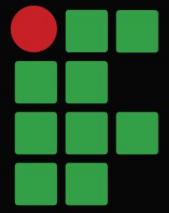
Comandos Condicionais

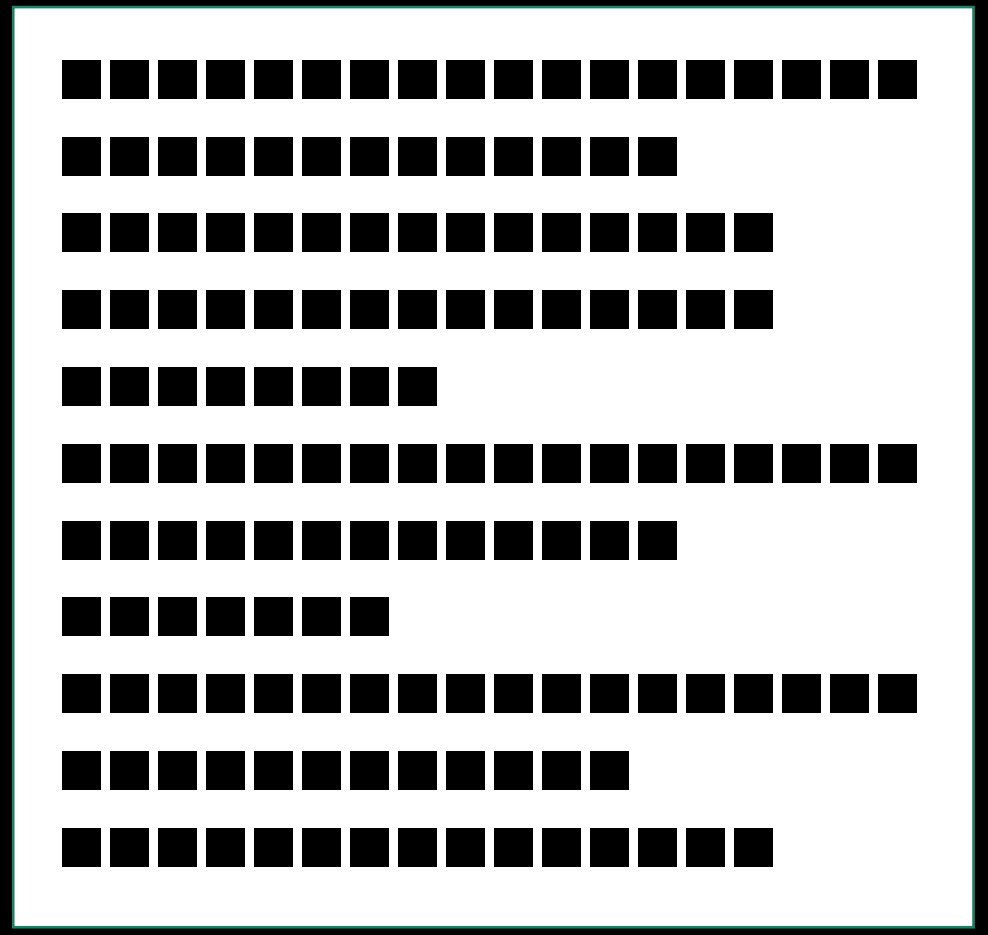
Prof. Alexandro M. S. Adário



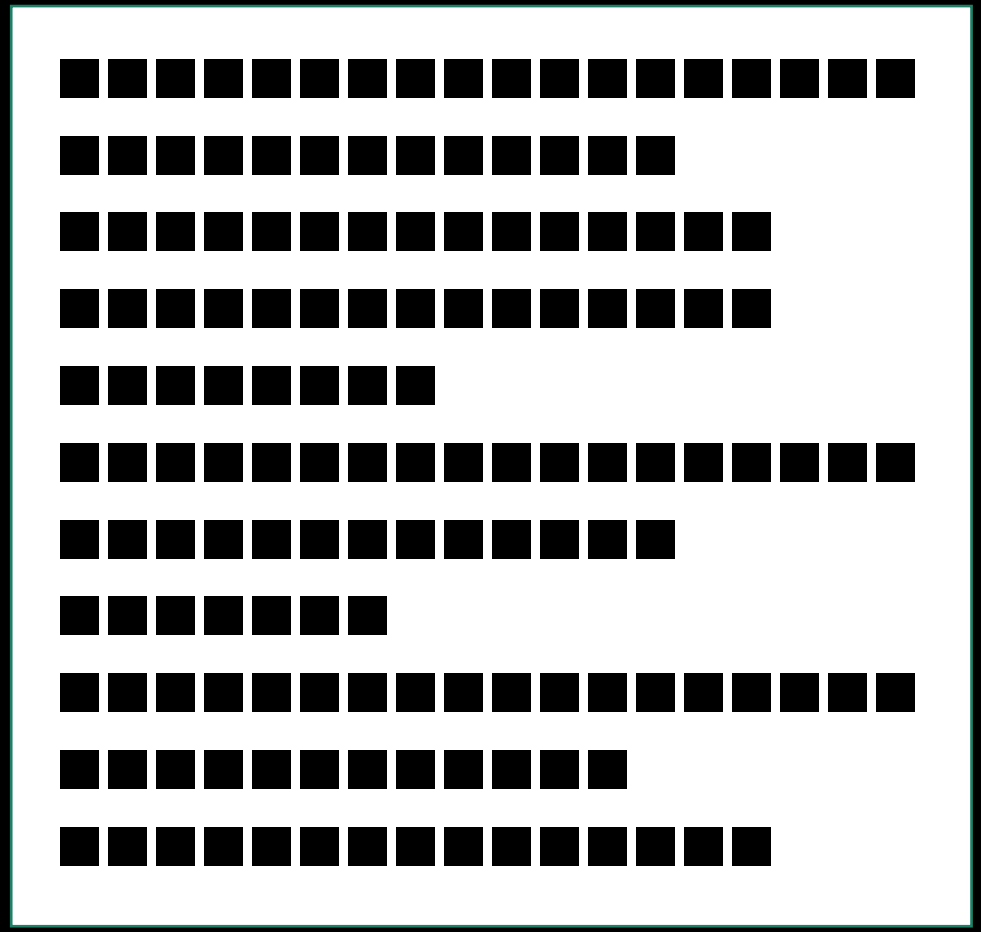
INSTITUTO FEDERAL
Rio Grande do Sul
Campus Erechim

Até
agora...

Os algoritmos que desenvolvemos eram estritamente sequenciais.



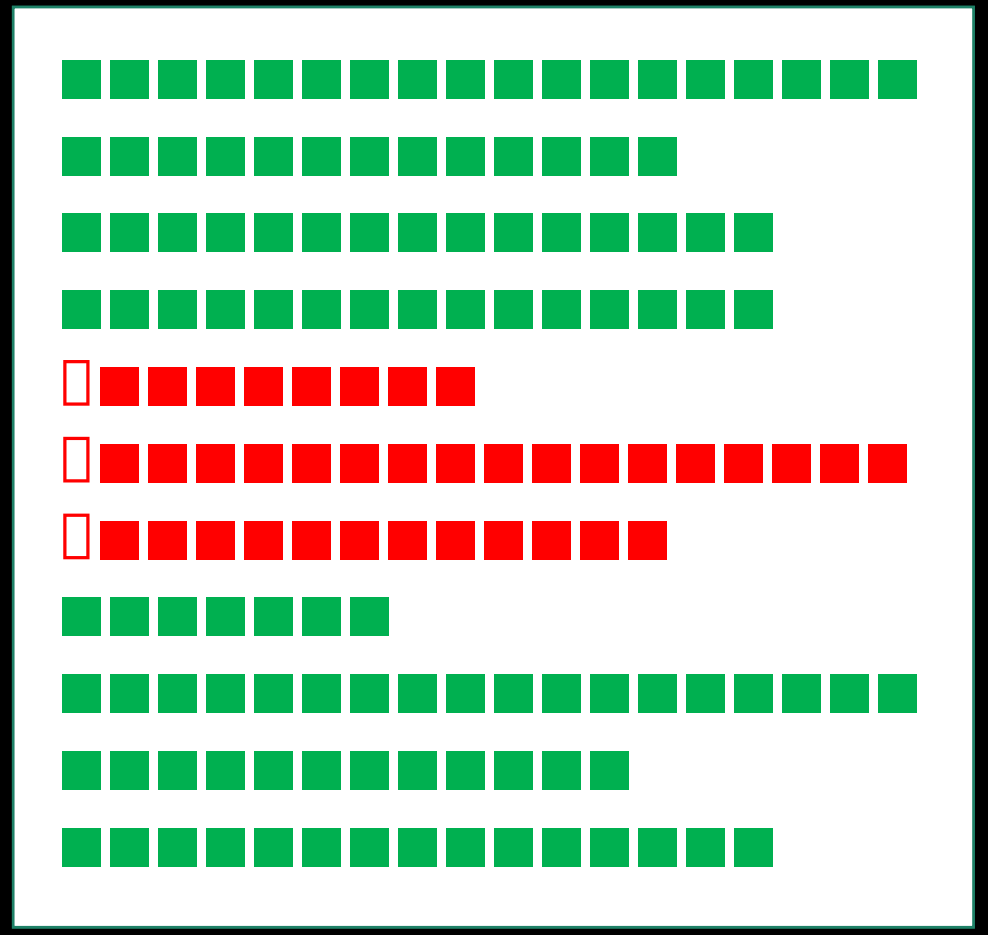
As instruções eram executadas numa ordem pré-definida, uma após a outra, sem alteração.



Essa maneira de executar nos dá PREVISIBILIDADE e FACILIDADE DE DEPURAÇÃO (testar um algoritmo).

Mas a execução puramente sequencial não é a realidade para a maioria dos casos de algoritmos.

Há casos em que, conforme as condições, não devem ser executada algumas instruções.



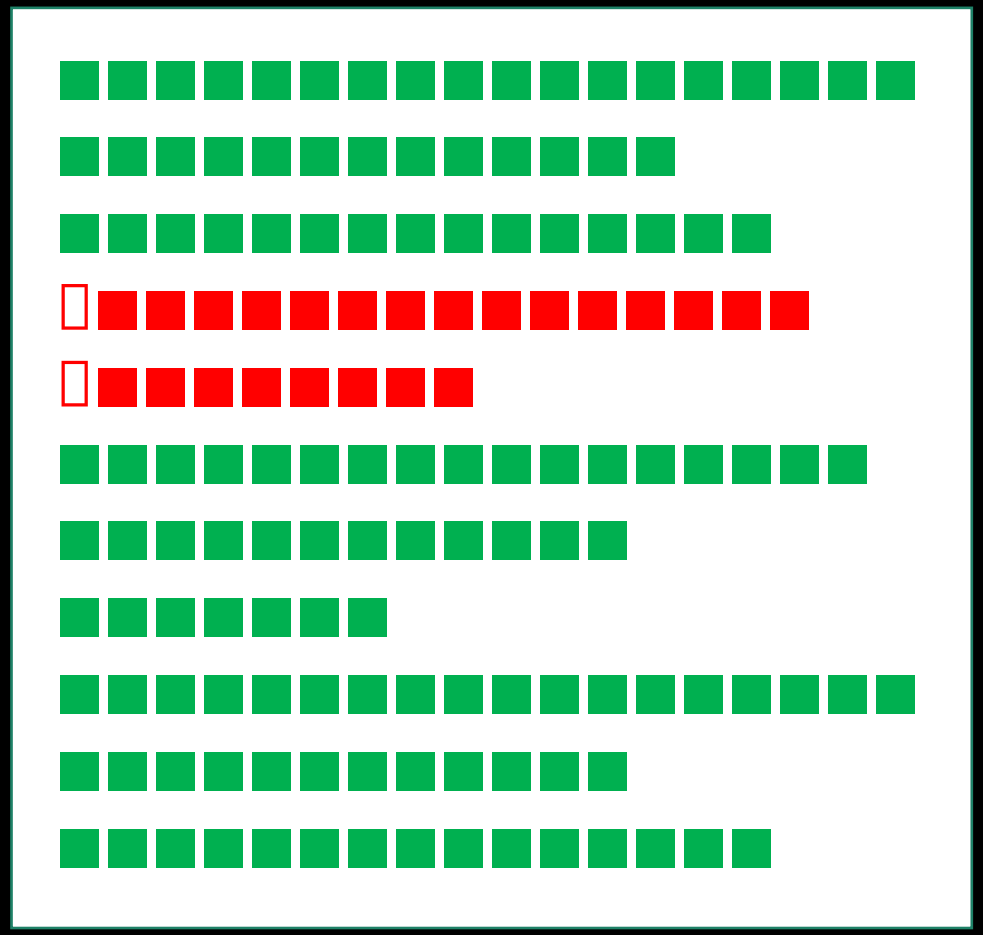
Exemplo:

Calcular a raiz quadrada de um número real

1º **Solicitamos que o usuário digite o número**

2º **Se o número é não negativo, calcula-se a raiz quadrada.**

Há outros casos em que, conforme as condições, são executadas instruções diferentes para cada condição.



Exemplo:
Determinar se um número é par ou ímpar

1º **Solicitamos que o usuário digite o número**

2º **Dividimos o número por 2 para obter o resto (pode ser 0 ou 1)**

3º **Se o resto for 0, indica que o número é par**

4º **Se o resto for 1, indica que o número é ímpar**

Estrutura de um comando condicional

A condição deve retornar um valor booleano, VERDADEIRO ou FALSO.

```
if( <condição> ) { //entao
    <instruções (uma ou mais) a
    serem realizadas se a
    condição for verdadeira>
} //fim_do_comando
```

Se a condição for VERDADEIRA, as instruções SÃO executadas.

Se a condição for FALSA, as instruções NÃO SÃO executadas.

EXEMPLO de um comando condicional

numero < 0 : condição FALSA
numero >= 0 : condição VERDADEIRA

```
if( numero >= 0 ) {
    resultado = Math.sqrt(numero);
    System.out.println(resultado);
}
```

As instruções vão ser executadas se a condição for VERDADEIRA (numero >= 0).

Quando numero < 0, condição FALSA, as instruções não serão executadas.

EXEMPLO de um algoritmo com comando condicional

```
import java.util.Scanner;
public class CalculaRaiz {
    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        int numero = teclado.nextInt();

        if( numero >= 0 ) {
            resultado = Math.sqrt(numero);
            System.out.println(resultado);
        }
    }
}
```

Estrutura de um comando condicional composto

A condição deve retornar um valor booleano, VERDADEIRO ou FALSO.

Se a condição for VERDADEIRA...

```
if( <condição> ) {
    <instruções a executar se
    a condição for verdadeira>
} else {
    <instruções a executar se
    a condição for falsa>
}
```

Se a condição for FALSA...

EXEMPLO de um comando condicional composto

numero ≠ 0 : condição FALSA
numero = 0 : condição VERDADEIRA

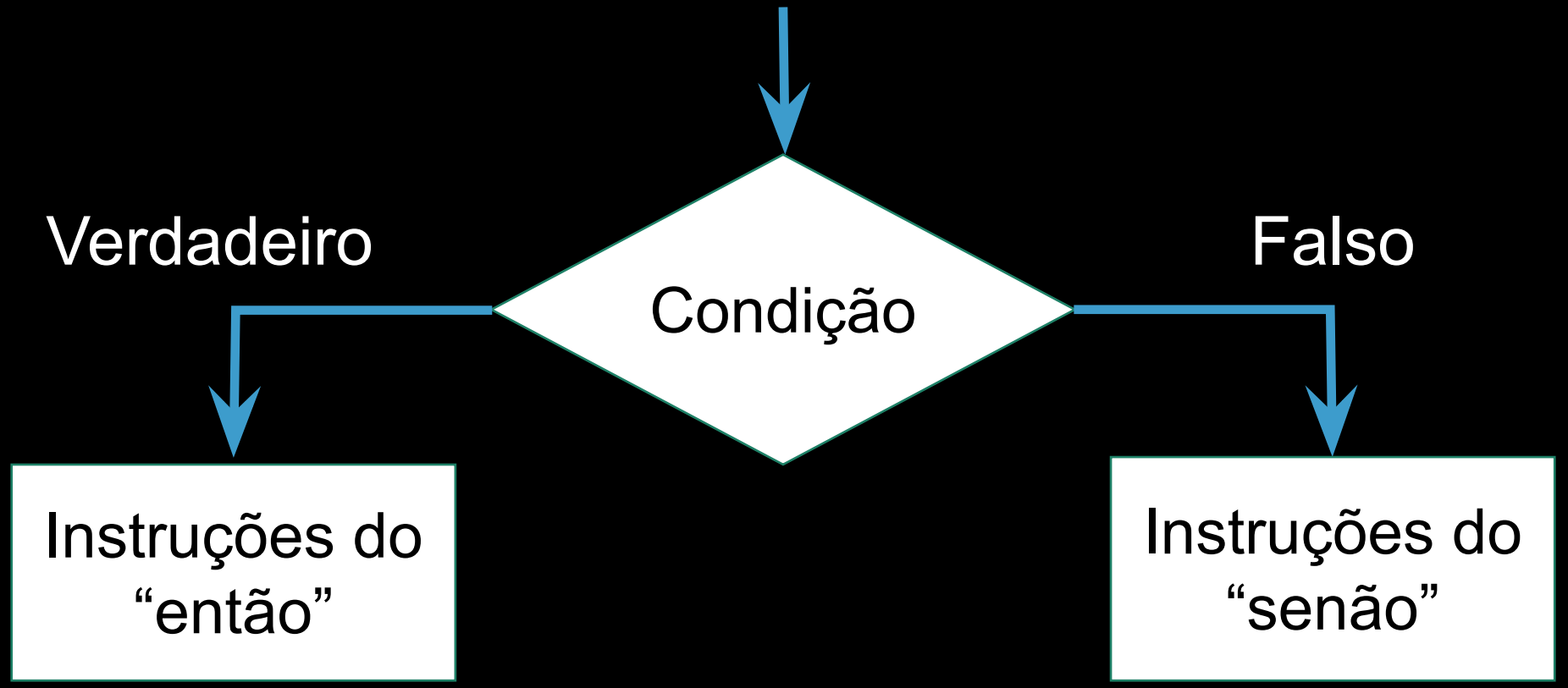
A instrução será executada se a condição for verdadeira (resto = 0).

```
if( resto == 0 ) {
    System.out.print("número é par");
} else {
    System.out.print("número é ímpar");
}
```

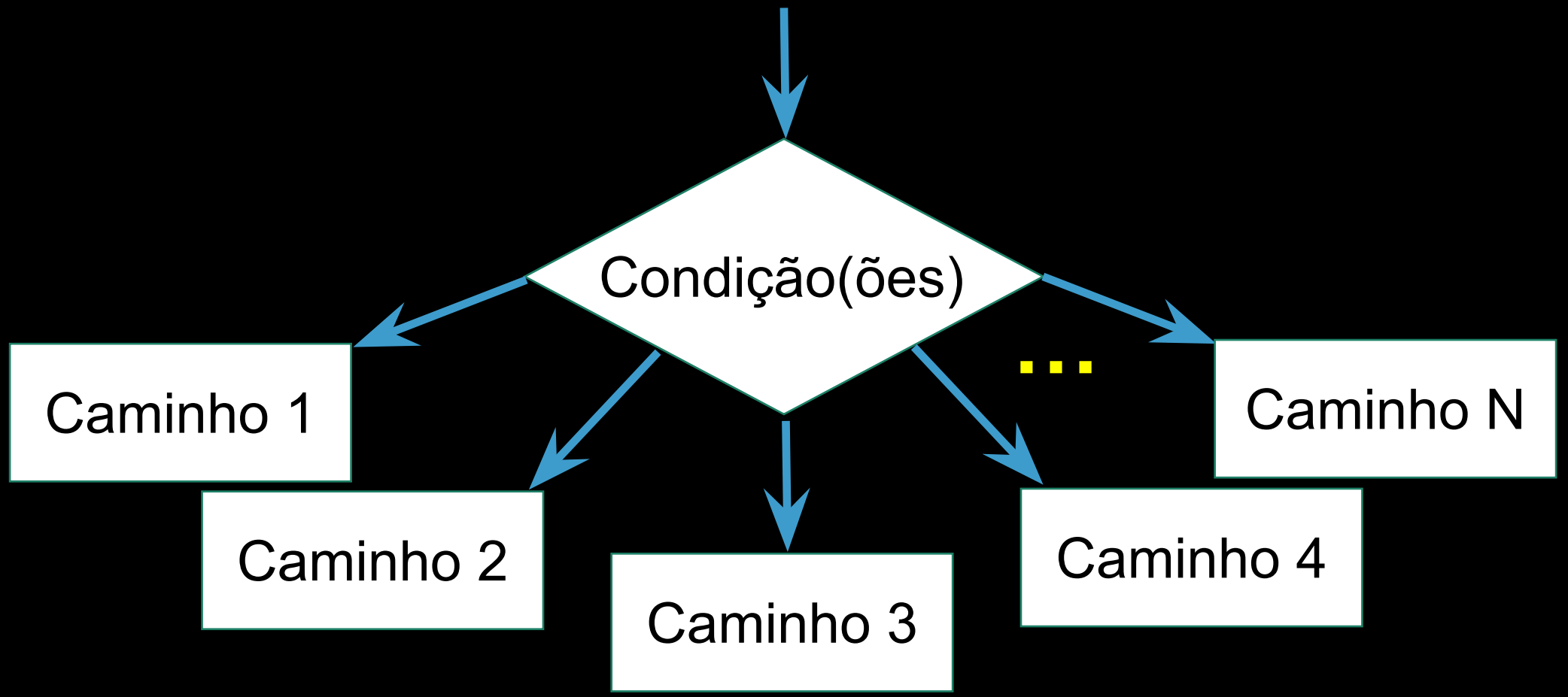
Quando resto ≠0, condição falsa, essa instrução vai ser executada.

EXEMPLO de um algoritmo com comando condicional composto

```
import java.util.Scanner;
public class ParOuImpar {
    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        int numero = teclado.nextInt();
        int resto = numero % 2;
        if( resto == 0 ) {
            System.out.print("número é par");
        } else {
            System.out.print("número é ímpar");
        }
    }
}
```



Decisões binárias



Decisões múltiplas

Exemplo:

Determinar if(um número é positivo, negativo ou zero

1º **Solicitamos que o usuário digite o número**

2º **Verificamos if(o número é positivo (>0) ou negativo (<0) ou igual (=) a zero**

Como???
São 3 casos!!!

Solução:
Comandos condicionais aninhados (multinível)
Agrupamento de condições.

É preciso DECIDIR a ordem em que as condições serão testadas e agrupadas.

numero <= 0 : condição FALSA
numero > 0 : condição VERDADEIRA

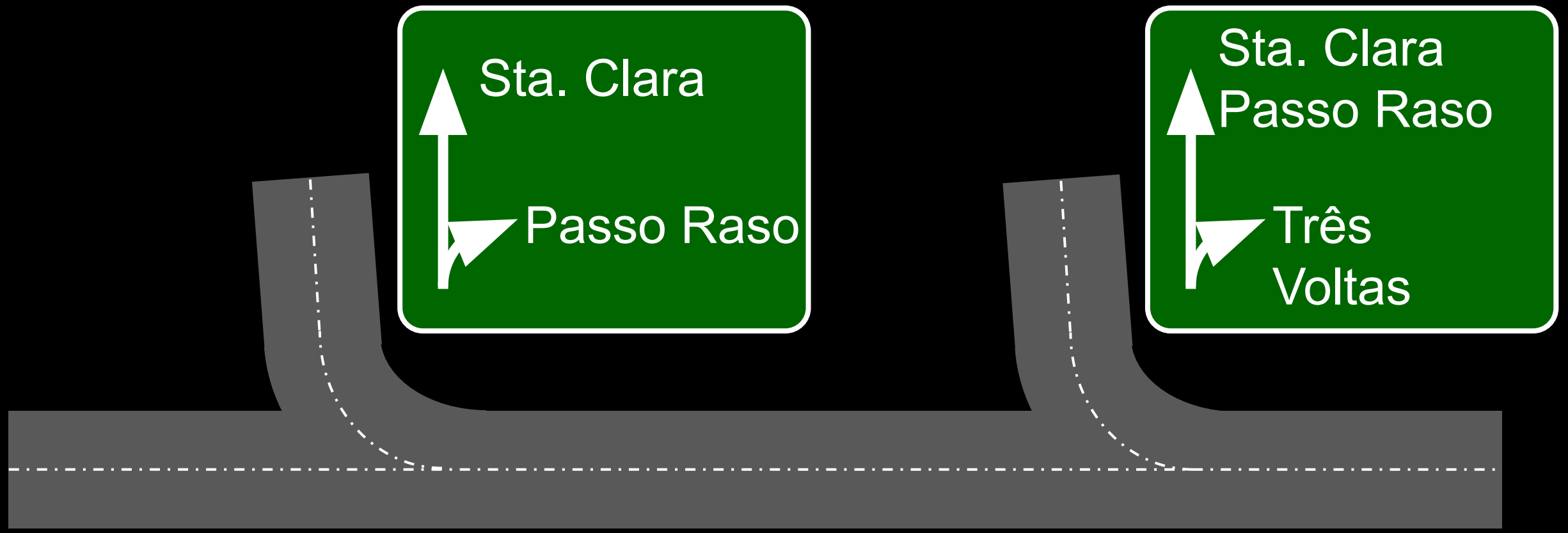
```
if( numero > 0 ) {
    // comandos executados caso o
    // número seja positivo
} else {
    // comandos executados caso o
    // número não seja positivo
}
```

```
if( numero > 0 ) {
    //execução if( o número é positivo
} else {
    //execução if( o número não é
    positivo
    if( numero < 0 ) {
        //execução if( o número é
    negativo
    } else {
        //execução if( o número é zero
    }
}
```

```
if( numero > 0 ) {
        //execução if( o número é positivo
} else {
        //execução if( o número não é
        positivo
    if( numero < 0 ) {
        //execução if( o número é
        negativo
    } else {
        //execução if( o número é zero
    }
}
```

```
if( numero > 0 ) {
    //execução if( o número é positivo
} else {
    //execução if( o número não é
    positivo
    if( numero < 0 ) {
        //execução if( o número é
        negativo
    } else {
        //execução if( o número é zero
    }
}
```

```
if( numero > 0 ) {
    //execução if( o número é positivo
} else {
    //execução if( o número não é
    positivo
    if( numero < 0 ) {
        //execução if( o número é
        negativo
    } else {
        //execução if( o número é zero
    }
}
```

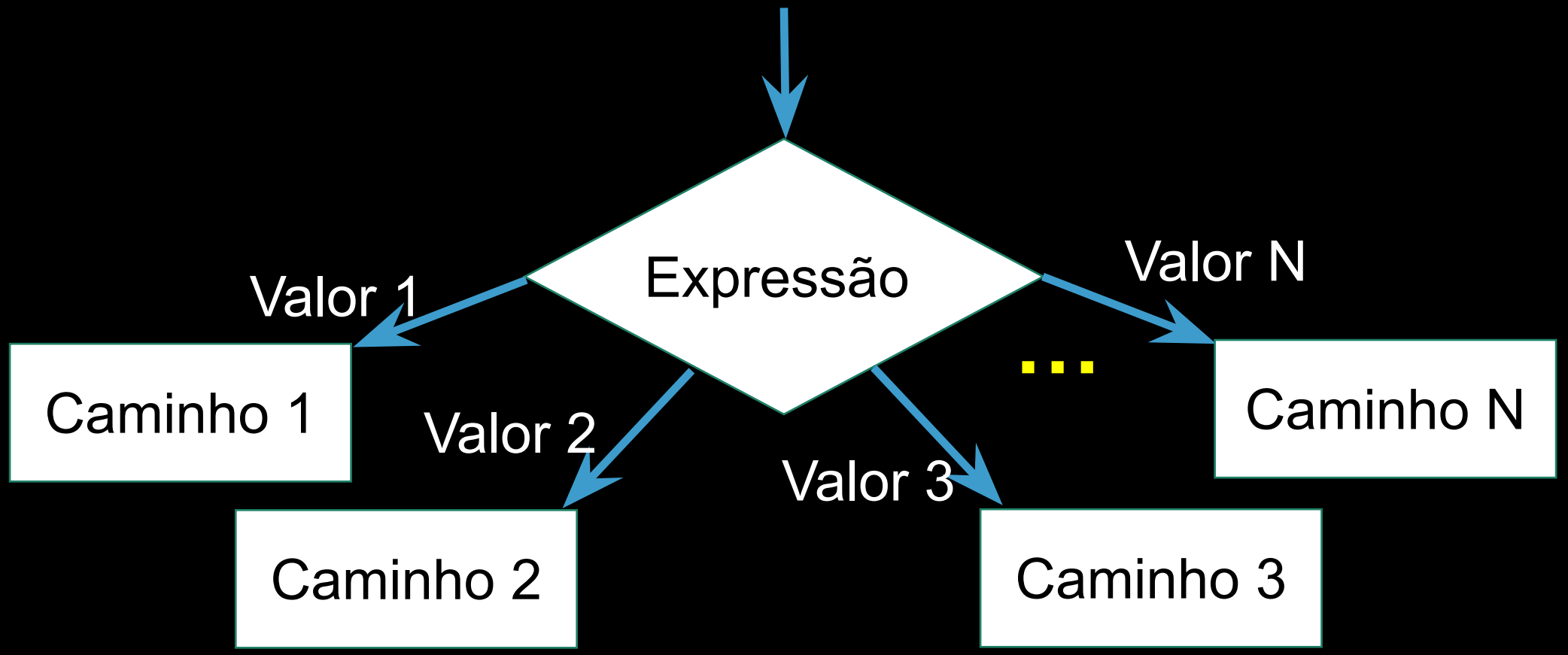

Sta. Clara
Passo Raso
Sta. Clara
Passo Raso
Três
Voltas

```
if( condicao1 ) {
      if( condicao2 ) {
         if( condicao4 ) {
         } else {
         }
      } else {
         if( condicao5 ) {
         } else {
         }
      }
} else {
      if( condicao3 ) {
         if( condicao6 ) {
         } else {
         }
      } else {
         if( condicao7 ) {
         } else {
         }
      }
}
```

```
1 if( condicao1 ) {
    2 if( condicao2 ) {
          // condicao1: V, condicao2: V 1
      } else {
          // condicao1: V, condicao2: F 2
      }
  } else {
    3 if( condicao3 ) {
          // condicao1: F, condicao3: V 3
      } else {
          // condicao1: F, condicao3: F 4
      }
  }
```

As condições de cada comando condicional não precisam ser baseadas nas mesmas variáveis!

```
if( presença >= 0.75 ) {
    if( nota >= 7.0 ) {
        System.out.print("Aprovado");
     } else {
         if( nota < 1.8 ) {
             System.out.print("Reprovado por nota");
         } else {
             System.out.print("Em exame");
         }
     }
} else {
    System.out.print("Reprovado por faltas");
}
```



Decisões múltiplas – Baseada na mesma expressão

```
if( expressao == valor1 ) {
    // bloco comandos 1
} else {
    if( expressao == valor2 ) {
        // bloco comandos 2
    } else {
        if( expressao == valor3 ) {
            // bloco comandos 1
        } else {
            if( expressao == valor4 ) {
                // bloco comandos 4
            } else {
                // bloco comandos 5
            }
        }
    }
}
```

Problemas:

- Solução deselegante
- Dificulta a legibilidade
- Dificulta a manutenção do código
- Facilita a introdução de erros

Solução:

Comando `"switch...case..."`

```
switch( expressao ) {
    case valor1 :
        // bloco comandos 1
        break;
    case valor2 :
        // bloco comandos 2
        break;
    case valor3 :
        // bloco comandos 3
        break;
    case valor4 :
        // bloco comandos 4
        break;
    default :
        // bloco comandos 5
}
```

```
if( expressao == valor1 ) {
    // bloco comandos 1
} else {
    if( expressao == valor2 ) {
        // bloco comandos 2
    } else {
        if( expressao == valor3 ) {
            // bloco comandos 3
        } else {
            if( expressao == valor4 ) {
                // bloco comandos 4
            } else {
                // bloco comandos 5
            }
        }
    }
}
```

```
switch( expressao ) {
    case valor1 :
        // bloco comandos 1
        break;
    case valor2 :
        // bloco comandos 2
        break;
    case valor3 :
        // bloco comandos 3
        break;
    case valor4 :
        // bloco comandos 4
        break;
    default :
        // bloco comandos 5
}
```

```
switch ( expressao ) {
    case valor1 :
        // bloco coma    s 1
        break;
    case valor2
        // bloco
        break;
    case valor3
         // bloco comandos 3
        break;
    case valor4 :
         // bloco comandos 4
        break;
    default :
        // bloco comandos 5
}
```

Expressão que serve para decidir qual bloco executará

```
switch( expressao ) {
    case valor1 :
        // bloco comandos
        break;
    case valor2 :
        // bloco comandos 2
        break;
    case valor3 :
        // bloco comandos 3
        break;
    case valor4 :
        // bloco coma
        break;
    default :
        // bloco coma
}
```

Cada "`case`“ define um caminho distinto que pode ser executadoo

Cada "`case`“ possui um valor correspondente. A expressão do “`switch`” será comparada com ele.

```
switch( expressao ) {
    case valor1 :
        // bloco comando  1
        break;
    case valor2 :
        // bloco comand
        break;
    case valor3 :
         // bloco coman
        break;
    case valor4 :
         // bloco comandos 4
        break;
    default :
        // bloco comandos 5
}
```

Se a expressão for IGUAL (==) a algum dos valores dos "`case`", o bloco referente a ele será executado.

```
switch( expressao ) {
    case valor1 :
        // bloco comandos 1
        break;
    case valor2 :
        // bloco comandos 2
        break;
    case valor3 :
         // bloco coma
        break;
    case valor4 :
         // bloco c
        break;
    default :
        // bloco comandos 5
}
```

Se a expressão NÃO CORRESPONDER a nenhum valor de nenhum caso, o bloco "`default`" será executado.

```
switch( expressao ) {
    case valor1 :
        // bloco comandos 1
        break;
    case valor2 :
        // bloco comandos 2
        break;
    case valor3 :
        // bloco comandos 3
        break;
    case valor4 :
        // bloco comandos 4
        break;
    default :
        // bloco comandos 5
}
```

O “`default`” é tudo aquilo que não se encaixa nos “`case`”, mas precisa de uma ação específica. Esse bloco é opcional !!!

ATENÇÃO – 1

A expressão do “`switch`” deve ser sempre compatível com um valor inteiro (`byte`, `short`, `int`, `long` ou `char`) ou enumerado ou String.

ATENÇÃO – 2

A avaliação da expressão do “`switch`” é sempre IGUAL ou DIFERENTE.
Não é possível testar faixas de valores (<, >, >=, <=). Quando isso é preciso, é necessário usar “`if`”

ATENÇÃO – 3

A instrução “**`break`**” no final de cada “**`case`**” é essencial para terminar o bloco e sair do “**`switch`**”. Se não for usada, o bloco seguinte (o do “**`case`**” logo abaixo) será executado.

Dica:

Em alguns casos, não usar o “**`break`**” é intencional e importante, para que 2 ou mais valores executem o mesmo bloco de comandos.

```
switch( expressao ) {
    case valor1 :
    case valor2 :
        // bloco comandos 1
        break;
    case valor3 :
    case valor4 :
    case valor5 :
        // bloco comandos 2
        break;
    default :
        // bloco comandos 3
}
```

Exemplo:

Classificar um atleta de acordo com o ano de nascimento.

1º **Solicitamos que o usuário digite o ano**

2º **Verificamos o ano**

3º **Definimos a categoria do atleta**