

# Algoritmos

## Desvios Case

Prof. Dr. Osmar Betazzi Dordal

7

# Desvios Case

# Desvios Condicionais

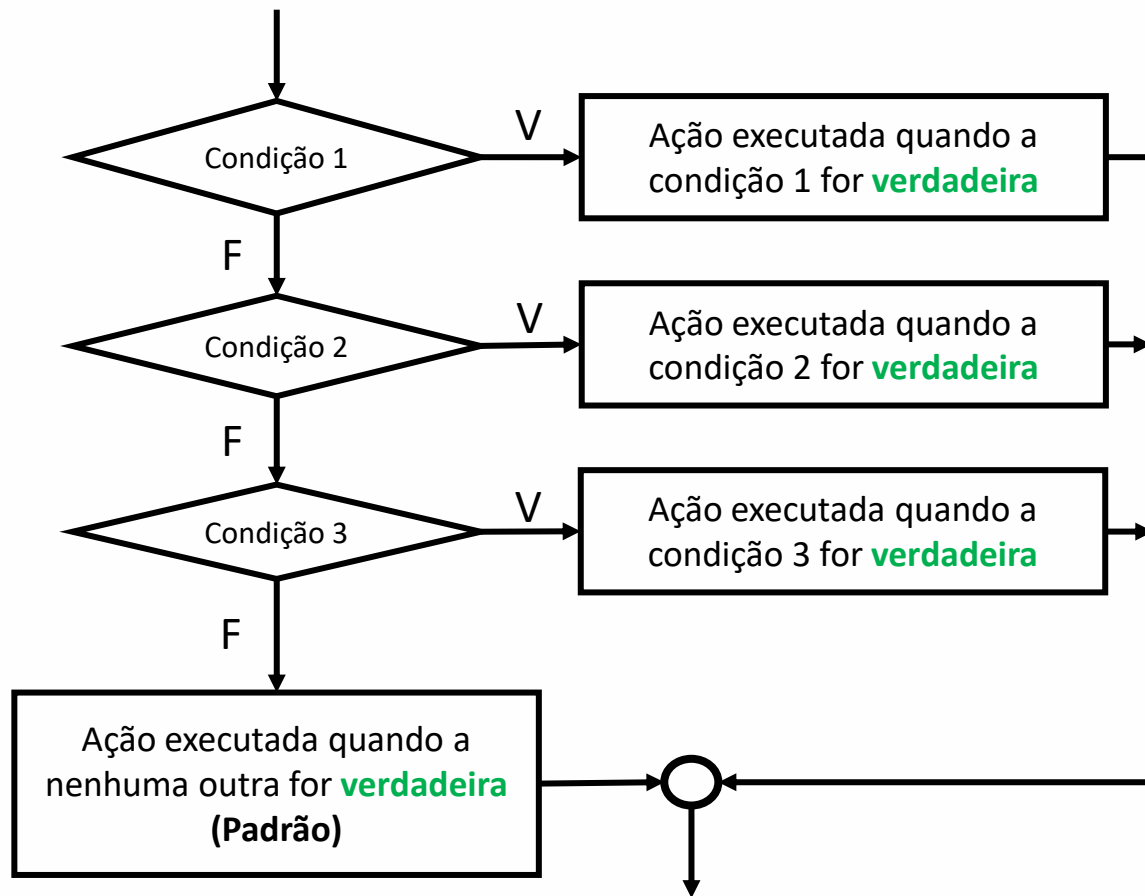
- As estruturas de desvios são utilizadas em algoritmos para efetuar uma tomada de decisão simples;
- Os símbolos de diagrama de blocos para a tomada de decisão são:

- Decisão: 

- Conexão 

- Ação 

# Desvios com Múltiplas Escolhas – *Case*



```
caso <variável>  
  seja <condição 1> faça  
    [Ação executada quando a  
    condição 1 for verdadeira]  
  seja <condição 2> faça  
    [Ação executada quando a  
    condição 2 for verdadeira]  
  seja <condição 3> faça  
    [Ação executada quando a  
    condição 3 for verdadeira]  
senão  
  [Ação executada quando nenhuma  
  condição for verdadeira]  
fim_caso
```

# Desvios com Múltiplas Escolhas – *Case*

```
1 Program Caso;  
2 var  
3   opcao: integer;  
4 Begin  
5  
6   write('Entre com uma opção: ');  
7   readln(opcao);  
8  
9   case opcao of  
10    1: writeln('você escolheu a opção 1');  
11    2: writeln('você escolheu a opção 2');  
12    3: writeln('você escolheu a opção 3');  
13    else writeln ('Você escolheu uma opção diferente de 1, 2, 3...' );  
14  end;  
15 End.
```

```
1 Program Caso2;  
2 var  
3   opcao: char;  
4 Begin  
5  
6   write('Entre com uma opção: ');  
7   readln(opcao);  
8  
9   case opcao of  
10    'a': writeln('você escolheu a opção a');  
11    'b': writeln('você escolheu a opção b');  
12    'c': writeln('você escolheu a opção c');  
13    else writeln ('Você escolheu uma opção diferente de a, b, c...' );  
14  end;  
15 End.
```

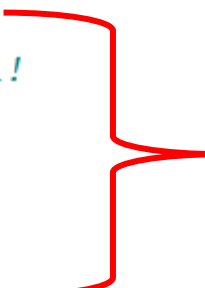
```
case integer of  
    1: código;  
    2: código;  
    ...: código;  
    n: código;  
    else código;  
end;
```

```
case char of  
    'a': código;  
    'b': código;  
    '...': código;  
    'z': código;  
    else código;  
end;
```

Podemos usar alguns símbolos

# Desvios com Múltiplas Escolhas – *Case*

```
1 Program Caso;
2 var
3   opcao, x: integer;
4 Begin
5
6   write('Entre com uma opção: ');
7   readln(opcao);
8   write('Entre com um valor inteiro para x: ')
9   readln(x);
10
11  case opcao of
12    1: begin
13      //posso escrever um código complexo aqui!
14      if(x >= 5) then
15        writeln('x é maior ou igual a 5')
16      else
17        writeln('x é menor que 5');
18    end;
19    2: writeln('você escolheu a opção 2 e por isso não falaremos da variável x');
20    3: writeln('você escolheu a opção 3 e por isso não falaremos da variável x');
21    else writeln ('Você escolheu uma opção diferente de 1, 2, 3...' );
22  end;
23 End.
```

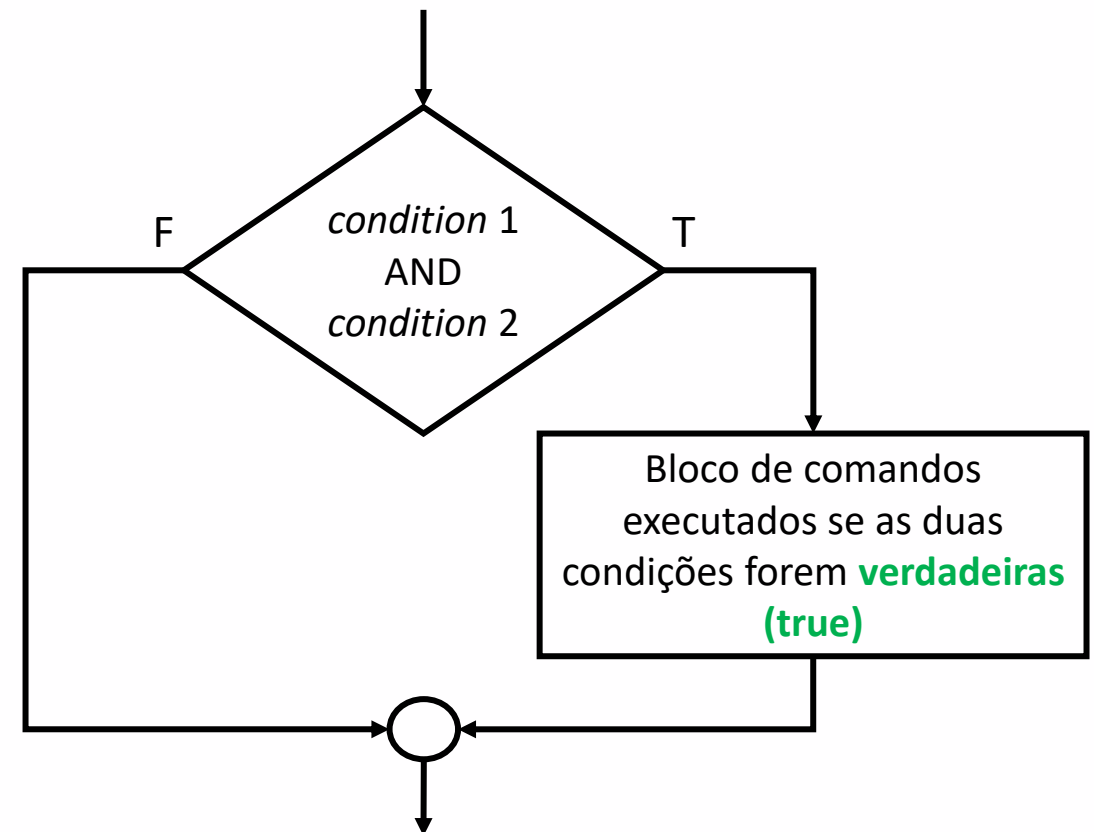


# Operadores Lógicos (E) (*and*)

- Com a utilização do operador logico (e/*and*), **todas as condições da operação devem ser verdadeiras** para que o bloco de comandos seja executado:

- Tabela verdade disjuntiva (E) / (*and*)

<i>condition 1</i>	<i>condition 2</i>	( <i>condition 1</i> ) <b>AND</b> ( <i>condition 2</i> )
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE



# Operadores Lógicos (E) (*and*)

- Com a utilização do operador logico (e/*and*), todas as condições da operação devem ser verdadeiras para que o bloco de comandos seja executado:
  - Tabela verdade disjuntiva (E) / (*and*)

Média	Faltas	<i>media</i>	<i>faltas</i>	<i>(media</i> >= 7.0) <b>AND</b> ( <i>faltas</i> < 20)
8.0	30	TRUE	FALSE	FALSE
5.0	15	FALSE	TRUE	FALSE
7.0	10	TRUE	TRUE	TRUE

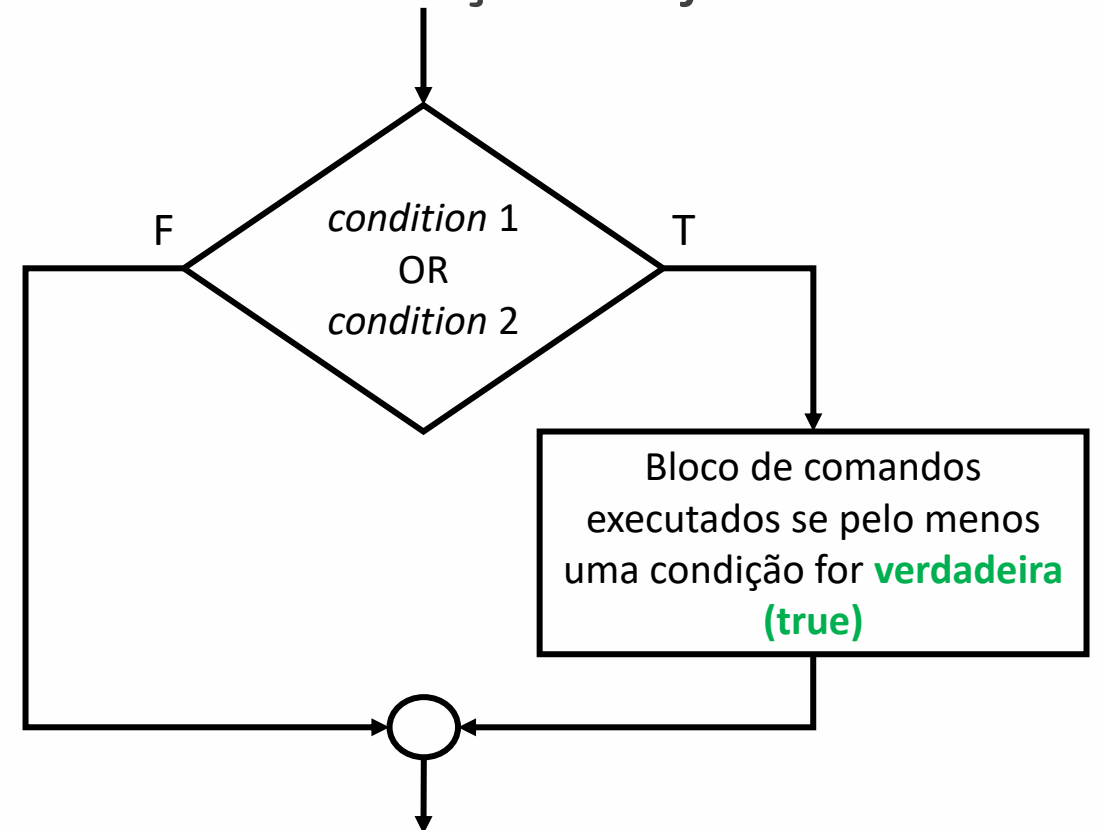
```
1 Program Operador_logico_E ;
2 var
3     faltas: integer;
4     media: real;
5
6 Begin
7
8     write('entre com o valor da média: ');
9     readln(media);
10
11    write('entre com o o número de faltas: ');
12    readln(faltas);
13
14    if(media >= 7.0) and (faltas < 20) then
15        writeln('Aprovado')
16    else
17        writeln('Reprovado');
18
19 End.
```



# Operadores Lógicos (OU) (*or*)

- O operador logico (ou/*or*) será utilizado em situações nas quais **basta qualquer uma das condições ser verdadeira** para que o resultado também seja verdadeiro e, conseqüentemente, o bloco de instruções seja executado:
  - Tabela verdade conjuntiva (OU) / (*or*)

<i>condition 1</i>	<i>condition 2</i>	<i>(condition 1) OR (condition 2)</i>
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE



# Operadores Lógicos (OU) (*or*)

- O operador logico (ou/*or*) será utilizado em situações nas quais **basta qualquer uma das condições ser verdadeira** para que o resultado também seja verdadeiro e, conseqüentemente, o bloco de instruções seja executado:
  - Tabela verdade conjuntiva (OU) / (*or*)

Código	codigo = 100	codigo = 200	(codigo = 100) <b>OR</b> (codigo = 200)
50	FALSE	FALSE	FALSE
100	TRUE	FALSE	TRUE
200	FALSE	TRUE	TRUE

```
1 Program Operador_logico_OU ;
2 var
3     codigo: integer;
4
5 Begin
6
7     write('digite o valor do código: ');
8     readln(codigo);
9
10    if(codigo = 100) or (codigo = 200) then
11        writeln('Código válido!')
12    else
13        writeln('Código inválido!');
14
15 End.
```

# Operadores Lógicos (NÃO) (*not*)

- O operador logico (**não/not**), de negação, será usado quando for necessário inverter o valor de uma condição, ou seja, tornar verdadeiro o que era falso e tornar falso o que era verdadeiro :
  - Tabela verdade conjuntiva (NÃO) / (*not*)

condition	<b>NOT</b> (condition)
TRUE	FALSE
FALSE	TRUE

```
1 Program Operador_logico_NOT ;
2 var
3     numero, resto: integer;
4
5 Begin
6
7     write('digite um número par: ');
8     readln(numero);
9
10    resto := (numero)mod(2);
11    if not(resto = 0) then
12        writeln('O número ', numero , ' é ímpar')
13    else
14        writeln('O número ', numero , ' é par');
15
16 End.
```

# Operadores Lógicos – Tabelas Verdade (*and*)

```
1 Program Tabela_Verdade_E;  
2 var  
3   p, q: boolean;  
4  
5 Begin  
6  
7   //V E V  
8   p := TRUE; q := TRUE;  
9   if (p and q) then  
10     writeln('p: ',p,' _E_ ', 'q: ',q);  
11  
12   //V E F  
13   p := TRUE; q := FALSE;  
14   if (p and q) then  
15     writeln('p: ',p,' _E_ ', 'q: ',q);  
16  
17   //F E V  
18   p := FALSE; q := TRUE;  
19   if (p and q) then  
20     writeln('p: ',p,' _E_ ', 'q: ',q);  
21  
22   //F E F  
23   p := FALSE; q := FALSE;  
24   if (p and q) then  
25     writeln('p: ',p,' _E_ ', 'q: ',q);  
26  
27 End.
```

Qual a saída?

# Operadores Lógicos – Tabelas Verdade (*and*)

```
1 Program Tabela_Verdade_E;  
2 var  
3   p, q: boolean;  
4  
5 Begin  
6  
7   //V E V  
8   p := TRUE; q := TRUE;  
9   if (p and q) then  
10    writeln('p: ',p,' _E_ ', 'q: ',q);  
11  
12   //V E F  
13   p := TRUE; q := FALSE;  
14   if (p and q) then  
15    writeln('p: ',p,' _E_ ', 'q: ',q);  
16  
17   //F E V  
18   p := FALSE; q := TRUE;  
19   if (p and q) then  
20    writeln('p: ',p,' _E_ ', 'q: ',q);  
21  
22   //F E F  
23   p := FALSE; q := FALSE;  
24   if (p and q) then  
25    writeln('p: ',p,' _E_ ', 'q: ',q);  
26  
27 End.
```

Qual a saída?

p: TRUE \_E\_ q: TRUE

<i>p</i>	<i>q</i>	<i>AND</i>
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

# Operadores Lógicos – Tabelas Verdade (*and*)

```
1 Program Tabela_Verdade_E;  
2 var  
3   p, q: boolean;  
4  
5 Begin  
6  
7   //V E V  
8   p := TRUE; q := TRUE;  
9   if (p and q) then  
10    writeln('p: ',p,' _E_ ', 'q: ',q);  
11  
12   //V E F  
13   p := TRUE; q := FALSE;  
14   if (p and q) then  
15    writeln('p: ',p,' _E_ ', 'q: ',q);  
16  
17   //F E V  
18   p := FALSE; q := TRUE;  
19   if (p and q) then  
20    writeln('p: ',p,' _E_ ', 'q: ',q);  
21  
22   //F E F  
23   p := FALSE; q := FALSE;  
24   if (p and q) then  
25    writeln('p: ',p,' _E_ ', 'q: ',q);  
26  
27 End.
```

Qual a saída?

p: TRUE \_E\_ q: TRUE

Qual a linha?

# Operadores Lógicos – Tabelas Verdade (*and*)

```
1 Program Tabela_Verdade_E;  
2 var  
3   p, q: boolean;  
4  
5 Begin  
6  
7   //V E V  
8   p := TRUE; q := TRUE;  
9   if (p and q) then  
10    writeln('p: ',p,' _E_ ', 'q: ',q);  
11  
12   //V E F  
13   p := TRUE; q := FALSE;  
14   if (p and q) then  
15    writeln('p: ',p,' _E_ ', 'q: ',q);  
16  
17   //F E V  
18   p := FALSE; q := TRUE;  
19   if (p and q) then  
20    writeln('p: ',p,' _E_ ', 'q: ',q);  
21  
22   //F E F  
23   p := FALSE; q := FALSE;  
24   if (p and q) then  
25    writeln('p: ',p,' _E_ ', 'q: ',q);  
26  
27 End.
```

Qual a saída?

p: TRUE \_E\_ q: TRUE

Qual a linha?

# Operadores Lógicos – Tabelas Verdade (*and*)

```
1 Program Tabela_Verdade_E;  
2 var  
3   p, q: boolean;  
4  
5 Begin  
6  
7   //V E V  
8   p := TRUE; q := TRUE;  
9   if (p and q) then  
10    writeln('p: ',p,' _E_ ', 'q: ',q);  
11  
12   //V E F  
13   p := TRUE; q := FALSE;  
14   if not(p and q) then  
15    writeln('p: ',p,' _E_ ', 'q: ',q);  
16  
17   //F E V  
18   p := FALSE; q := TRUE;  
19   if not(p and q) then  
20    writeln('p: ',p,' _E_ ', 'q: ',q);  
21  
22   //F E F  
23   p := FALSE; q := FALSE;  
24   if not(p and q) then  
25    writeln('p: ',p,' _E_ ', 'q: ',q);  
26  
27 End.
```

Saída?

p: TRUE \_E\_ q: TRUE  
p: TRUE \_E\_ q: FALSE  
p: FALSE \_E\_ q: TRUE  
p: FALSE \_E\_ q: FALSE

Por que?



# Operadores Lógicos – Tabelas Verdade (and)

```
1 Program Tabela_Verdade_E;  
2 var  
3   p, q: boolean;  
4  
5 Begin  
6  
7   //V E V  
8   p := TRUE; q := TRUE;  
9   if (p and q) then  
10    writeln('p: ',p,' _E_ ', 'q: ',q);  
11  
12   //V E F  
13   p := TRUE; q := FALSE;  
14   if not(p and q) then  
15    writeln('p: ',p,' _E_ ', 'q: ',q);  
16  
17   //F E V  
18   p := FALSE; q := TRUE;  
19   if not(p and q) then  
20    writeln('p: ',p,' _E_ ', 'q: ',q);  
21  
22   //F E F  
23   p := FALSE; q := FALSE;  
24   if not(p and q) then  
25    writeln('p: ',p,' _E_ ', 'q: ',q);  
26  
27 End.
```

Saída?

p: TRUE \_E\_ q: TRUE  
p: TRUE \_E\_ q: FALSE  
p: FALSE \_E\_ q: TRUE  
p: FALSE \_E\_ q: FALSE

Por que?

*not*

<i>p</i>	<i>q</i>	<i>AND</i>	<i>NOT</i>
TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	Not(FALSE)
FALSE	TRUE	FALSE	Not(FALSE)
FALSE	FALSE	FALSE	Not(FALSE)

# Operadores Lógicos – Tabelas Verdade (or)

Qual a saída?

```
1 Program Tabela_verdade_OU;
2 var
3   p, q: boolean;
4
5 Begin
6
7   //V OU V
8   p := TRUE; q := TRUE;
9   if (p or q) then
10     writeln('p: ',p, ' _OU_ ', 'q: ',q);
11
12   //V OU F
13   p := TRUE; q := FALSE;
14   if (p or q) then
15     writeln('p: ',p, ' _OU_ ', 'q: ',q);
16
17   //F OU V
18   p := FALSE; q := TRUE;
19   if (p or q) then
20     writeln('p: ',p, ' _OU_ ', 'q: ',q);
21
22   //F OU F
23   p := FALSE; q := FALSE;
24   if (p or q) then
25     writeln('p: ',p, ' _OU_ ', 'q: ',q);
26
27 End.
```

# Operadores Lógicos – Tabelas Verdade (or)

```
1 Program Tabela_verdade_OR;  
2 var  
3   p, q: boolean;  
4  
5 Begin  
6  
7   //V OU V  
8   p := TRUE; q := TRUE;  
9   if (p or q) then  
10    writeln('p: ', p, ' _OU_ ', 'q: ', q);  
11  
12  //V OU F  
13  p := TRUE; q := FALSE;  
14  if (p or q) then  
15    writeln('p: ', p, ' _OU_ ', 'q: ', q);  
16  
17  //F OU V  
18  p := FALSE; q := TRUE;  
19  if (p or q) then  
20    writeln('p: ', p, ' _OU_ ', 'q: ', q);  
21  
22  //F OU F  
23  p := FALSE; q := FALSE;  
24  if (p or q) then  
25    writeln('p: ', p, ' _OU_ ', 'q: ', q);  
26  
27 End.
```

Qual a saída?

p: TRUE \_OU\_ q: TRUE  
p: TRUE \_OU\_ q: FALSE  
p: FALSE \_OU\_ q: TRUE

<i>p</i>	<i>q</i>	<i>OR</i>
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

# Operadores Lógicos – Tabelas Verdade (or)

```
1 Program Tabela_verdade_OR;
2 var
3   p, q: boolean;
4
5 Begin
6
7   //V OU V
8   p := TRUE; q := TRUE;
9   if (p or q) then
10     writeln('p: ', p, ' _OU_ ', 'q: ', q);
11
12   //V OU F
13   p := TRUE; q := FALSE;
14   if (p or q) then
15     writeln('p: ', p, ' _OU_ ', 'q: ', q);
16
17   //F OU V
18   p := FALSE; q := TRUE;
19   if (p or q) then
20     writeln('p: ', p, ' _OU_ ', 'q: ', q);
21
22   //F OU F
23   p := FALSE; q := FALSE;
24   if (p or q) then
25     writeln('p: ', p, ' _OU_ ', 'q: ', q);
26
27 End.
```

Qual a saída?

p: TRUE \_OU\_ q: TRUE  
p: TRUE \_OU\_ q: FALSE  
p: FALSE \_OU\_ q: TRUE

Quais a linha?

# Operadores Lógicos – Tabelas Verdade (or)

```
1 Program Tabela_verdade_OR;
2 var
3   p, q: boolean;
4
5 Begin
6
7   //V OU V
8   p := TRUE; q := TRUE;
9   if (p or q) then
10     writeln('p: ', p, ' _OU_ ', 'q: ', q);
11
12   //V OU F
13   p := TRUE; q := FALSE;
14   if (p or q) then
15     writeln('p: ', p, ' _OU_ ', 'q: ', q);
16
17   //F OU V
18   p := FALSE; q := TRUE;
19   if (p or q) then
20     writeln('p: ', p, ' _OU_ ', 'q: ', q);
21
22   //F OU F
23   p := FALSE; q := FALSE;
24   if (p or q) then
25     writeln('p: ', p, ' _OU_ ', 'q: ', q);
26
27 End.
```

Qual a saída?

p: TRUE \_OU\_ q: TRUE  
p: TRUE \_OU\_ q: FALSE  
p: FALSE \_OU\_ q: TRUE

Quais a linha?

# Operadores Lógicos – Tabelas Verdade (or)

```
1 Program Tabela_verdade_OU;
2 var
3   p, q: boolean;
4
5 Begin
6
7   //V OU V
8   p := TRUE; q := TRUE;
9   if (p or q) then
10     writeln('p: ',p, ' _OU_ ', 'q: ',q);
11
12   //V OU F
13   p := TRUE; q := FALSE;
14   if (p or q) then
15     writeln('p: ',p, ' _OU_ ', 'q: ',q);
16
17   //F OU V
18   p := FALSE; q := TRUE;
19   if (p or q) then
20     writeln('p: ',p, ' _OU_ ', 'q: ',q);
21
22   //F OU F
23   p := FALSE; q := FALSE;
24   if not(p or q) then
25     writeln('p: ',p, ' _OU_ ', 'q: ',q);
26
27 End.
```

Saídas?

p: TRUE \_OU\_ q: TRUE  
p: TRUE \_OU\_ q: FALSE  
p: FALSE \_OU\_ q: TRUE  
p: FALSE \_OU\_ q: FALSE

Por que?

# Operadores Lógicos – Tabelas Verdade (or)

```
1 Program Tabela_verdade_OU;
2 var
3   p, q: boolean;
4
5 Begin
6
7   //V OU V
8   p := TRUE; q := TRUE;
9   if (p or q) then
10     writeln('p: ',p, ' _OU_ ', 'q: ',q);
11
12   //V OU F
13   p := TRUE; q := FALSE;
14   if (p or q) then
15     writeln('p: ',p, ' _OU_ ', 'q: ',q);
16
17   //F OU V
18   p := FALSE; q := TRUE;
19   if (p or q) then
20     writeln('p: ',p, ' _OU_ ', 'q: ',q);
21
22   //F OU F
23   p := FALSE; q := FALSE;
24   if not(p or q) then
25     writeln('p: ',p, ' _OU_ ', 'q: ',q);
26
27 End.
```

Saídas?

p: TRUE \_OU\_ q: TRUE  
p: TRUE \_OU\_ q: FALSE  
p: FALSE \_OU\_ q: TRUE  
p: FALSE \_OU\_ q: FALSE

Por que?

*not*

<i>p</i>	<i>q</i>	<i>OR</i>	<i>NOT</i>
TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	TRUE	TRUE
FALSE	TRUE	TRUE	TRUE
FALSE	FALSE	FALSE	Not(FALSE)

# Operadores Lógicos – Tabelas Verdade (xor)

```
1 Program Tabela_verdade_OU_Exclusivo;
2 var
3   p, q: boolean;
4
5 Begin
6
7   //V OU EXCLUSIVO V
8   p := TRUE; q := TRUE;
9   if (p xor q) then
10     writeln('p: ', p, ' _XOR_ ', 'q: ', q);
11
12   //V OU EXCLUSIVO F
13   p := TRUE; q := FALSE;
14   if (p xor q) then
15     writeln('p: ', p, ' _XOR_ ', 'q: ', q);
16
17   //F OU EXCLUSIVO V
18   p := FALSE; q := TRUE;
19   if (p xor q) then
20     writeln('p: ', p, ' _XOR_ ', 'q: ', q);
21
22   //F OU EXCLUSIVO F
23   p := FALSE; q := FALSE;
24   if (p xor q) then
25     writeln('p: ', p, ' _XOR_ ', 'q: ', q);
26
27 End.
```

Saídas?

p: TRUE \_XOR\_ q: FALSE  
p: FALSE \_XOR\_ q: TRUE

<i>p</i>	<i>q</i>	<i>XOR</i>
TRUE	TRUE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE



# Referências

- FORBELLONE, André L. **Lógica de Programação**. Prentice Hall Brasil, 3ª edição, 2005.
- VELOSO, Paulo; et al. **Estrutura de dados**. Rio de Janeiro: Campus, 4ª edição, 1996.
- LAGES & GUIMARAES. **Algoritmos e Estrutura de dados**. Ed. LTC, 1994.
- FARRER, H. **Algoritmos estruturados**. Rio de Janeiro: Guanabara Koogan, 3ª edição, 1989.
- LUIZ, Jaime. **Estrutura de dados e seus algoritmos**. Editora LTC.
- GUEDES, S. **Lógica de Programação Algorítmica**. Editora Pearson, 2014.
- MANZANO, José Augusto N. G. **Algoritmos lógica para desenvolvimento de programação de computadores**. Ed. 1, São Paulo, Erica 2016.
- MANZANO, José Augusto N. G. **Algoritmos técnicas de programação**. Ed 2, São Paulo, Erica, 2016.