

---

---

# Estruturas Condicionais

— Aula 2 —

---

---

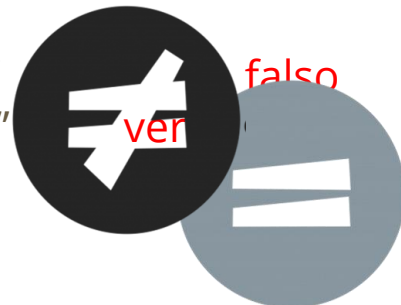
# Operadores Relacionais

- As operações aritméticas (soma, adição, multiplicação, divisão) resultam em um valor numérico.
- Os operadores relacionais testam relações entre variáveis ou expressões, gerando resultados lógicos (VERDADEIRO ou FALSO).
  - Os operadores lógicos também geram em um valor lógico.
- Podemos comparar uma expressão relacional a uma pergunta cujo veracidade estamos testando. Se a expressão relacional for verdade, ela resulta no valor lógico VERDADEIRO. Caso contrário, ela resulta no valor lógico FALSO.

"um é igual a dois?"

"um mais um é menor que quatro?"

"a raiz quadrada de quatro é dois?" verdadeiro



# Operadores Relacionais

- No VisuAlg, os operadores relacionais são:

Operador	Nome	Exemplo
>	Maior que	$A > B$ (variável A é maior que a variável B?)
<	Menor que	$A < B$ (variável A é menor que a variável B?)
>=	Maior ou igual a	$A \geq B$ (variável A é maior ou igual a variável B?)
<=	Menor ou igual a	$A \leq B$ (variável A é menor ou igual a variável B?)
= ( == no Portugol Studio )	Igual a	$A = B$ (variável A é igual a variável B?)
<> ( != no Portugol Studio )	diferente de	$A \neq B$ (variável A é diferente da variável B?)

# Operadores Relacionais

- Uma operação relacional, basicamente, consiste em uma comparação entre dois valores, a qual resulta em VERDADEIRO ou FALSO.
- Por exemplo, se compararmos os números 1 e 2 para sabermos se eles são iguais, podemos usar a expressão:

$1 = 2$  (lê-se: um é igual a dois?)

O resultado disso será FALSO, pois os valores são obviamente diferentes.

- Também podemos fazer o contrário:

$1 \neq 2$  (lê-se: um é diferente de dois?)

- Agora, o resultado é VERDADEIRO, pois os dois valores são diferentes.

# Operadores Relacionais

- Podemos observar também o que ocorre com os outros operadores:

$1 > 2$  (lê-se: um é maior que dois?) // FALSO

$1 < 2$  (lê-se um é menor que dois?) // VERDADEIRO

$1 \geq 2$  (lê-se um é maior ou igual a dois?) // FALSO

$1 \leq 2$  (lê-se um é menor ou igual a dois?) // VERDADEIRO

- Obviamente, com os operadores  $\geq$  e  $\leq$ , é verificado também se os valores são iguais. Se forem iguais, o resultado é verdadeiro

# Valores de operações relacionais

- Alguns operadores relacionais “não funcionam” (até por não fazer sentido) com alguns tipos de variáveis. Se forem valores do tipo caractere ou lógico só é possível usar o = (igual a) e <> (diferente de). Para os tipos numéricos real e inteiro podemos usar qualquer operador lógico para comparação:

- VERDADEIRO = VERDADEIRO // Correto
- “Mario” = “Luigi” // Correto
- 100.50 <> 10 // Correto
- 23 >= 30 // Correto
- VERDADEIRO > FALSO // Errado
- “Ricardo” >= “Roger” // Errado

	=	<>	>	<	>	>=	<=
logico	✓	✓	✗	✗	✗	✗	✗
caractere	✓	✓	✗	✗	✗	✗	✗
inteiro	✓	✓	✓	✓	✓	✓	✓
real	✓	✓	✓	✓	✓	✓	✓

# Valores de operações relacionais

- Mesmo que fosse possível usar operadores relacionais com todos os tipos de dados, somente é possível comparar dois valores do mesmo tipo, ou seja, tipo lógico com tipo lógico, tipo numérico com tipo numérico e tipo caractere com tipo caractere.
- Por exemplo, não é possível fazer a seguinte expressão:

`2 >= "Mario" // (2 é igual a "Mário"??)`

- Essa expressão não faz nem sentido, pois um número não pode ser comparado com um texto.\*
- Outro exemplo é:

`VERDADEIRO <> 32.03 // (VERDADEIRO é diferente de 32.03??)`

- Novamente não é possível comparar esses valores, pois um é lógico e outro é real.

\* Em algumas linguagens de programação é possível comparar tipos diferentes, mas não é o nosso caso com o VisuAlg.

## Valores de operações relacionais

Quando tentamos fazer essas operações relacionais com tipos diferentes o VisuAlg apresenta um erro de compilação, pois estamos tentando fazer uma operação que é impossível de ser feita e o programa não consegue converter isso em um resultado válido.

### Aviso

VisuAlg encontrou o seguinte problema em seu algoritmo, na linha 9 :

Erro na atribuição de valores à variável RESULTADO:  
LOGICO para INTEIRO.

Conteúdo da linha:

```
resultado := 1 = "Mario"
```

Explicação:

Não há explicação disponível para este problema.

Continuar

Terminar



# Valores de operações relacionais

- Podemos usar variáveis, valores e expressões com os valores lógicos:

`VariavelA <> VariavelB` // Comparação entre variáveis

`VariávelA > 12%2` // Comparação usando variável e expressão.

`"Henrique" = "henrique"` // Comparação entre valores

- Lembrando que as variáveis precisam ser do mesmo tipo para a comparação ser considerada válida (exceto inteiro com real, pois são números).

# Exemplos

var

A : logico

B : inteiro

inicio

A <- VERDADEIRO // Variável A é igual VERDADEIRO

B <- 5 //Variável B é igual 5

escreval(VERDADEIRO = A) // VERDADEIRO é igual a VERDADEIRO?

escreval(2 > 2.2) // 2 é igual a 2.2?

escreval(5 <= B) // 5 é menor ou igual a 5?

escreval("Mario" <> "Toad") // Mario é diferente de Toad?

fimalgoritmo

# Exercício

1. Marque como V (válida) ou I (inválida), as expressões lógicas abaixo:

a. VERDADEIRO = FALSO ( )

e. VERDADEIRO <> "FALSO" ( )

b.  $5 > 2$  ( )

f.  $67 \geq 3$  ( )

c.  $4 = "4"$  ( )

g. "Mario" > "Luigi" ( )

d.  $4 < 3$  ( )

h. FALSO < VERDADEIRO ( )

2. Responda qual o resultado de das expressões lógica abaixo:

a. VERDADEIRO = VERDADEIRO

e. "Mario" <> "Toad"

# Operadores Lógicos


- Os operadores lógicos permitem combinar expressões relacionais e testá-las simultaneamente.
- Da mesma maneira que os operadores relacionais, os operadores lógicos também retornam um valor lógico como resultado.
- Os operadores lógicos são uma forma de realizar testes lógicos com variáveis lógicas.
- No VisuAlg, há três tipos de operadores lógicos:
  - O operador **E**
  - O operador **OU**
  - O operador **NAO** (lê-se: não)

# Operador E

- Este operador serve para fazer a comparação entre dois valores lógicos e o resultado depende dos valores lógicos que comparamos.
- Vamos supor que temos duas variáveis p e q, dependendo do valor dessas variáveis o resultado pode ser VERDADEIRO ou FALSO:

p	q	p E q
VERDADEIRO	VERDADEIRO	VERDADEIRO
VERDADEIRO	FALSO	FALSO
FALSO	VERDADEIRO	FALSO
FALSO	FALSO	FALSO

Somente quando q e p são VERDADEIROS o resultado é VERDADEIRO



- Então, quando usamos o operador E para compararmos dois valores lógicos, o resultado só vai ser verdadeiro quando os dois valores são verdadeiros.

# Exemplos

```
var
  p, q: logico
inicio
  p <- VERDADEIRO
  q <- VERDADEIRO
  escreval(p E q) // Resultado VERDADEIRO
fimalgoritmo
```

```
var
  p, q: logico
inicio
  p <- FALSO
  q <- VERDADEIRO
  escreval(p E q) // Resultado FALSO
fimalgoritmo
```

```
var
  p, q: logico
inicio
  p <- VERDADEIRO
  q <- FALSO
  escreval(p E q) // Resultado FALSO
fimalgoritmo
```

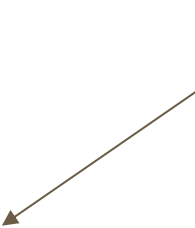
```
var
  p, q: logico
inicio
  p <- FALSO
  q <- FALSO
  escreval(p E q) // Resultado FALSO
fimalgoritmo
```

# Operador OU

- Assim como o operador E, o operador OU compara dois valores lógicos.
- Vamos supor novamente que temos duas variáveis p e q, dependendo do valor dessas variáveis o resultado pode ser VERDADEIRO ou FALSO:

p	q	p OU q
VERDADEIRO	VERDADEIRO	VERDADEIRO
VERDADEIRO	FALSO	VERDADEIRO
FALSO	VERDADEIRO	VERDADEIRO
FALSO	FALSO	FALSO

Somente quando q e p são FALSOS o resultado é FALSO



- Então, quando usamos o operador OU para compararmos dois valores lógicos, o resultado vai ser verdadeiro se qualquer um dos dois valores for verdadeiro.

# Exemplos

```
var
  p, q: logico
inicio
  p <- VERDADEIRO
  q <- VERDADEIRO
  escreval(p OU q) // Resultado VERDADEIRO
fimalgoritmo
```

```
var
  p, q: logico
inicio
  p <- FALSO
  q <- VERDADEIRO
  escreval(p OU q) // Resultado VERDADEIRO
fimalgoritmo
```

```
var
  p, q: logico
inicio
  p <- VERDADEIRO
  q <- FALSO
  escreval(p OU q) // Resultado VERDADEIRO
fimalgoritmo
```

```
var
  p, q: logico
inicio
  p <- FALSO
  q <- FALSO
  escreval(p OU q) // Resultado FALSO
fimalgoritmo
```



# Operadores Lógicos

- Para compreender melhor os operadores **E**, basta se fazer a seguinte pergunta:

ValorA **E** ValorB são verdadeiros?

Se, e somente se, os dois forem verdadeiros o resultado será VERDADEIRO.

- Para o operador OU, da mesma forma:

ValorA **OU** ValorB são verdadeiros?

Se, o ValorA ou o ValorB (tanto faz) for verdadeiro, o resultado será VERDADEIRO, mas se ambos forem FALSO o resultado será FALSO.



# Operador NAO (Não)

- Sugestivamente o operador **NAO** inverte o valor do tipo lógico
  - Em outras palavras, se o valor for **VERDADEIRO** o **NAO VERDADEIRO** vai ser **FALSO**, assim como o **NAO FALSO** será **VERDADEIRO**.
- Por exemplo:

**NAO VERDADEIRO** // O resultado será FALSO

**NAO FALSO** // O resultado será VERDADEIRO

# Tabela Verdade

- Abaixo segue a Tabela Verdade que mostra, de forma geral, como os operadores lógicos funcionam:

p	q	$p \text{ E } q$	$p \text{ OU } q$
VERDADEIRO	VERDADEIRO	VERDADEIRO	VERDADEIRO
VERDADEIRO	FALSO	FALSO	VERDADEIRO
FALSO	VERDADEIRO	FALSO	VERDADEIRO
FALSO	FALSO	FALSO	FALSO

p	NAO p
VERDADEIRO	FALSO
FALSO	VERDADEIRO

# Expressões com Operadores Lógicos e Relacionais

- Podemos combinar os operadores lógicos com os operadores relacionais para formar expressões lógicas mais complexas:

$(2 > 3) \text{ E } (23 \geq 30)$  // O resultado é FALSO

$(1 + 2 \geq 3) \text{ OU } (3 - 2 \leq 1)$  // O resultado é verdadeiro

operação relacional      operação relacional

operação lógica

- É importante observar que: sempre devemos separar operações relacionais de operações lógicas com o uso dos “( )” parênteses, para não confundir ou gerar algum erro.

# Ordem de Precedência

- Quando se combinam operadores relacionais nas expressões, em conta a ordem de operadores relacionais e

1. Operações seguindo a mesma ordem expressões aritméticas.

2. Operadores relacionais com ordem esquerda para a direita.

3. Operadores lógicos, onde o E é verificado primeiro, do OU e por último o

Ordem	Operadores	
1º	lógicos	() e
2º	se	potência e raiz
3º	lógicos	Aritiméticos
4º	de precedência	das + -
5º	de importância	Relacionais
6º	primeiro,	E
7º	é Lógicos	OU
8º		NAO

# Exercício

1. Observe as expressões abaixo e responda se seu resultado é verdadeiro ou falso.
  - a.  $(12 > 10) \text{ E } (10 > 12)$
  - b.  $(32 * (4 - 2) > 69) \text{ OU } (30 * (4 - 2) > 69)$
  - c. VERDADEIRO E FALSO OU VERDADEIRO
  - e.  $((32 / 4) - 3 = 5) \text{ E } ((26 + 6) / 4 = 5)$
  - f.  $\text{NAO}(\text{FALSO OU } (14 / 2 * 3 \geq 21))$

# Aprenda Mais...

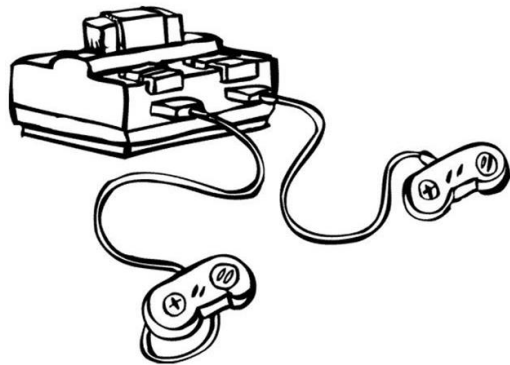
- Operadores Lógicos e Relacionais - Curso de Algoritmos #04 - Gustavo Guanabara  
<<https://www.youtube.com/watch?v=lg4QZNpVZYs>>
- Visualg Aula 8 - Operadores Lógicos  
<<https://www.youtube.com/watch?v=yEvnbyjjc0k>>
- Victor Schinaider - Primeiros Passos - Álgebra Booleana - Video 3  
<<https://www.youtube.com/watch?v=mYv71G-lpZw>>

# Estrutura Condicional



# Condições

- As condições estão presentes na nossa vida o dia e sempre aparecem quando precisamos tomar alguma decisão sobre algo.
- Por exemplo, na afirmação abaixo:  
**Se tivermos dinheiro o suficiente, compramos um novo videogame novo.**
- Nesse exemplo, a condição para que possamos comprar um videogame novo é termos dinheiro para comprá-lo.
  - Considere que o videogame novo custe 2500 reais, se tivermos uma quantia igual ou maior que essa, poderemos comprá-lo.
- Na programação, as condições servem para definir se uma ação será realizada ou não, quando uma certa condição for atendida.



# Condições na Programação

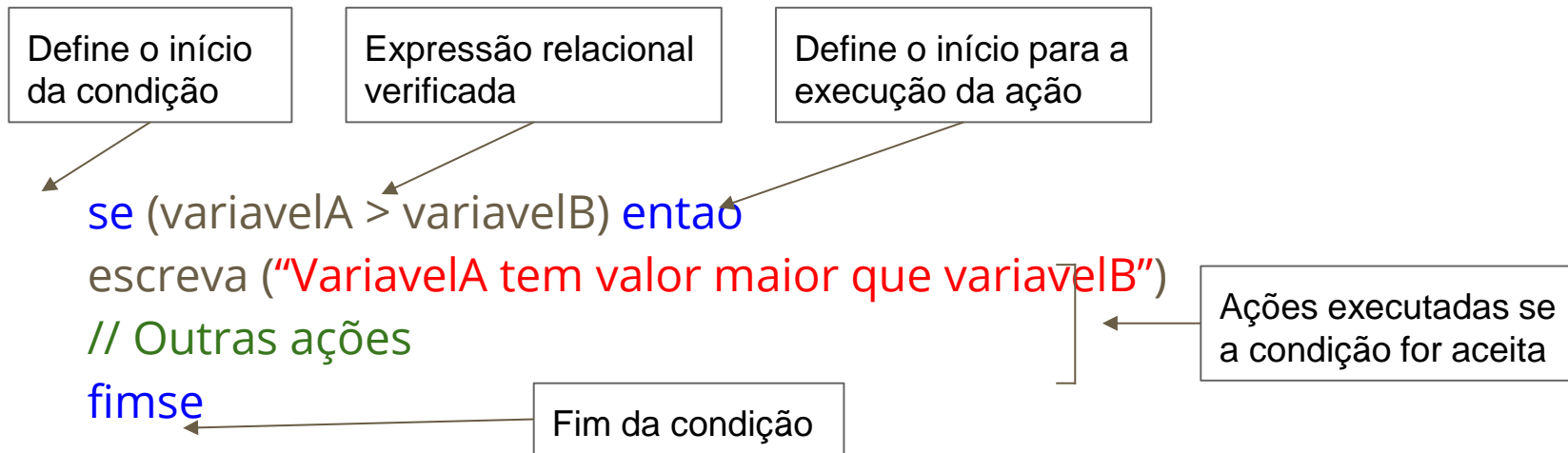
- Basicamente, em programação, os comandos condicionais necessitam de uma expressão lógica ou relacional para “mudar” o fluxo do programa, a partir da veracidade ou não de tal expressão.
- O **se... então** é o comando condicional mais simples
- Por exemplo, se quisermos que algo ocorra somente se o valor de uma variável (variavelA) for maior que o valor de outra variável (variavelB), podemos fazer:

**se** (variavelA > VariavelB) **então** // Lê-se: Se variavelA for maior que variavelB...

- No caso, se supormos que a variavelA tem o valor 2 e a variavelB tem valor 1, a condição será verdadeira.

# Estrutura Condicional Simples

- Quando uma condição é VERDADEIRA, o programa executará tudo que fizer parte da estrutura dessa condição:



- No caso, tudo que está depois do **ENTAO** e antes do **FIMSE**, será executado se a condição que está sendo testada resultar em **VERDADEIRO**.

# Estrutura Condicional Simples

- Então, essa estrutura que vimos chama-se **condicional simples**, que é composta por:
  1. O comando inicial **se**;
  2. Logo após alguma expressão lógica ou relacional entre () (parênteses) - que resulte em um valor lógico;
  3. Depois vem a palavra reservada **entao**;
  4. Após o **entao**, nas próximas linhas, colocam-se os comandos que se quer executar se o resultado da expressão for **verdadeiro**, ou seja, se a condição for aceita;
  5. E, quando quisermos finalizar a execução dos comandos da estrutura condicional, usamos o **fimse** (lê-se: fim se) - ele marca o final do bloco condicional **se**;

# Exemplo

- Voltemos a pergunta do início:

**Se tivermos dinheiro suficiente, compramos um videogame novo.**

- Agora, se passarmos para forma de algoritmo, poderíamos fazer:

```
se (nosso_dinheiro >= 2500) entao
    escreval("Vamos jogar o novo Super Mario")
fimse
```

- O código acima poderia ser lido como: *Se nosso dinheiro for maior ou igual a 2500, então escreva a mensagem e finalize a condição.*

# Algoritmo da Maioridade

- Agora, vamos para um exemplo prático onde queremos que um programa receba a idade do usuário e verifique se ele já atingiu a maioridade.

```
algoritmo "algoritmo maioridade"  
  
var  
    idade: inteiro  
inicio  
    escreval("Digite sua idade: ") // Pede ao usuário a idade  
    leia(idade)                     // Atribui esse valor à variável idade  
  
    SE (idade >= 18) ENTÃO          // Se idade for maior ou igual a 18 então...  
        escreva("Você atingiu a maioridade") // escreve a mensagem que atingiu  
    FIMSE                          // Fim da condição SE  
fimalgoritmo
```

# Código Completo

## VisuAlg

**algoritmo** "Maioridade"

**var**

idade: inteiro

**inicio**

// Pede ao usuário a idade  
escreval("Digite sua idade")  
leia(idade)

// Verifica se a idade é maior ou igual a 18  
SE (idade >= 18) ENTAO  
    escreva("Você atingiu a maioridade")

FIMSE

**fimalgoritmo**

## Portugol Studio

**programa**

{

**funcao** inicio()

    {

**inteiro** idade

        // Pede ao usuário a idade

        escreva("Digite sua idade")

        leia(idade)

        // Verifica se a idade é maior ou igual a 18

**se** (idade >= 18) {

            escreva("Você atingiu a maioridade")

        }

    }

}

# SENAO

- Agora, sabemos como construir uma estrutura condicional onde SE uma expressão resultar em VERDADEIRO o programa executa uma série de instruções predefinidas.
- Porém, e se quiséssemos também executar instruções predefinidas caso o resultado dessa expressão não fosse VERDADEIRO?
- O **senao** é o comando que complementa a estrutura da condição SE para executar comandos, caso a condição não seja aceita.
- Por exemplo:

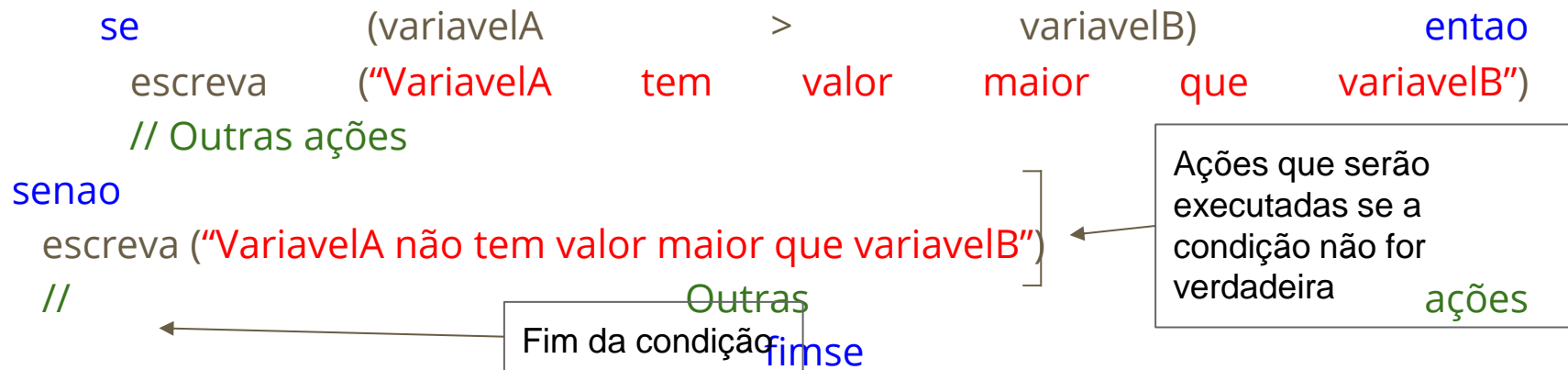
**Se tivermos dinheiro suficiente, compramos um videogame novo...**

**Se não tivermos dinheiro, temos que nos conformar**



# Estrutura Condicional Composta

- Quando a condição abaixo não for **verdadeira**, o programa executará tudo que fizer parte da estrutura do **senao**:



- No caso tudo que está depois do **senao** e antes do **fimse**, será executado se a condição não resultar em **verdadeiro**.

# Estrutura Condicional Composta

- Essa estrutura é chamada de condicional composta e segue a seguinte sintaxe:
  1. Iniciar com o comando **se**;
  2. Logo após uma expressão lógica ou relacional entre () (parênteses);
  3. Depois usamos com o comando **entao**;
  4. Nas próximas linhas colocamos os comandos que queremos executar se o resultado a expressão for **verdadeiro**, ou seja, se a condição for aceita;
  5. Usamos o **senao** (lê-se: se não);
  6. Colocamos os comandos que queremos executar se o resultado a expressão **não** for VERDADEIRO, ou seja, se a condição **não** for aceita;
  7. E para finalizar a execução dos comandos da estrutura da condição, usamos o **fimse**;

# Exemplo

- Voltemos a questão do videogame:

**Se tivermos dinheiro suficiente, compramos um videogame novo.  
Se não tivermos dinheiro, nos conformamos...**

- Agora se passarmos para forma de algoritmo, poderíamos fazer:

```
se (nosso_dinheiro >= 2500) entao
  escreval("Vamos jogar o novo Super Mario")
senao
  escreval("Fazer o quê, né!?")
fimse
```

- O código acima poderia ser lido como: *Se nosso dinheiro for maior ou igual a 2500 então escreva a mensagem, se não escreva a outra mensagem e finalize a condição.*

# Código depois da condição SE/SENAO

É importante entender que, após qualquer estrutura de código (neste caso após as instruções do SE e SENAO), o programa executará normalmente os comandos que estiverem depois dessa estrutura, independente se a condição foi aceita ou não:

```
var
  A, B: inteiro
inicio

A <- 1
B <- 2
SE (A > B) ENTAO                                // Condição não aceita
  escreval("A é maior que B") // Comando ignorado
  // Outros comandos ignorados
SENAO
  escreval("B é maior que A") // Executa esse comando
  // Outros comando que serão executados
FIMSE

// Essa linha é executda normalmente
escreval("Outros comandos")
finalgoritmo
```

# Algoritmo do Ímpar ou Par

- Mais um exemplo prático, onde temos um algoritmo que pede ao usuário um número e ele verifica se esse número é ímpar ou é par

algoritmo "Par ou Impar"

var

    numero: inteiro

inicio

    escreva("Digite um numero: ")

    leia(numero)

    SE ((numero % 2) = 0) ENTÃO // Se o resto da divisão for 0 (zero)

        escreval("é Par")

    SENAO

        escreval("é Impar")

    fimse

fimalgoritmo

# Código Completo

## VisuAlg

**algoritmo** "Par ou Ímpar"

**var**

numero: inteiro

**inicio**

escreva("Digite um número: ")

leia(numero)

SE ((numero % 2) = 0) ENTAO // Se o resto da divisão for 0 (zero)

    escreval("é Par")

SENAO

    escreval("é Ímpar")

FIMSE

**fimalgoritmo**

## Portugol Studio

**programa**

{

**funcao** inicio()

    {

**inteiro** numero

        escreva("Digite um número: ")

        leia(numero)

        // Se o resto da divisão for 0 (zero)

        se ((numero % 2) == 0) {

            escreva("é Par!")

        } senao {

            escreva("é Ímpar!")

        }

    }

}

# Exercício

1. Faça um programa que pede ao usuário três notas de 0 a 10, calcule a média aritmética dessas notas e verifique se a média é maior ou igual a 7, se for o programa deve imprimir a frase “Aluno Aprovado!”, senão o programa deve imprimir “Aluno Reprovado!”.
2. Altere o programa anterior para mostrar no final as notas do aluno e a média calculada.
3. Altere o programa anterior para calcular a média ponderada ao invés da média aritmética.

# Aprenda mais...

- Estruturas Condicionais 1 - Curso de Algoritmos #07 - Gustavo Guanabara <[https://www.youtube.com/watch?v=\\_g05aHdBAEY](https://www.youtube.com/watch?v=_g05aHdBAEY)>
- VisuALG #5 – Desvio condicional composto (SE... ENTÃO... SENÃO... FIMSE) <<https://andreysmith.wordpress.com/2014/05/18/visualg-5-desvio-condicional-composto-se-entao-senao-fimse/>>
- JovemProgramadorBR - Lógica de Programação com VisualG - Estrutura de Seleção ou Decisão - 02 <<https://www.youtube.com/watch?v=mmHfui8nenw>>
- Aula 9 - Estruturas Condicionais <<https://pt.slideshare.net/LuizAugustoMacdoMoraes/aula-9-estruturas-condicionais>>
- Raphael H. Vieira - Lógica de Programação - Aula3 Estrutura Condicional (SE, ENTÃO, SENÃO, FIMSE) <<https://www.youtube.com/watch?v=QlioTTYRxrg>>



# Condicionais Aninhadas

- Anteriormente, foi apresentada a estrutura condicional composta, que oferece duas escolhas
  1. Se a expressão for verdadeira, faça algo;
  2. Se a expressão não for verdadeira, faça outra coisa;
- Porém, não é sempre que vamos nos deparar com apenas duas escolhas. Voltando ao dilema de comprar um videogame novo, por exemplo:

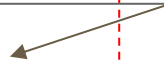
**Se tivermos dinheiro suficiente, compramos um videogame novo...**  
**Senão, se tivermos algum dinheiro, compramos um videogame usado...**  
**Se Não, nos conformamos... :(**

# Condicionais Aninhadas

- Como observado, nós temos três possibilidades de decisão que dependem se a condição for aceita ou não.

```
se (nosso_dinheiro >= 2500) entao
    escreval("Vamos comprar um Playstation 4")
senao
    se (nosso_dinheiro >= 1500) entao
        escreval("Vamos comprar um Playstation 3")
    senao
        escreval("Fazer o quê, né!?")
fimse
fimse
```

Criamos outra condicional dentro do SENAO. Caso a primeira condição não seja aceita, o programa irá para o bloco do SENÃO e testará a segunda condicional..



# Condicionais Aninhadas

- Basicamente, as condicionais aninhadas são condições que colocamos dentro de outras condições (daí vem o nome aninhadas), possibilitando fazer verificações mais complexas e com mais opções de decisão.
- No caso do videogame, usamos duas condicionais compostas da seguinte forma:
  - Verificamos se a variável `nosso_dinheiro` tinha o valor igual ou maior que 2500
  - Se o resultado fosse **verdadeiro**, imprimiria **"Vamos comprar um Playstation 4"**
  - Se não, Verificamos se a variável `nosso_dinheiro` tinha o valor igual ou maior que 1500 e menor que 2500
  - Se o resultado fosse **verdadeiro**, imprimiria **"Vamos comprar um Playstation 3"**
  - Se não, imprimiria **"Fazer o quê, né!?"**

# Algoritmo Situação Aluno

Vamos ver um exemplo mais prático: o usuário fornece a média do aluno o programa deve imprimir “aprovado” se a média maior ou igual a 7; “em recuperação” se a média for menor que 7, porém maior ou igual a 5; e “reprovado” se a média for menor que 5.

```
algoritmo "SituacaoAluno"  
var  
    media: real  
inicio  
    // Recebe a media do aluno  
    escreva("Digite a média do aluno")  
    leia(media)  
  
    // Verifica se a média é maior ou igual a 7  
    SE (media >= 7) ENTAO  
        escreva("Aprovado!")  
    SENAO // Se não for...  
        // Verifica se a média é menor que 7 e maior ou igual a 5  
        SE (media < 7 E media >= 5) ENTAO  
            escreva("Em recuperação")  
        SENAO // Se não for...  
            escreva("Reprovado")  
        FIMSE // Fim do SE  
    FIMSE // Fim do SE  
fimalgoritmo
```

# Código Completo

## VisuAlg

**algoritmo** "SituacaoAluno"

**var**

media: real

**inicio**

// Recebe a média do aluno

escreva("Digite a média do aluno ")

leia(media)

// Verifica se a média é maior ou igual a 7

SE (media  $\geq$  7) ENTAO

escreva("Aprovado!")

SENAO // Se não for...

// Verifica se a média é menor que 7 e maior ou igual a 5

SE (media < 7 E media  $\geq$  5) ENTAO

escreva("Em recuperação")

SENAO // Se não for...

escreva("Reprovado")

FIMSE // Fim do SE

FIMSE // Fim do SE

**fimalgoritmo**

## Portugol Studio

**programa** {

**funcao** inicio() {

**real** media

// Recebe a média do aluno

escreva("Digite a média do aluno ")

leia(media)

// Verifica se a média é maior ou igual a 7

se (media  $\geq$  7) {

escreva("Aprovado")

} senao {

se (media < 7 e media  $\geq$  5) {

escreva("Em recuperação")

} senao {

escreva("Reprovado")

}

}

}

}

# Aprenda mais

- Evandro Júnior - Estrutura condicional Se.. senao  
<<https://www.youtube.com/watch?v=bqW4TWBLROY>>
- Canal Byte - 05 - Resolvendo exercícios de SE...ENTAO  
<<https://www.youtube.com/watch?v=JMw9GQleQIE>>
- One Day Code - Lógica de Programação Para Iniciantes (Condicionais Múltiplas)- VisuALG #02 <<https://youtu.be/VzefcnQBuQk>>

# ESCOLHA-CASO

- Com a estrutura de condicional aninhada podemos verificar várias condições para várias possibilidades de saídas
- Porém, quando existem muitas condições, ficar usando várias estruturas **se**, uma dentro da outra, pode ser muito trabalhoso, além de dificultar a leitura e manutenção do código, por parte de outros programadores.
- Para evitar isso temos a opção de usar a estrutura ESCOLHA-CASO.
- O **escolha** serve para quando temos muitas comparações com valores pré-estabelecidos. Basicamente, se verifica se uma variável está com o mesmo valor que algum dos casos definidos no código. Quando isso acontecer, executa-se o código referente a esse caso.



# Escolha caso...

- Então, o `escolha` executa determinada ação `caso` a variável possua determinado valor:

```
escolha(variavel) // Variavel a ser testada
    caso 1 // O 1 é o valor que queremos verificar
        // Instruções caso a variável possua o valor 1
    caso 2 // O valor testado agora é o 2
        // Instruções caso a variável possua o valor 2
    // Mais casos: CASO 3, CASO 4 ...
    outrocaso
        // Instruções caso a variável não possua nenhum dos valores
        testados
    fimescolha // Finaliza a estrutura ESCOLHA
```

- O comando `outrocaso` é opcional, somente para as situações onde queremos realizar algo caso a nossa variável não possua nenhum dos valores testados nos outros `caso`.



# Algoritmo da Calculadora Básica

- Para entender melhor, vamos construir passo-a-passo um algoritmo que vai simular uma calculadora com as operações básicas: adição subtração, multiplicação e divisão.

1. Primeiro vamos declarar as variáveis, serão quatro: uma para cada número da operação do tipo real, uma para receber o tipo de operador (+, -, \* ou /) do tipo caractere e uma para receber o resultado da operação do tipo real.

var

numero1, numero2: real

operador: caractere

resultado: real

inicio

.....

# Algoritmo da Calculadora Básica

2. Depois, vamos pedir ao usuário os valores dos números, e o tipo de operação que vai ser:

início

escreva("Digite o primeiro número")

leia(numero1)

escreva("Digite o tipo de operador [ +, -, \*, / ]")

leia(operador)

\_\_\_\_\_ escreva("Digite o segundo número")

leia(numero2)

\_\_\_\_\_.....

Basicamente, queremos pegar os valores passados e formar uma oração aritmética com o numero1 operador numero2, tipo numero1 + numero2 ou numero1 - numero2 etc.

# Algoritmo da Calculadora Básica

3. Agora o mais importante, vamos usar a estrutura ESCOLHA-CASO para verificar o valor do operado recebido pelo usuário, em cada caso

escolha (operador)

caso "+" // Caso o valor do operador for +

resultado <- numero1 + numero2 // O resultado recebe a soma dos números

caso "-" // Caso o valor do operador for -

resultado <- numero1 - numero2 // O resultado recebe a subtração do numero1 pelo

numero2

caso "\*" // Caso o valor do operador for \*

resultado <- numero1 \* numero2 // O resultado recebe a multiplicação dos números

caso "/" // Caso o valor do operador for /

resultado <- numero1 / numero2 // O resultado recebe a divisão do numero1 pelo

numero2

fimsescolha

# Algoritmo da Calculadora Básica


4. Por último, escrevemos na tela o valor da variável resultado que foi calculada em algum dos casos do ESCOLHA.

```
escreva ("O resultado é: ", resultado)
```

finalgoritmo

- Com isso, temos nosso programa de calculadora básica que, ao ser executado, vai pedir para o usuário um número, depois um operador e outro número. Em seguida, é verificado o tipo de operação com base no valor do operador digitado, essa operação é feita e seu resultado é atribuído a uma variável cujo valor será impresso no final.


# Código Completo

VisuAlg	
<p><b><u>algoritmo</u></b> "CalculadoraBasicaComSE"</p> <p><b><u>var</u></b></p> <p>numero1 : <u>real</u></p> <p>numero2 : <u>real</u></p> <p>operador: <u>caractere</u></p> <p>resultado : <u>real</u></p> <p><b><u>inicio</u></b></p> <p>escreva ("Digite o primeiro número: ")</p> <p>leia (numero1)</p> <p>escreva ("Digite a operação: ")</p> <p>leia (operador)</p> <p>escreva ("Digite o segundo número: ")</p> <p>leia (numero2)</p> 	<p>ESCOLHA (operador)</p> <p>CASO "+"</p> <p>resultado &lt;- numero1 + numero2</p> <p>CASO "-"</p> <p>resultado &lt;- numero1 - numero2</p> <p>CASO "**"</p> <p>resultado &lt;- numero1 * numero2</p> <p>CASO "/"</p> <p>resultado &lt;- numero1 / numero2</p> <p>FIMESCOLHA</p> <p>ESCREVA ("Resultado: ", resultado)</p> <p><b><u>fimalgoritmo</u></b></p>

# Código Completo

## Portugol Studio

```
programa {  
  
    funcao inicio() {  
  
        real numero1, numero2  
        caracter operador  
        real resultado  
  
        escreva ("Digite o primeiro número: ")  
        leia(numero1)  
        escreva ("Digite a operação: ")  
        leia(operador)  
        escreva ("Digite o segundo número: ")  
        leia(numero2)  
  
        escolha (operador) {  
            caso '+':  
                resultado = numero1 +  
numero2  
            pare  
            caso '-':  
                resultado = numero1 -  
numero2  
            pare  
            caso '*':  
                resultado = numero1 *  
numero2  
            pare  
            caso '/':  
                resultado = numero1 /  
numero2  
        }  
  
        escreva("O resultado é: " + resultado)  
    }  
}
```



# Exercício

1. Faça um programa que peça ao usuário três notas de 0 a 10, calcule a média aritmética dessas notas e verifique se a média é maior ou igual a 7. Caso seja, o programa deve imprimir a frase “Aluno Aprovado!”, se a média for menor que 7, mas maior ou igual a 4, o programa deve imprimir “Aluno em Recuperação”, se não, o programa deve imprimir “Aluno Reprovado!”.
2. Crie um programa no qual se pede ao usuário um valor inteiro e o programa deve imprimir o mês do ano correspondente ao valor recebido (ex. recebendo o valor 1, o programa imprime “Janeiro”; recebendo o valor 2, o programa imprime “Fevereiro” e ect.)

# Aprenda Mais...

- Estrutura de seleção múltipla ESCOLHA-CASO

<<http://www.dicasdeprogramacao.com.br/estrutura-de-selecao-multipla-escolha-caso/>>

- Estruturas Condicionais 2 - Curso de Algoritmos #08 - Gustavo Guanabara  
<<https://youtu.be/7gGFHzqh4d8>>
- Tuto Studio - VisuAlg Aula 12 - Escolha Caso - <<https://youtu.be/5FNebG7sBP4>>
- One Day Code - Lógica de Programação Para Iniciantes (Escolha Caso) - VisuALG #04  
<<https://youtu.be/uT4y7NXAprA>>