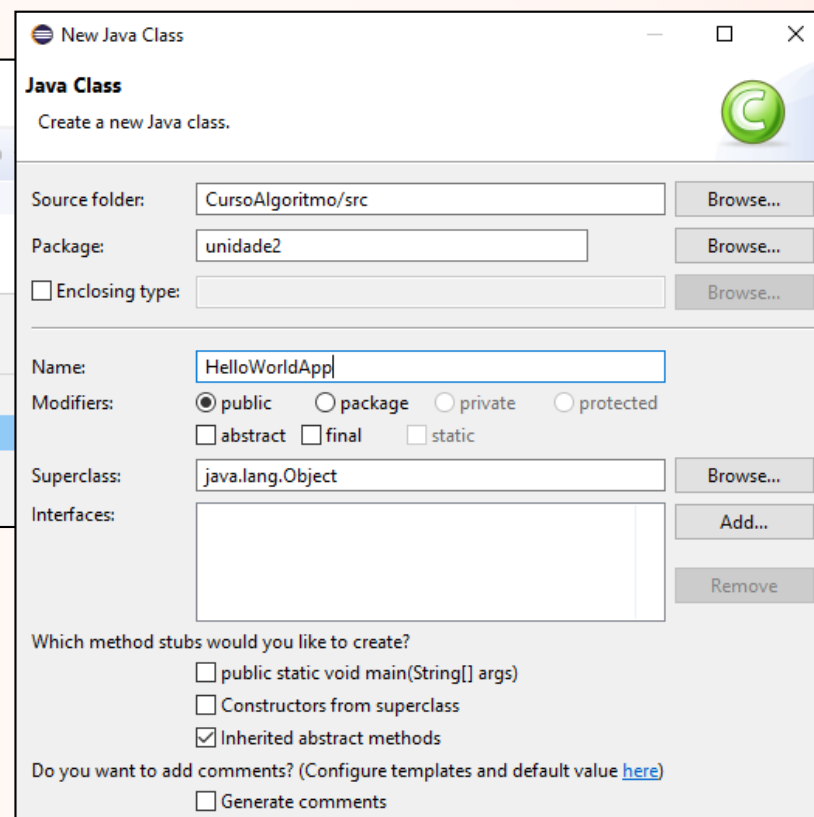
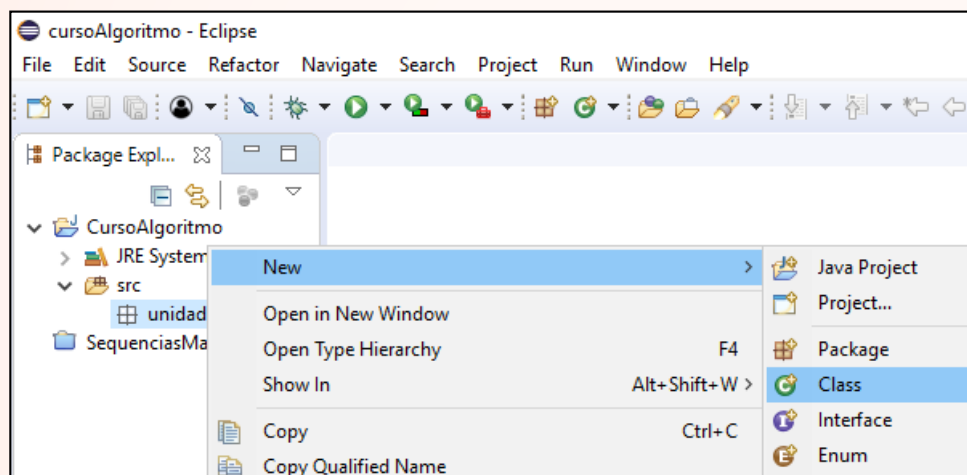
# ESTRUTURAS DE SELEÇÃO

# **Primeiro Projeto JAVA no Eclipse**

# Primeiro Pacote JAVA no Eclipse



# Primeira Classe JAVA no Eclipse

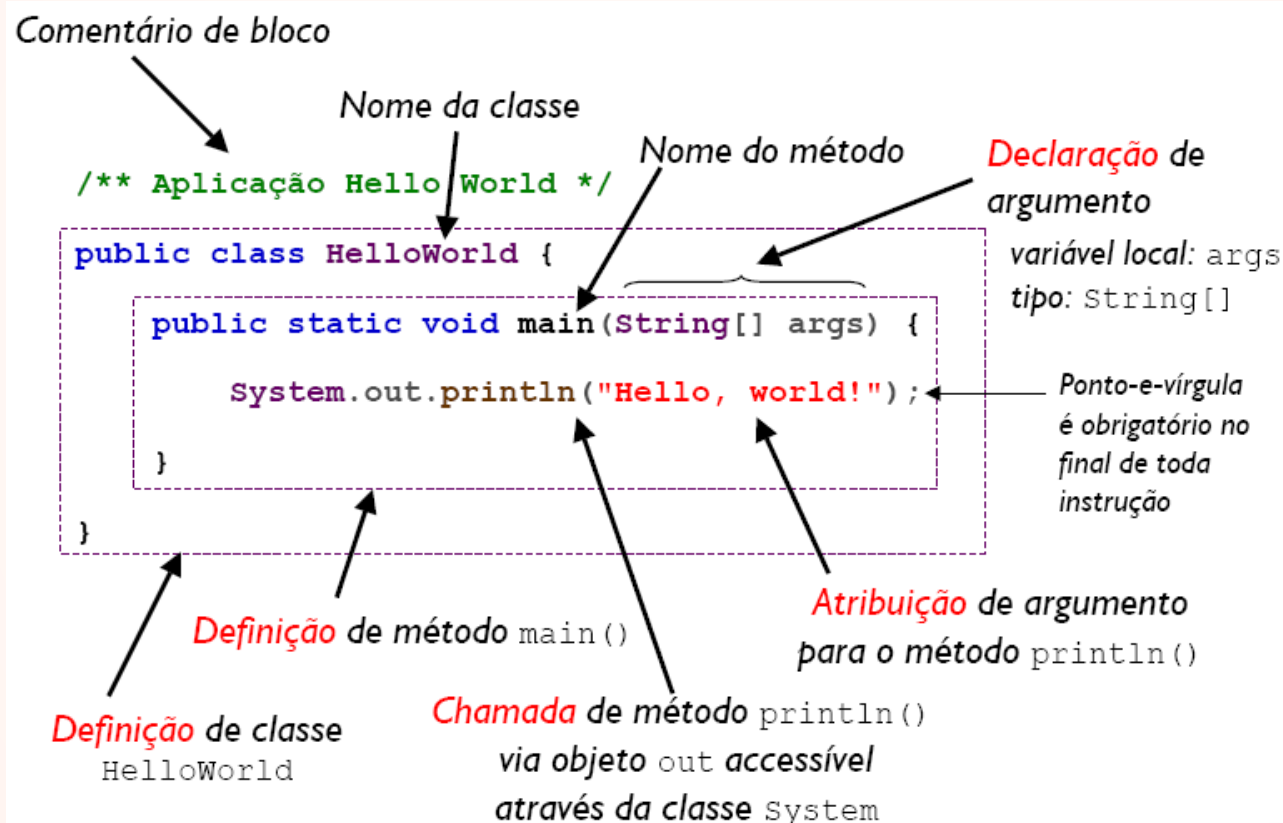


# Primeira Classe JAVA no Eclipse

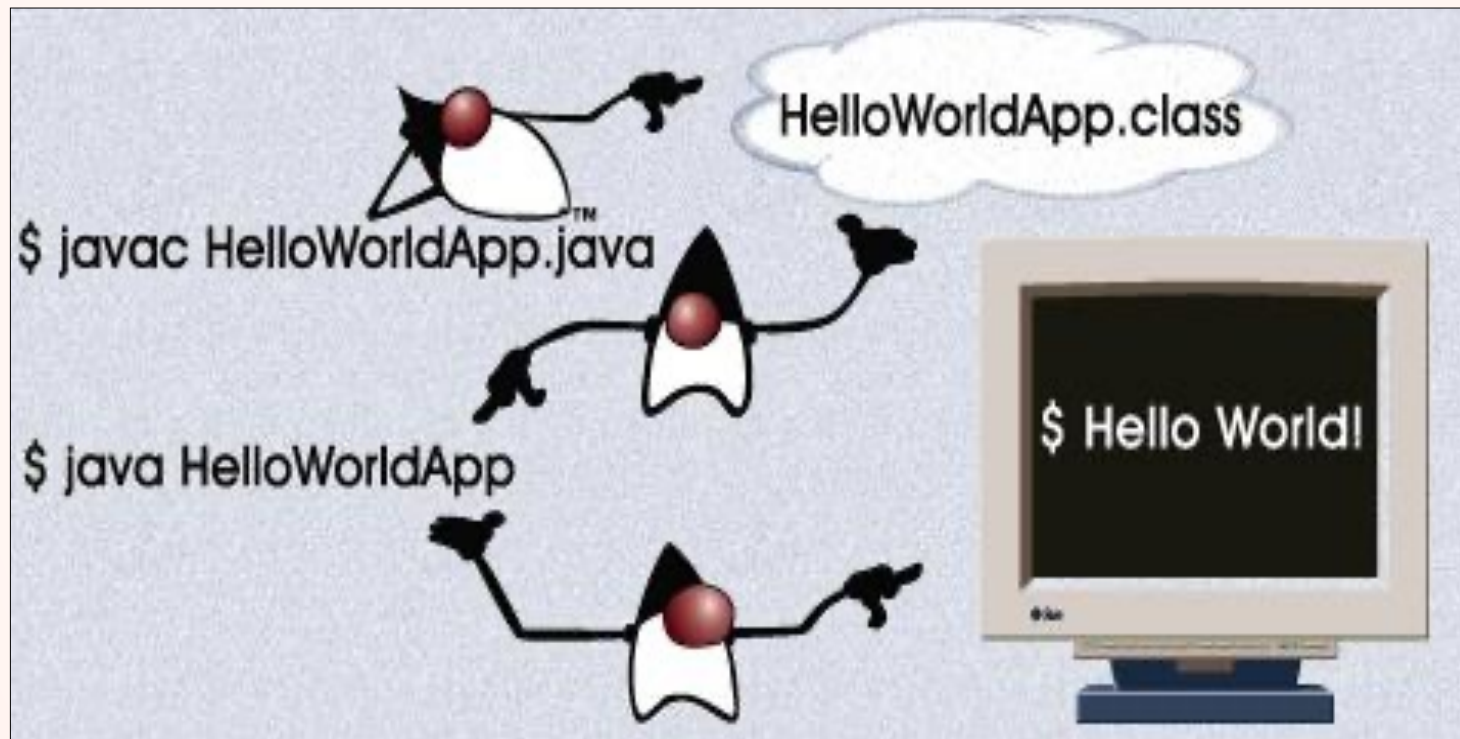
```
/** Primeiro Programa Java */  
package unidade2;  
  
class HelloWorldApp  
{  
    public static void main(String arg[])  
    {  
        System.out.println("Hello World!");  
    }  
}
```



# Análise da Primeira Classe JAVA



# Execução da Primeira **Classe JAVA**



## Sem **javac**

- A partir do Java 11, é possível executar classes Java diretamente sem a necessidade de compilar explicitamente com o javac e, em seguida, executar com o java.
- Isso só se tornou possível graças à introdução de um novo recurso chamado "Execução de Código-Fonte Direto" (ou "Single-File Source-Code Execution") no Java 11. Esse recurso permite executar programas Java contidos em um único arquivo fonte diretamente com o comando java, sem a necessidade de compilação prévia.

```
java MeuPrograma.java
```

- Esse recurso é mais adequado para programas simples ou pequenos. Para projetos maiores, ainda é recomendável usar o processo de compilação tradicional com javac e, em seguida, executar o bytecode resultante com java.

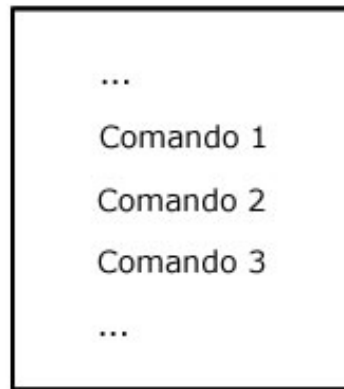
# Exercícios

- 1) Criar as classes **EscreveVariaveis**, **LeituraEscrita** e **CalculaDobro** no ambiente **Eclipse**.
- 2) Criar as classes **Jantar**, **CalculoHoras**, **Numero**, **JantarPizza**, **OpcoesPagamento**, **MediaAritmetica**, **Locadora**, **RevendaVeiculos** e **LanHouse**.

# Fluxos de Execução

# Fluxos de Execução

- No fluxo “normal” de execução das instruções (entrada/processamento/saída) de um algoritmo, cada comando só é executado após a finalização do comando anterior, seguindo um fluxo sequencial do **Início** até o **Fim** do algoritmo.



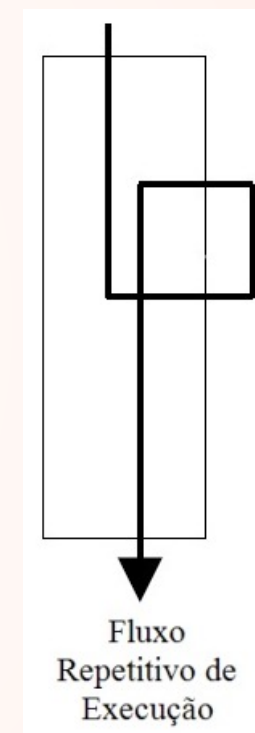
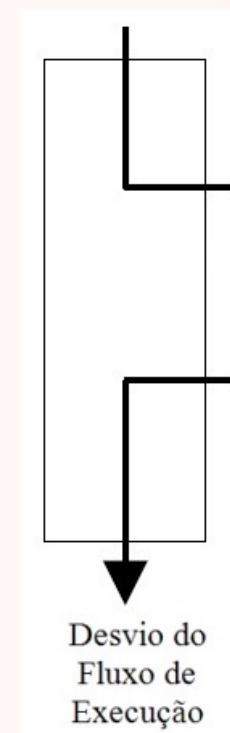
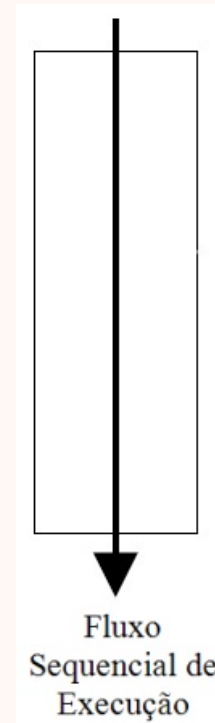
Trecho sequencial de um algoritmo



Trecho sequencial de um fluxograma

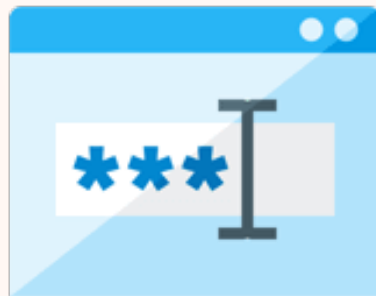
# Fluxos de Execução

- Contudo, o fluxo de execução das instruções pode ser alterado de acordo com o resultado da avaliação de uma ou mais condições.
- Por exemplo, se a média aritmética do aluno for igual ou superior a cinco (5.0), ele será aprovado e irá cursar o próximo ano letivo; caso contrário, ele terá que repetir o ano letivo.



## Desvio do Fluxo de **Execução - Condições**

- Na construção de Algoritmos, determinadas instruções só deverão ser executadas se uma determinada **CONDIÇÃO FOR SATISFEITA!**
- Por exemplo, para acessar determinado sistema de informação, é imperativo o fornecimento de dados pessoais com uma senha ou com os dados biométricos.



Conhecimento



Posse

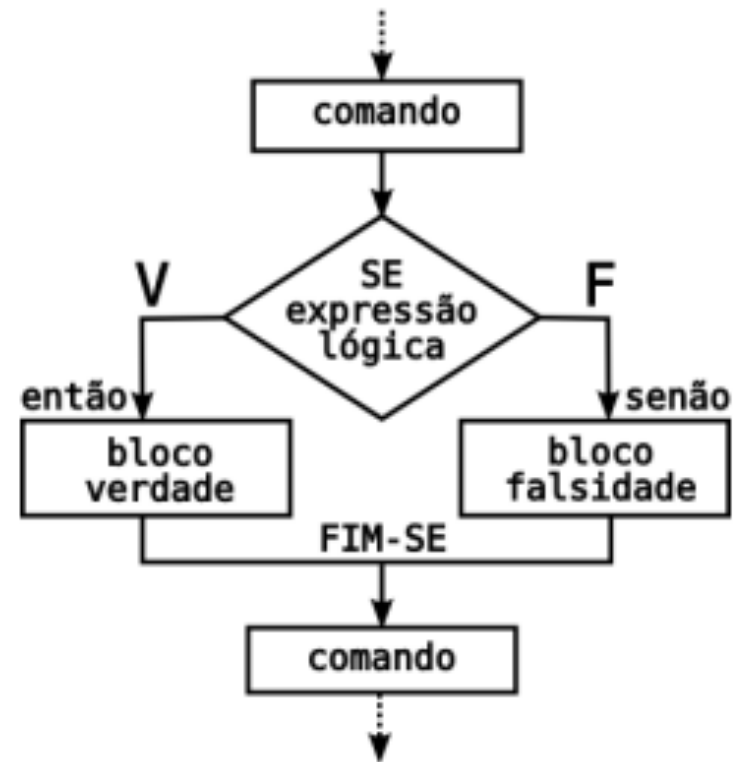


Fator Pessoal



## Estruturas de Seleção

- Uma **Estrutura de Seleção** permite a escolha de um grupo de ações (instruções) a serem executadas quando determinadas **Condições** (representadas por expressões lógicas ou relacionais) forem ou não satisfeitas.
- **<Condição>** é qualquer expressão cujo resultado seja **Verdadeiro** ou **Falso**.



# Tipos de Estruturas de **Seleção**

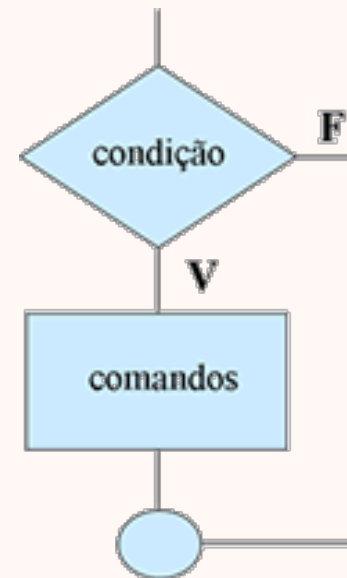
- Podem ser de 04 Tipos:
  - **SELEÇÃO SIMPLES**
  - **SELEÇÃO COMPOSTA**
  - **SELEÇÃO ENCADEADA**
  - **SELEÇÃO DE MÚLTIPLA ESCOLHA**

# Seleção Simples

# Seleção Simples

- Quando é necessário testar uma determinada condição antes de se executar uma ação.

```
Se <Condição>  
  Início  
    Passo (1) ;  
    Passo (2) ;  
    Passo (3) ;  
    ...  
    Passo (N) ;  
  Fim
```



# Seleção Simples

- EXEMPLO:

```
/** Algoritmo Média */  
Algoritmo CalculaMedia  
  Declaração de Variáveis  
  ...  
  INÍCIO  
    media ← (n1+n2+n3+n4)/4;  
    Se (media < 5)  
      escreva ("Aluno Reprovado!");  
  FIM.
```

# Seleção Simples

- EXEMPLO:

```
/** Algoritmo NúmeroPar */  
Algoritmo NumeroPar  
  Declaração de Variáveis  
  ...  
  INÍCIO  
  ...  
  Se (numero%2 == 0)  
    escreva ("Este número é Par!");  
  FIM.
```

## Seleção **Simples** em JAVA

```
if (expressão booleana)  
    instrução_simples;
```

```
if (expressão booleana) {  
    instruções  
}
```

# Seleção Simples em JAVA

```
package unidade3;
class Aluno
{
    public static void main(String arg[])
    {
        double media;
        double n1, n2, n3, n4;
        Scanner scn = new Scanner(System.in);
        n1 = scn.nextDouble(); n2 = scn.nextDouble();
        n3 = scn.nextDouble(); n4 = scn.nextDouble();
        media = (n1+n2+n3+n4)/4;
        if (media < 5)
            System.out.println("Aluno Reprovado!");
    }
}
```



# Seleção Simples em JAVA

```
package unidade3;  
class Numero  
{  
    public static void main(String arg[])  
    {  
        Scanner scn = new Scanner(System.in);  
        int numero = scn.nextInt();  
        if (numero%2 == 0)  
            System.out.println("Este número é Par!");  
    }  
}
```

# Exercícios

- 1) Implementar as classes **Aluno** e **Numero**.
- 2) Escrever um programa para calcular as raízes de uma equação do 2º grau.
- 3) Elaborar um programa que leia um número e calcule a sua raiz exata.

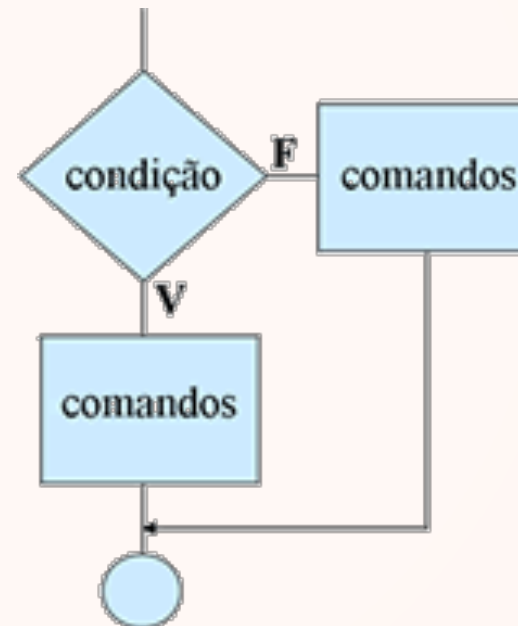
# Correção Raiz Quadrada

# Seleção Composta

# Seleção Composta

- Quando ao se testar uma determinada condição, há duas alternativas que dependem do seu resultado.

```
Se <Condição>  
  Início  
    Passo (1) ;  
    Passo (N) ;  
  Fim  
Senão  
  Início  
    Passo (1) ;  
    Passo (N) ;  
  Fim
```



# Seleção Composta

- EXEMPLO:

```
/** Algoritmo Média */  
Algoritmo CalculaMedia  
  Declaração de Variáveis  
  ...  
  INÍCIO  
    media ← (n1+n2+n3+n4)/4;  
    Se (media < 5)  
      escreva ("Aluno Reprovado!");  
    Senão  
      escreva ("Aluno Aprovado!");  
  FIM.
```

# Seleção Composta

- EXEMPLO:

```
/** Algoritmo NúmeroPar **/  
Algoritmo NumeroPar  
Declaração de Variáveis  
...  
INÍCIO  
...  
Se (numero%2 == 0)  
    escreva ("Este número é Par!");  
Senão  
    escreva ("Este número é Ímpar!");  
FIM.
```

# Seleção **Composta** em JAVA

```
if (expressão booleana) {  
    instruções  
} else {  
    instruções  
}
```



## Seleção Composta em JAVA

```
package unidade3;
class Aluno {
    public static void main(String arg[])
    {
        double media;
        double n1, n2, n3, n4;
        Scanner scn = new Scanner(System.in);
        n1 = scn.nextDouble(); n2 = scn.nextDouble();
        n3 = scn.nextDouble(); n4 = scn.nextDouble();
        media = (n1+n2+n3+n4)/4;
        if (media < 5)
            System.out.println("Aluno Reprovado!");
        else
            System.out.println("Aluno Aprovado!");
    }
}
```

# Seleção Composta em JAVA

```
package unidade3;
class Numero
{
    public static void main(String arg[])
    {
        Scanner scn = new Scanner(System.in);
        int numero = scn.nextInt();
        if (numero%2 == 0)
            System.out.println("Este número é Par!");
        else
            System.out.println("Este número é Ímpar!");
    }
}
```

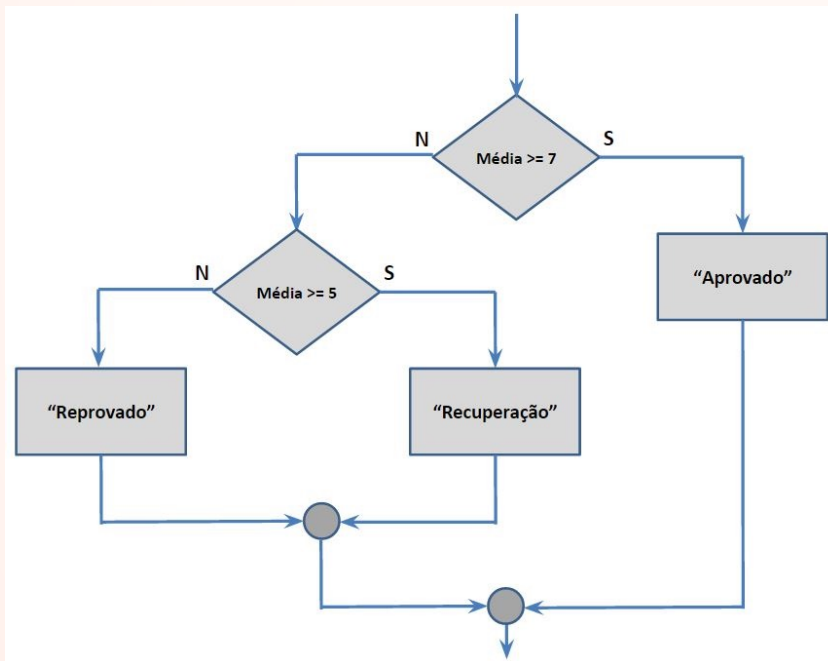
## Exercícios

- 1) Refatorar as classes **Aluno** e **Numero**. No caso da classe **Aluno**, acrescentar que o aluno para ser aprovado também terá que ter frequência superior a 75%.
- 2) Refatorar o programa para calcular as raízes de uma equação do 2º grau. Caso delta seja menor do que zero, exibir a mensagem **“Não existem raízes reais”**.
- 3) Refatorar o programa que lê um número e calcula a sua raiz quadrada. Caso ela seja exata, informar o valor; caso contrário, exibir a mensagem **“Não há raiz exata!”**.
- 4) Elaborar um programa que leia a altura e o sexo de uma pessoa e informe qual é o seu peso ideal. Utilizar as seguintes fórmulas:
  - Peso ideal para homens:  $(72.7 * h) - 58$ ;
  - Peso ideal para mulheres:  $(62.1 * h) - 44.7$ ;

# Seleção Encadeada

# Seleção Encadeada

- Quando várias seleções são agrupadas a fim de testar um conjunto de possibilidades.



```
Se <Condição1>  
  Início  
    Se <Condição11>  
      Passo1;  
    Senão  
      Passo2;  
  Fim  
Senão  
  Início  
    Se <Condição2>  
      Passo1;  
    Senão  
      Passo2;  
  Fim
```

# Seleção Encadeada

```
INÍCIO
  Se (media >= 7)
    escreva ("Aluno Aprovado!");
  Senão
    Início
      Se (media >= 5)
        escreva ("Aluno em Recuperação!");
      Senão
        escreva ("Aluno Reprovado!");
    Fim
FIM.
```

## Seleção Encadeada em JAVA

```
if (expressão booleana) {  
    instruções  
} else if (expressão booleana) {  
    instruções  
} else {  
    instruções  
}
```

# Seleção Encadeada em JAVA

```
{  
    if (media >= 7)  
        System.out.println("Aluno Aprovado!");  
    else  
    {  
        if (media >= 5)  
            System.out.println("Aluno em Recuperação!");  
        else  
            System.out.println("Aluno Reprovado!");  
    }  
}
```



## Exercícios

- 1) Dado o algoritmo a seguir, responda o que se pede:
- a) Se A = Verdade, B = Verdade, C = Falsidade, quais comandos serão executados?
  - b) Se A = Falsidade, B = Verdade, C = Falsidade, quais comandos serão executados?
  - c) Se A = Falsidade, B = Verdade, C = Verdade, quais comandos serão executados?
  - d) Quais são os valores de A, B, C para que somente o comando C5 e C6 sejam executados?

```
Início
  lógico: A, B, C;
  Se (A)
    C1;
  Senão
    Início
      Se (B)
        Início
          Se (C)
            C2;
          Senão
            Início
              C3;
              C4;
            Fim
          Fim
        C5;
      Fim
    C6;
  Fim
```

## Exercícios

2) Escreva um programa que leia uma nota (0 a 100) e escreva o conceito associado.

**[90,100] “Excelente”**

**[70,90[ “Bom”**

**[50, 70[ “Regular”**

**[0,50[ “Insuficiente”**

3) Escreva um programa que leia a idade de um nadador e classifique-o em uma das seguintes categorias:

**- Até 5 anos [Infantil A]; de 6 a 8 anos [Infantil B]; de 9 até 11 [Infantil C]; de 12 até 13 anos [Juvenil A]; de 14 até 17 anos [Juvenil B]; Acima de 18 anos [Adulto].**

4) Escreva um programa que leia 03 valores (A, B, C) e verifique se esses valores podem ser lados de um triângulo. Em caso afirmativo, informar qual é o tipo de triângulo (escaleno, equilátero e isósceles).

## Exercícios

5) Escreva um programa **CaixaEletrônico** que leia o valor de saque solicitado pelo cliente e informe o número mínimo de notas de R\$100, R\$50 e R\$10 necessárias para concluir este saque.

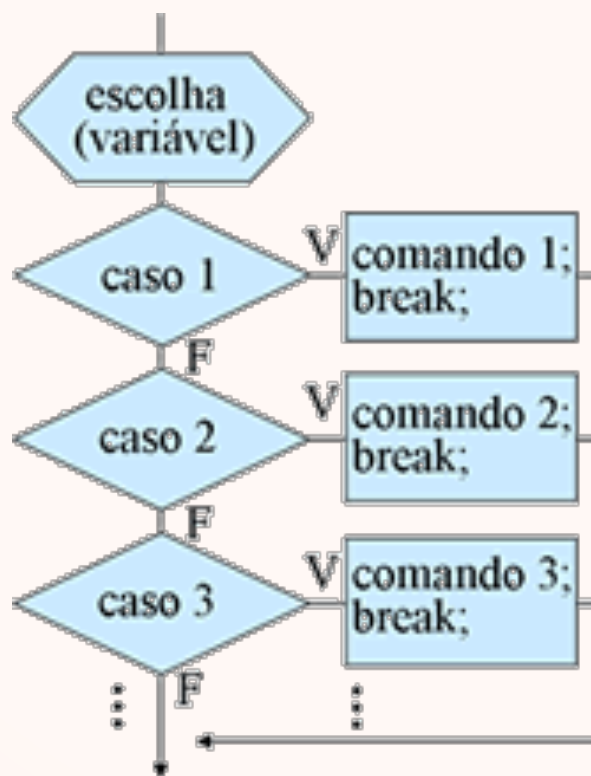
# **Seleção de Múltipla Escolha**

## Seleção de **Múltipla Escolha**

- Quando se tem várias estruturas de seleção aninhadas, é uma solução elegante que pode ser adotada.

```
leia(valor_real);  
leia(tipo_moeda);  
Escolha (tipo_moeda)  
  Início  
    Caso (moeda == 1):  
      escreva ("O valor em convertido em Libras é ", valor_real/5);  
    Caso (moeda == 2):  
      escreva ("O valor em convertido em Dólar é ", valor_real/4);  
    Caso (moeda == 3):  
      escreva ("O valor em convertido em Euro é ", valor_real/4.4);  
  Fim
```

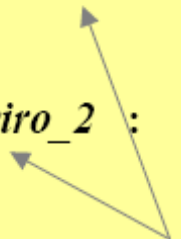
# Seleção de **Múltipla Escolha**



## Seleção de **Múltipla Escolha em JAVA**

```
switch (seletor_inteiro) {  
    case valor_inteiro_1 :  
        instruções;  
        break;  
    case valor_inteiro_2 :  
        instruções;  
        break;  
    ...  
    default:  
        instruções;  
}
```

*uma constante inteira (inclui char)*



```
switch (letra) {  
    case 'A' :  
        System.out.println("A");  
        break;  
    case 'B' :  
        System.out.println("B");  
        break;  
    ...  
    default:  
        System.out.println("?");  
}
```

## Seleção de **Múltipla Escolha** em JAVA

```
switch (tipo_moeda) {  
    case '1':  
        System.out.println ("O valor em convertido em Libras é ", valor_real/5);  
        break;  
    Caso '2':  
        System.out.println("O valor em convertido em Dólar é ", valor_real/4);  
        break;  
    Caso '3':  
        System.out.println("O valor em convertido em Euro é ", valor_real/4.4);  
        break;  
    default:  
        System.out.println("Opção digitada incorreta!");  
}
```



# Exercícios

- 1) Escreva um programa que implemente a classe **Moeda**.
- 2) Escreva um programa que calcule o que deve ser pago por um produto, considerando a tabela abaixo:

Código	Condição de Pagamento
1	À vista em dinheiro ou cheque, recebe 10% de desconto
2	À vista no cartão de crédito, recebe 5% de desconto
3	Em 2 vezes sem juros
4	Em 3 vezes com juros de 10%

# Switch Expressions

# Switch Expressions

- A partir do Java 12, introduziu-se a capacidade de usar o switch como uma expressão, permitindo que ele retorne um valor que pode ser atribuído a uma variável ou usado em outras expressões.

```
JOptionPane.showMessageDialog(null, switch(tipo_moeda)
{
    case 1 -> valor_convertido = valor/5;
    case 2 -> valor_convertido = valor/4;
    case 3 -> valor_convertido = valor/4.4;
    default -> "Unexpected value: ";
});
```

- As *switch expressions* oferecem uma maneira mais concisa e expressiva de lidar com várias condições do que a forma tradicional de instrução switch. Elas podem ser usadas para simplificar o código e melhorar a legibilidade em muitos casos.