 INFORMA, FORMA, TRANSFORMA.


# Operador de Atribuição

- Atribuição simples ( = )
- A expressão da direita é atribuída à variável que está à esquerda.



```
x = 70;  
Y = x + 10;
```

- Pode ser utilizado de forma encadeada

```
x = y = z = 100;
```




3

  INFORMA, FORMA, TRANSFORMA.



# Operadores Binários

- Requerem dois operandos
- Operações aritméticas básicas
- Funcionam com variáveis e literais

```
int x, y, z;  
x = 230 + 40;      // adição  
y = 70 - 5;        // subtração  
z = 15 * 4;        // multiplicação  
x = y / z;         // divisão  
y = x % z;         // resto da divisão
```




4



INFORMA. FORMA. TRANSFORMA.

# Operadores Unários

- O operador ++ incrementa de 1
- O operador -- decrementa de 1
- Duas formas de utilização: pré-fixada e pós-fixada.

```
int x, y;  
x = 5;  
y = x++; // pós-fixado => x = 6 e y = 5  
  
x = 5;  
y = ++x; // pré-fixado => x = 6 e y = 6
```


5



INFORMA. FORMA. TRANSFORMA.

# Operadores Relacionais

Os operadores relacionais sempre retornam um valor do tipo boolean:

- > Maior que
- >= Maior ou igual a
- < Menor que
- <= Menor ou igual a
- == Igual a
- != Diferente de


6



  INFORMA, FORMA, TRANSFORMA.

# Operadores Lógicos Básicos

Os operadores lógicos básicos são:


&&	E
	Ou
!	Não

 7

  INFORMA, FORMA, TRANSFORMA.

## Exercício ***10\_AlgoritmosJava***

Desenvolva uma Classe em Java com o nome ***Operadores***, que leia o nome do aluno, 4 notas, informe a média destas notas.

 8

## Estrutura de Decisão

## Desvio Condicional Simples

```
...  
if (<condição>) {  
    <instruções para  
    condição  
    verdadeira>  
}  
<instruções para  
condição falsa ou  
após ser  
verdadeira>
```

Em Pseudocódigo: *Se .. Fimse*  
Em Java: *if { .. }*

10

Início

"Digite 2 números"

A, B

$X \leftarrow A + B$

$X > 10$

N

S

"O valor da soma é" + X

Fim

```
package controller;
import util.Teclado;

public class DesvioCondicionalSimples {
    public static void main(String[] args) {

        int a, b, x;
        a = Teclado.lerInt("Informe o primeiro número:");
        b = Teclado.lerInt("Informe o segundo número:");
        x = a + b;
        if (x > 10) {
            System.out.println("O valor da soma é " + x);
        }
    }
}
```

Fazer isto dentro do projeto 10\_AlgoritmosJava

11

Condição

N

S

Instruções executadas quando condição falsa

Instruções executadas quando condição verdadeira

```
...
if (<condição>) {
    <instruções para
    condição verdadeira>
}else{
    <instruções para
    condição falsa>
}
```

Em Pseudocódigo: Se .. Senao .. Fimse  
Em Java: if { .. } else { .. }

12

Fabício Curvello Gomes

6

INFORMA. FORMA. TRANSFORMA.

# Desvio Condicional Composto (Cont.)

Exemplo: Programa para ler dois valores numéricos, efetuar a adição. Caso a soma seja maior ou igual a 10, apresente o resultado somando 5. Caso a soma seja menor que 10, apresente o resultado subtraindo 7.

```

package controller;
import util.Teclado;
public class DesvioCondicionalComposto {
    public static void main(String[] args) {

        int a, b, x;
        a = Teclado.lerInt("Informe o primeiro número:");
        b = Teclado.lerInt("Informe o segundo número:");
        x = a + b;
        if (x >= 10){
            System.out.println("O resultado acrescido de 5 é " + (x + 5));
        }else{
            System.out.println("O resultado subtraído de 7 é " + (x - 7));
        }
    }
}

```

Fazer isto dentro do projeto **10\_AlgoritmosJava**

```

graph TD
    Inicio([Início]) --> Digite{{"Digite 2 números:"}}
    Digite --> AB[A, B]
    AB --> Soma["X <- A + B"]
    Soma --> Cond{X >= 10}
    Cond -- N --> Sub["O resultado subtraído de 7 é " + (X - 7)]
    Cond -- S --> Sum["O resultado acrescido de 5 é " + (X + 5)]
    Sub --> Fim([Fim])
    Sum --> Fim

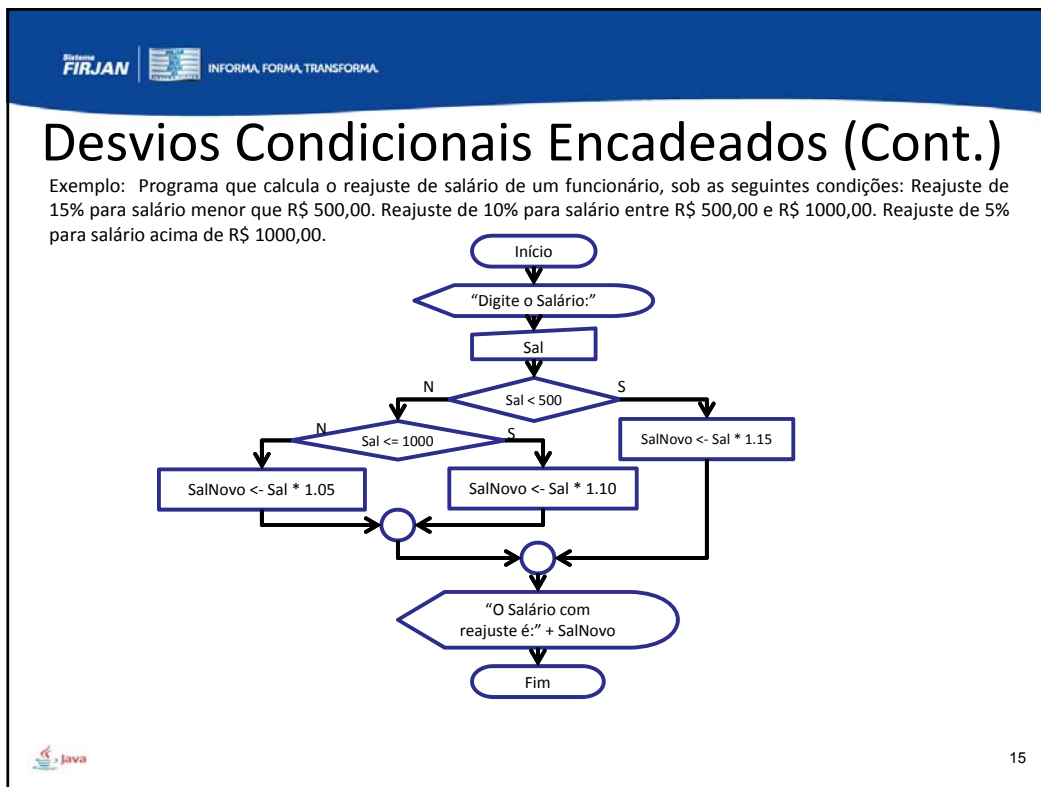
```

13

# Desvios Condicionais Encadeados

```
graph TD; Start(( )) --> Cond1{Condição 1}; Cond1 -- N --> Cond2{Condição 2}; Cond1 -- S --> Exec1[Instruções executadas quando condição 1 verdadeira]; Cond2 -- N --> Exec2[Instruções executadas quando condições 1 e 2 são falsas]; Cond2 -- S --> Exec3[Instruções executadas quando condição 1 falsa, mas a condição 2 é verdadeira]; Exec1 --> Join1(( )); Exec2 --> Join1; Exec3 --> Join2(( )); Exec1 --> Join2; Join1 --> Join2; Join2 --> End(( ))
```

The flowchart illustrates cascaded conditional branching. It starts with an entry arrow leading to a decision diamond labeled 'Condição 1'. If 'Condição 1' is true (S), the flow goes to a box labeled 'Instruções executadas quando condição 1 verdadeira'. If 'Condição 1' is false (N), the flow goes to another decision diamond labeled 'Condição 2'. From 'Condição 2', if true (S), the flow goes to a box labeled 'Instruções executadas quando condição 1 falsa, mas a condição 2 é verdadeira'. If 'Condição 2' is false (N), the flow goes to a box labeled 'Instruções executadas quando condições 1 e 2 são falsas'. The paths from the 'Condição 1' true branch and the 'Condição 2' false branch merge at a junction point. The paths from the 'Condição 2' true branch and the 'Condição 1' false branch merge at another junction point. Finally, the paths from the first junction point and the 'Condição 1' false branch merge at a third junction point, leading to the exit arrow.



**Desvios Condicionais Encadeados (Cont.)**

Exemplo: Programa que calcula o reajuste de salário de um funcionário, sob as seguintes condições: Reajuste de 15% para salário menor que R\$ 500,00. Reajuste de 10% para salário entre R\$ 500,00 e R\$ 1000,00. Reajuste de 5% para salário acima de R\$ 1000,00.

```
package controller;
import util.Teclado;
public class DesvioCondicionalEncadeado {
    public static void main(String[] args) {
        double sal, salNovo;
        sal = Teclado.lerDouble("Digite o Salário: ");
        if (sal < 500){
            salNovo = sal * 1.15;
        }else{
            if (sal <= 1000){
                salNovo = sal * 1.10;
            }else{
                salNovo = sal * 1.05;
            }
        }
        System.out.println("O Salário com reajuste é: " + salNovo);
    }
}
```

Fazer isto dentro do projeto 10\_AlgoritmosJava

16





**Sistema FIRJAN** | **INFORMA, FORMA, TRANSFORMA.**

## Laço com Teste Lógico no Início

Exemplo: Programa para pedir a leitura de um valor para a variável X, multiplicar este valor por 3, colocar o valor obtido na variável R, e apresentar o valor de R, repetindo a sequência cinco vezes.

```
graph TD; Inicio([Início]) --> I1[I <- 1]; I1 --> Cond{I <= 5}; Cond -- N --> Fim([Fim]); Cond -- S --> Input[/Digite um número: /]; Input --> X[X]; X --> Calc[R <- X * 3]; Calc --> R[R]; R --> Inc[I <- I + 1]; Inc --> Cond;
```



```
package controller;
import util.Teclado;
public class LacoTesteInicio {
    public static void main(String[] args) {
        int r, x, i;
        i = 1;
        while (i <= 5){
            x = Teclado.lerInt("Digite um número: ");
            r = x * 3;
            System.out.println(r);
            i = i + 1;
        }
    }
}
```

Fazer isto dentro do projeto **10\_AlgoritmosJava**

Em Pseudocódigo: Enquanto .. Faça  
Em Java: while { .. }

Java

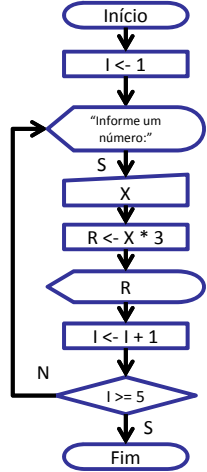
18

INFORMA, FORMA, TRANSFORMA.

## Laço com Teste Lógico no Fim

Vamos utilizar o mesmo exemplo aplicado anteriormente: Programa para pedir a leitura de um valor para a variável X, multiplicar este valor por 3, colocar o valor obtido na variável R, e apresentar o valor de R, repetindo a sequência cinco vezes.



```

graph TD
    Inicio([Início]) --> I1[I <- 1]
    I1 --> Prompt[/Informe um número:/]
    Prompt --> S1{S}
    S1 --> X[X]
    X --> R1[R <- X * 3]
    R1 --> R[R]
    R --> I2[I <- I + 1]
    I2 --> Cond1{I >= 5}
    Cond1 -- N --> Prompt
    Cond1 -- S --> Fim([Fim])
        
```



```

package controller;
import util.Teclado;
public class LacoTesteFim {
    public static void main(String[] args) {
        int r, x, i;
        i = 1;
        do {
            x = Teclado.lerInt("Digite um número: ");
            r = x * 3;
            System.out.println(r);
            i = i + 1;
        } while (i <= 5);
    }
}
        
```

Fazer isto dentro do projeto **10\_AlgoritmosJava**

Em Pseudocódigo: Repita .. Até  
Em Java: `do { .. } while`

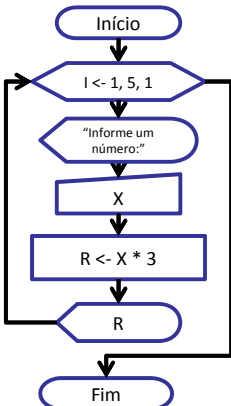
19

INFORMA, FORMA, TRANSFORMA.

## Laço com Variável de Controle

Vamos utilizar mais uma vez o mesmo exemplo aplicado anteriormente: Programa para pedir a leitura de um valor para a variável X, multiplicar este valor por 3, colocar o valor obtido na variável R, e apresentar o valor de R, repetindo a sequência cinco vezes.



```

graph TD
    Inicio([Início]) --> I1[I <- 1, 5, 1]
    I1 --> Prompt[/Informe um número:/]
    Prompt --> X[X]
    X --> R1[R <- X * 3]
    R1 --> R[R]
    R --> I2[I <- 1, 5, 1]
    I2 --> Prompt
    R --> Fim([Fim])
        
```


```

package controller;
import util.Teclado;
public class LacoVariavelControle {
    public static void main(String[] args) {
        int r, x, i;
        for (i=1; i<=5; i++){
            x = Teclado.lerInt("Digite um número: ");
            r = x * 3;
            System.out.println(r);
        }
    }
}
        
```

Fazer isto dentro do projeto **10\_AlgoritmosJava**


Em Pseudocódigo: Para .. Fimpara  
Em Java: `for { .. }`

20

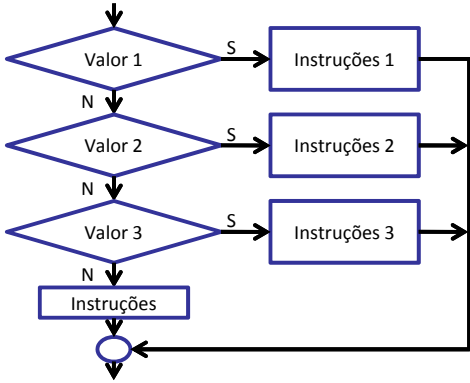




## Programação com Múltipla Escolha




## Programação com Múltipla Escolha




```
switch (<expressão de seleção>){  
  case <Valor 1>:  
    <instruções 1>;  
    break;  
  case <Valor 2>:  
    <instruções 2>;  
    break;  
  case <Valor 3>:  
    <instruções 3>;  
    break;  
  default:  
    <instruções>;  
}
```

Em Pseudocódigo: *Escolha .. Caso .. Outrocaso .. fimescolha*  
Em Java: `switch { .. case .. default .. }`



22



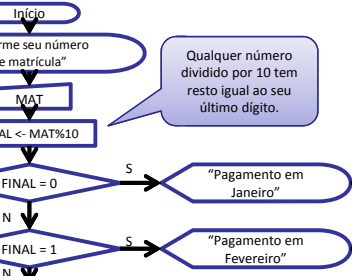
**Sistema**  
**FIRJAN**

INFORMA, FORMA, TRANSFORMA.

# Programação com Múltipla Escolha (Cont.)

**Exemplo:** Programa que pergunte o nº de matrícula de sócio de um Clube, e exibe o mês de pagamento da anuidade, conforme tabela ao lado:

Nº Final da Matrícula	Mês de Pagamento
0	Janeiro
1	Fevereiro
2	Março
3	Abril
Outro final	Maio




```


graph TD
    Inicio([Início]) --> Input[/Informe seu número de matrícula/]
    Input --> MAT[MAT]
    MAT --> Calc[FINAL <- MAT%10]
    Calc --> Dec1{FINAL = 0}
    Dec1 -- S --> Jan[/Pagamento em Janeiro/]
    Dec1 -- N --> Dec2{FINAL = 1}
    Dec2 -- S --> Feb[/Pagamento em Fevereiro/]
    Dec2 -- N --> Dec3{FINAL = 2}
    Dec3 -- S --> Mar[/Pagamento em Março/]
    Dec3 -- N --> Dec4{FINAL = 3}
    Dec4 -- S --> Apr[/Pagamento em Abril/]
    Dec4 -- N --> Dec5{Pagamento em Maio}
    Jan --> Con(( ))
    Feb --> Con
    Mar --> Con
    Apr --> Con
    Dec5 --> Con
    Con --> Fim([Fim])
    
```

```

package controller;
import util.Teclado;
public class ProgramacaoMultiplaEscolha {
    public static void main(String[] args) {
        int matr, numFinal;
        matr = Teclado.lerInt("Informe matrícula");
        numFinal = matr%10;
        switch (numFinal){
            case 0:
                System.out.println("Pagamento em Janeiro");
                break;
            case 1:
                System.out.println("Pagamento em Fevereiro");
                break;
            case 2:
                System.out.println("Pagamento em Março");
                break;
            case 3:
                System.out.println("Pagamento em Abril");
                break;
            default:
                System.out.println("Pagamento em Maio");
        }
    }
}
    
```

Fazer isto dentro do projeto **10\_AlgoritmosJava**

**Sistema  
FIRJAN**



INFORMA, FORMA, TRANSFORMA.


# Projeto *InfoNote\_04*

## Objetivos:


- Implementar entrada de dados no sistema
- Implementar estrutura de decisão, repetição e escolha.
- Implementar navegação de telas.

Esta tarefa está descrita num documento específico com o seguinte nome:


[PD - TI - 05.1 - Instruções Projeto InfoNote\\_04.pdf](#)




# Dúvidas?




25




# Bibliografia



Java Como Programar 8ª Edição  
Paul Deitel e Harvey Deitel  
Ed. Pearson



Java 7 Ensino Didático  
Sérgio Furgeri  
Ed. Érica



Fundamentos de Computação e Orientação a Objetos Usando Java  
Francisco A. C. Pinheiro  
Ed. LTC

26