#### Estruturas de Decisão

**Introdução**

Até o momento foram vistos comandos de entrada (*scanf, gets, getch, getche*), de saída (*printf*) e de atribuição (*var = expressão*), o que permite a elaboração de programas com ***instruções sequenciais***, seguindo a arquitetura geral de processos e algoritmos:

***entradas*** 🡪 ***processamentos*** 🡪 ***saídas***.

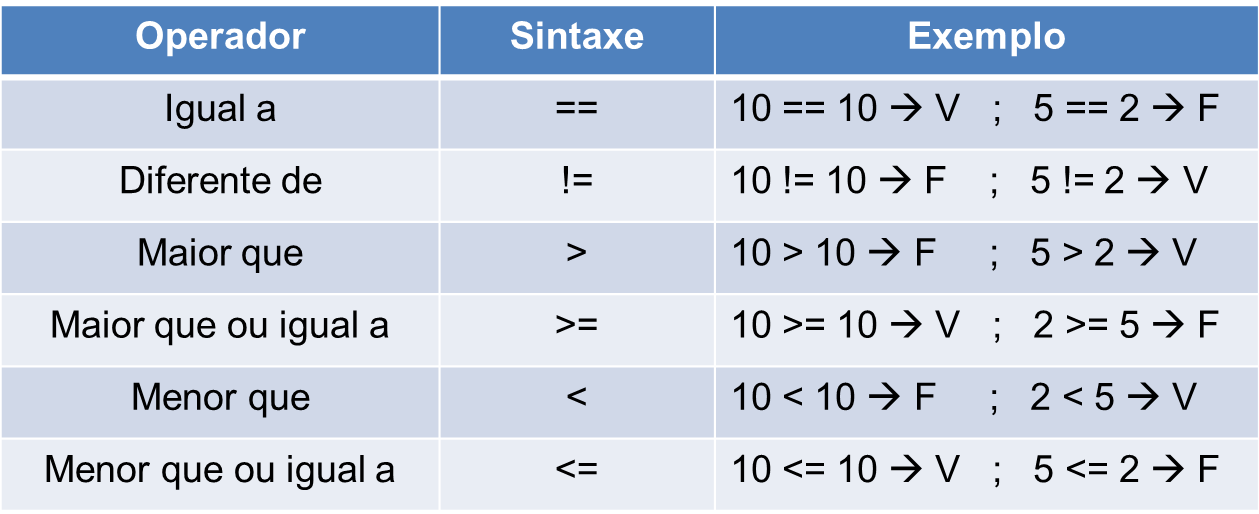
Entretanto, algumas vezes os algoritmos precisam ser mais sutis em seu comportamento. Algumas instruções devem ser executadas somente se determinadas condições foram satisfeitas. Por exemplo, uma mensagem no *inbox* de um *e-mail* deverá ser mostrada em **negrito** se ela ainda não foi lida pelo usuário do e-mail. Ou um aluno será aprovado se sua média for igual ou superior ao valor de corte de aprovação... e ainda se a sua frequência for igual ou maior do que 75% das aulas ministradas.

Estas situações são tratadas por meio das denominadas ***instruções condicionais***, estruturas que permitem que determinadas instruções somente sejam executadas condicionadas ao resultado de ***expressões lógicas***, que checam as condições e modificam o fluxo de execução das instruções.

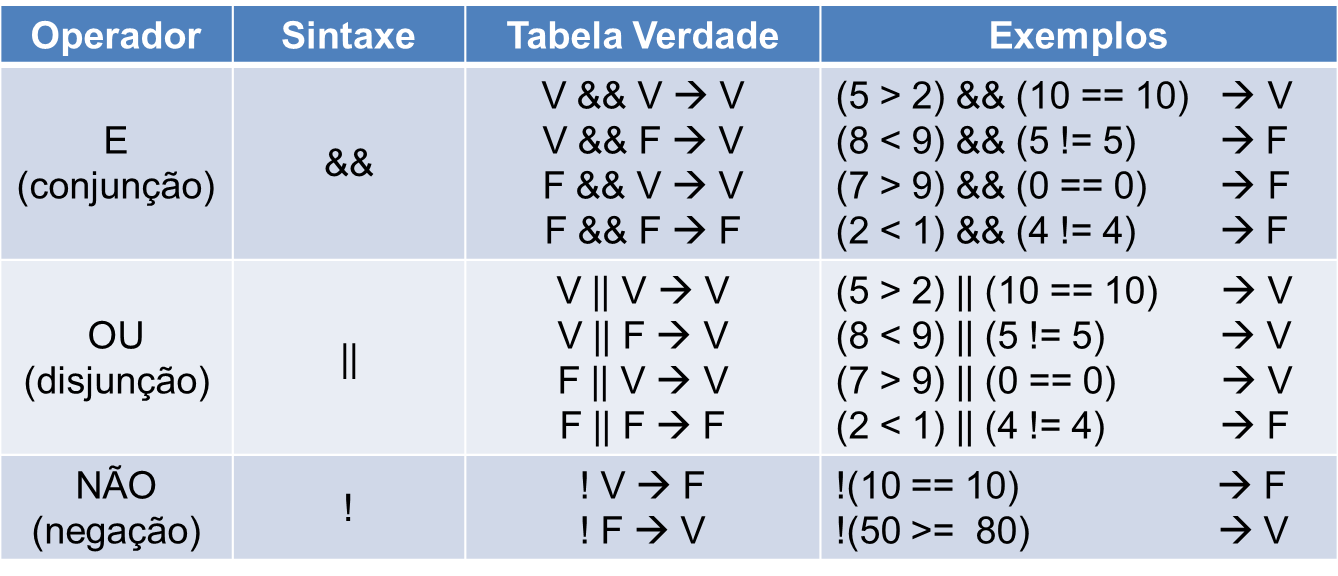
**Expressões Lógicas**

A maioria das linguagens de programação possuem um **tipo primitivo** para valores ***lógicos*** ou ***booleanos***. Este tipo, geralmente denominado ***boolean***, possui apenas dois valores em seu escopo: ***True*** e ***False*** (ou em português, ***verdadeiro*** e ***falso***). Entretanto, na linguagem C não existe um tipo ***boolean*** ou equivalente. Valores lógicos são representados em C por números interiros: **Falso = 0** e **Verdadeiro = 1** (prá ser mais preciso, em C: **Falso = 0** e **Verdadeiro ≠ 0**).

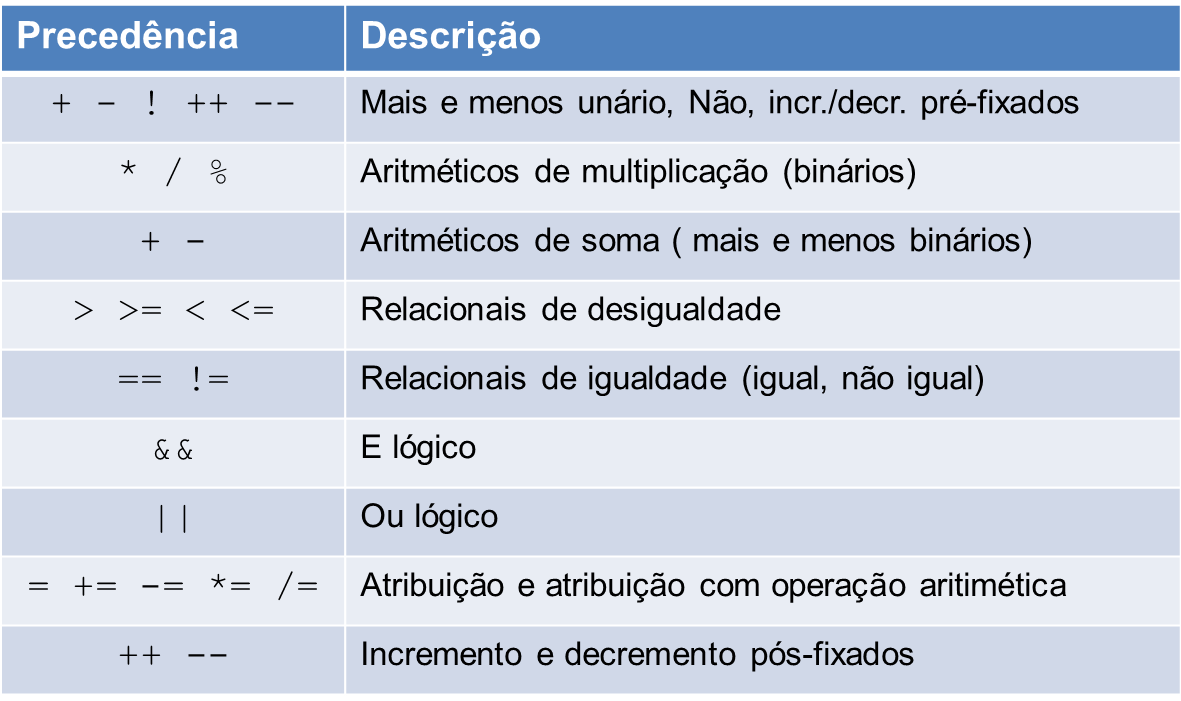
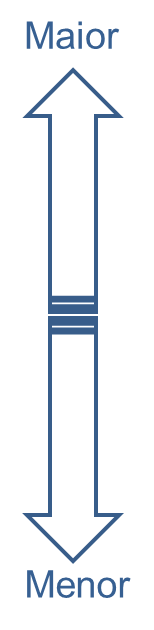
Neste contexto, ***expressões lógicas*** são expressões que estabelecem relações (comparações) entre operandos e sempre retornam um valor ***lógico*** (ou ***booleano***) – ***V*** ou ***F***. Uma ***expressão lógica*** é caracterizada pela presença de um ***operador lógico***, que relaciona (ou compara) duas outras expressões, e produz o resultado lógico da comparação. Por exemplo [10 > 5] produz ***V***. Outro exemplo, [4\*2 **≠** 5+3], produz ***F*** (8 não é diferente de 8!) .O quadro a seguir descreve os 6 ***operadores lógicos*** e respectivas sintaxes em C:



Para atender a situações onde as condições são mais complexas, como no caso citado de um aluno depender de duas condições para aprovação (média ≥ nota de corte e frequência ≥ 75% das aulas ministradas) existem os ***conectivos lógicos***, descritos no quadro abaixo:

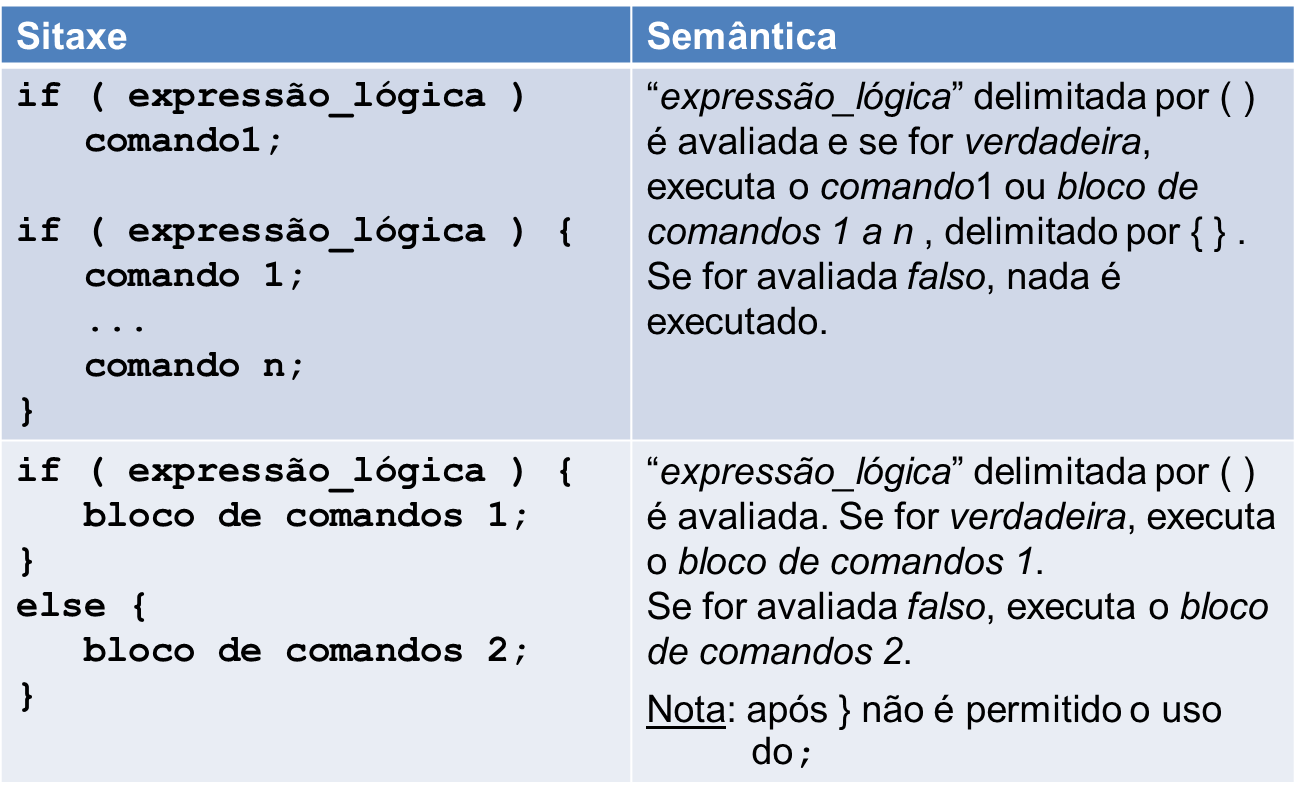


Observando os exemplos apresentados, nota-se que ***expressões lógicas*** podem ter ***expressões aritméticas*** como operandos. Deste modo, faz-se necessário estabelecer uma precedência global entre todos os operadores para determinar a correta execução das operações. O quadro a seguir define essas precedências, lembrando que parênteses permitem priorizar execuções alterando precedências.



**Comando de Decisão - *IF***

Com o intuito de escrever programas efetivamente úteis, é necessária a habilidade de verificar condições e mudar o comportamento do programa adequadamente. Os comandos de decisão, seleção e repetição provêem essa habilidade. A forma mais simples de fazer isso é o ***comando de decisão***, ou simplesmente comando ***if***. Esta estrutura permite executar condicionalmente um comando ou bloco de comandos. O quadro a seguir descreve a sintaxe e semântica do comando ***if*** na linguagem C:



A seguir são listados alguns programas para ilustrar o uso do comando if:

1. Ler a idade de uma pessoa e mostra se ela é maior de idade ou menor.

#include <stdio.h>

void main ( ) {

int idade;

printf ( " Digite a idade: " );

scanf ( “%d”, &idade );

**if( idade >= 18 )**

**printf ( "\n Maior de idade." );**

**else**

**printf ( "\n Menor de idade." );**

getch( );

}

1. Ler dois números inteiros e mostrar o maior deles.

(Note que “escondida” nesta especificação há uma terceira possibilidade: os números lidos serem iguais!)

#include <stdio.h>

void main ( ) {

int num1, num2;

printf ( " Digite dois números inteiros: " );

scanf ( “%d%d”, &num1, &num2 );

**if ( num1 > num2 )**

**printf( "\n %d maior que %d ", num1, num2 );**

**else**

**if ( num1 < num2)**

**printf( "\n %d maior que %d ", num2, num1 );**

**else**

**printf( "\n %d igual a %d ", num1, num2 );**

getch( );

}

1. Ler um número inteiro e informar se ele é par ou impar.

#include <stdio.h>

int main (){

int num;

printf ("Entre com numero: ");

scanf ("%d",&num);

**if(a%2 == 0)**

**printf ("\n O numero e par.\n");**

**else**

**printf ("\n O numero e impar.\n");**

return 0;

}

Nota do programador: um número é par se ele for divisível por 2, isto é, se o resto da divisão por 2 for igual a 0.