Especialização em Back End III

Operadores

Em toda linguagem de programação, precisamos de operadores para resolver problemas de programação. Vamos revisar o que é um operador, seus tipos e funcionalidades.

O que é um operador?

Um operador é um elemento do programa que é aplicado a um ou vários operandos em uma expressão ou instrução. Os operadores, junto com os operandos, conformam uma expressão que é uma fórmula que define o cálculo de um valor.

Quais são os tipos de operadores que existem?

Existem diferentes tipos de operadores. Go possui os seguintes:

- Aritméticos
- Relacionais
- Lógicos
- De atribuição
- De direcionamento

Vamos ver detalhadamente cada um deles.

Operadores aritméticos

Esses operadores nos permitirão realizar operações básicas sobre as variáveis e constantes que declaramos. Na tabela a seguir, veremos alguns desses operadores. A coluna “Exemplo” é calculada presumindo que X:=10 e Y:=5.

Operador	Descrição	Exemplo
+	Soma os operados dos extremos.	X + Y resulta 15
–	Subtrai o operando da direita do operando da esquerda.	X – Y resulta 5
*	Multiplica os operadores dos extremos.	X * Y resulta 50
/	Divisão do número da esquerda (numerador) pelo número da direita (denominador).	X / Y resulta 2
%	Operador modular ou residual de divisão inteira.	X / Y resulta 0
++	Operador de incremento. Incrementa em 1 o operador da esquerda. Não permite o decremento pré-fixado (pré decremento).	X++ resulta 11
--	Operador de decremento. Decrementa em 1 o operador da esquerda, ou seja, não permite o decremento pré-fixado.	X-- resulta 9

Operadores relacionais

Esses operadores são aqueles que nos retornam um valor de verdade. Na tabela a seguir, veremos alguns exemplos. A coluna “Exemplo” está calculada presumindo que X:=1 e Y:=2.

Operador	Descrição	Exemplo
==	Retorna verdadeiro quando ambos os operandos forem iguais. Devolve falso em qualquer outro caso.	X == Y retorna falso
!=	Retorna verdadeiro somente quando os operadores forem diferentes.	X != Y retorna verdadeiro
>	Retorna verdadeiro quando o operando da esquerda for maior ao da direita.	X > Y retorna falso
<	Retorna verdadeiro quando o operando da esquerda for menor ao da direita.	X < Y retorna verdadeiro
>=	Retorna verdadeiro quando o operador da esquerda for maior ou igual ao da direita.	X >= Y retorna falso
<=	Retorna verdadeiro quando o operador da esquerda for menor ou igual ao da direita.	X <= Y retorna verdadeiro

Operadores lógicos

Operador	Descrição	Exemplo
&&	Operador lógico de conjunção (*AND*). Compara dois valores bool ou expressões racionais.	A && B resulta falso X > 0 && Y < 6 resulta verdadeiro
||	Operador lógico de disjunção (*OR*). Compara dois valores bool ou expressões relacionais.	A || B resulta verdadeiro X < 0 || Y > 6 resulta falso
!	Operador de negação. Inverte (nega) o valor *bool* do operando.	!A resulta falso !B resulta verdadeiro

Operadores de atribuição

Esses operadores nos permitem alterar o valor das nossas variáveis durante a execução do programa.

Operador	Descrição	Exemplo
=	Operador de atribuição simples.	X = Y + Z atribui a X a soma entre Y e Z (Y e Z não são alterados).
+=	Operador de soma e atribuição.	X += Y atribui a X o valor de X + Y
-=	Operador de subtração e atribuição.	X -= Y atribui a X o valor de X – Y
*=	Operador de multiplicação e atribuição.	X *= Y atribui a X o valor de X * Y
/=	Operador de divisão e atribuição.	X /= Y atribui a X o valor de X /Y
%=	Operador de módulo e atribuição.	X %= Y atribui a X o valor de X % Y

Operadores de endereço

Operador	Descrição	Exemplo
&	Retorna o endereço em memória do operando.	&X retorna o endereço em memória de X
*	Apontador para uma variável.	*P aponta para uma variável

Precedência de operadores

A precedência de um operador indica quão “estreitamente” duas expressão juntas estão unidas. Por exemplo, a expressão **1 + 5 * 3**, a resposta é 16 e não 18, porque o operador de multiplicação (“*”) tem uma precedência maior do que o operador de ação (“+”). Se for necessário, os parênteses podem ser usados para forçar a precedência. Por exemplo: **(1 + 5) * 3**, é avaliado como **18**.

O que aprendemos?

Para concluirmos, podemos destacar a importância do uso de operadores na programação. Aprendemos o que é um operador e os seus diferentes tipos em Go.