Operátory a výrazy

Od každého programu se očekává, že bude vykonávat nějakou činnost, ať už je to přesun dat z jedné proměnné do druhé nebo sečtení dvou čísel. Programátor sděluje počítači, jakou základní operaci chce provést, pomocí operátorů. Operátor je vyhrazený znak, který má nějakou specifickou funkci. Všichni například znají operátor +, který sečte dva operandy (operand je symbol, na který je použit operátor) i všechny ostatní základní aritmetické operátory. Zdaleka nejpoužívanějším operátorem při programování je operátor (příkaz) přiřazení. Tento operátor je definován dvěma symboly := a jeho použití bylo již v této kapitole několikrát demonstrováno. Např. A := B; je jednoduchý kód, který se zaslouží o to, že se obsah proměnné B zkopíruje do proměnné A. Obě proměnné mají po vykonání příkazu stejnou hodnotu a původní obsah proměnné A je nenávratně ztracen. Kromě přesunu dat mezi proměnnými potřebujeme provádět v programech ještě řadu dalších operací. K tomu účelu je k dispozici celá řada dalších operátorů, které je možné podle funkce rozdělit do několika skupin.

Aritmetické operátory

Aritmetické operátory provádějí základní matematické operace. Tyto operátory se dále dělí na unární (používají pouze jeden operand) a binární (spojuji dva nebo více operandů). Následující tabulka uvádí jejich přehledný seznam.

Operátor	Operace	Typ Operandu	Typ výsledku	Příklad použití
+	součet	integer, real	integer, real	X+Y
-	rozdíl	integer, real	integer, real	vysledek -1
*	násobení	integer, real	integer, real	P * Polomer
1	dělení	ínteger, real	real	X/2
div	celočíselné dělení	integer	integer	Celek div počet
mod	zbytek po celočíselném dělení	integer	integer	Y mod 6
+ (unární)	kladné znaménko	integer, real	integer, real	+7
- (unární)	záporné znaménko	ínteger, real	integer, real	-X

Funkce všech operátorů je zřejmá, kromě operátorů div a mod, které budeme demonstrovat na příkladu.

```
5 div 2 = 2    // pět děleno dvěma je 2
5 mod 2 = 1 //pět děleno dvěma je 2 a zbytek po celočíselném dělení je jedna
```

Logické operátory

Logické operátory provádějí logické operace s operandy typu Boolean. Výsledky logických operaci jsou opět výhradně typu Boolean. V Delphi jsou k dispozici čtyři logické operátory: and (logický součin), or (logický součet), xor (exkluzivní součet) a not (negace). Následující tabulka ukazuje, jaké výsledky dávají jednotlivé operátory při různých hodnotách operandů.

A	B	A and B	A or B	A xor B	not A
TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE	FALSE
FALSE	TRUE	FALSE	TRUE	TRUE	TRUE
FALSE	FALSE	FALSE	FALSE	FALSE	TRUE

Relační operátory

K porovnáváni velikosti jednotlivých operandů slouží relační operátory. Výsledek takového porovnáni je typu Boolean a má hodnotu podle toho, zda je výraz pravdivý či nikoli. Nepřiklad v případě 2 < 3 má výsledek hodnotu TRUE, narozdíl od 5 < 4 kdy má hodnotu FALSE. Jinak použití ani význam relačních operátorů v sobě neskrývá nic nejasného, proto k seznámení s nimi postačí tabulka s jejich přehledem.

Operátor	Význam	Typ výsledku	Příklad použití
=	rovná se	Boolean	1= Max
<>	nerovná se	Boolean	X <> Y
<	menši než	Boolean	X<Y
>	větší než	Boolean	Len> O
<=	menší nebo rovno	Boolean	X <= 2
>=	věšti nebo rovno	Boolean	1>= 1

Výrazy

Spojení operátoru a operandů vytváří výraz, který má nějakou hodnotu, jak bylo uvedeno. Výraz je tedy posloupnost proměnných, konstant a operátorů. Každý výraz má nějakou hodnotu, kterou lze dále zpracovávat, ale nelze ji měnit. To znamená, že výraz může stát na pravé straně přiřazení, ale nikdy ne na levé. Proto např. i:=1; i := i+1; je korektní příkaz, neboť do proměnné i se přiřazuje výsledek výrazu i + 1 (hodnota i se zvětší o jedničku), ale i+1 := 25; je chyba, protože výrazu se přiřazuje hodnota, a to není možné. Zajímavou vlastností výrazů je to, že mohou uvnitř obsahovat další výrazy, např. X+Y*Z. Zde se ovšem vyskytuje problém, v jakém pořadí se takové složené výrazy vyhodnocují. Jinými slovy, jakou má který operátor prioritu. Celý problém není nijak složitý, protože platí základní matematická pravidla. Výrazy se vyhodnocují zleva doprava, první se vyhodnocují výrazy v závorkách, pokud závorky chybí, má přednost násobení před sčítáním a odčítáním atd. Však si na to pamatujete ze školy. Ke správnému určení výsledku výrazu je však zapotřebí znát prioritu všech operátorů, a ne jen těch aritmetických. Proto následující tabulka ukazuje kompletní seznam všech operátorů Delphi (včetně těch, o kterých dosud nepadla ani zmínka) v pořadí, v jakém se budou vyhodnocovat ve výrazu, pokud v

něm nebudou závorky.

Operátory	**Priorita**
@, not	první (nejvyšší)
*, I, div, mod, and, shl, shr, as	druhá
+, -, or, xor	třetí
=, <>, <, >, <=, >=, in, is	čtvrtá (nejnižší)

Zde platí zlaté pravidlo: pokud máš pochybnosti o prioritách, závorkuj. Nejenom, že se předejde chybám, ale výrazy obsahující závorky jsou mnohem čitelnější. Logické výrazy obsahují logické operátory a jejích výsledek je typu Boolean. Mají neocenitelný význam při řízení činnosti programu a podobných akcích. Operandy v logických výrazech mohou být libovolného typu. Pokud ale v logickém výrazu porovnáváme rovnost operandů např. A = B nelze použít typ Real, protože vlivem operací v pohyblivé řádové čárce dochází k jistým nepřesnostem, které způsobují chybné vyhodnocení výrazu. Tyto chyby jsou velice záludné a obtížně se hledají. Výraz R = O, kde R je proměnná typu Real má vždy výsledek FALSE, protože vlivem zaokrouhlovacích chyb se na nějakém vzdáleném desetinném místě objeví nenulová hodnota, která zvrátí celý výsledek. Pro počítač je totiž číslo 0.000000000001 nenulové. Pokud byste čekali, až bude výraz pravdivý (tedy až R bude mít na všech desetinných místech nulu), nemuseli byste se vůbec dočkat.