navigation

- Main Page
- Documentation
- FAQ
- Downloads
- Glossary
- Index
- Recent changes
- Random page
- Help

search

Search

Go Search

tools

- What links here
- Related changes
- Special pages
- Printable version
- Permanent link
- Page information

Assignment and Operations

| Deutsch (de) | English (en) | español (es) | français (fr) | 日本語 (ja) | 한국어 (ko) | русский (ru) | 中文（中国大陆）(zh_CN) |

◄ ▲ ►

1E - Assignment and Operations (author: Tao Yue, state: unchanged)

Once you have declared a variable, you can store values in it. This is called assignment.

To assign a value to a variable, follow this syntax:

```
variable_name := expression;
```

Note that unlike other languages, whose assignment operator is just an equals sign, Pascal uses a colon followed by an equals sign, similarly to how it's done in most computer algebra systems.

The expression can either be a single value:

```
some_real := 385.385837;
```

or it can be an arithmetic sequence:

```
some_real := 37573.5 * 37593 + 385.8 / 367.1;
```

The arithmetic operators in Pascal are:

Operator	Operation	Operands	Result
+	Addition or unary positive	real or integer	real or integer
-	Subtraction or unary negative	real or integer	real or integer
*	Multiplication	real or integer	real or integer
/	Real division	real or integer	real
div	Integer division	integer	integer
mod	Modulus (remainder division)	integer	integer

`div` and **`mod`** only work on *integers*. **`/`** works on both *reals* and *integers* but will always yield a *real* answer. The other operations work on both *reals* and *integers*. When mixing *integers* and *reals*, the result will always be a *real* since data loss would result otherwise. This is why Pascal uses two different operations for division and integer division. `7 / 2 = 3.5` (real), but `7 div 2 = 3` (and `7 mod 2 = 1` since that's the remainder).

Each variable can only be assigned a value that is of the same data type. Thus, you cannot assign a real value to an integer variable. However, certain data types will convert to a higher data type. This is most often done when assigning integer values to real variables. Suppose you had this variable declaration section:

```
var
  some_int : integer;
  some_real : real;
```

When the following block of statements executes,

```
some_int := 375;
some_real := some_int;
```

`some_real` will have a value of `375.0`.

Changing one data type to another is referred to as typecasting. Modern Pascal compilers support explicit typecasting in the manner of C, with a slightly different syntax. However, typecasting is usually used in low-level situations and in connection with object-oriented programming, and a beginning programming student will not need to use it. Here is information on typecasting from the GNU Pascal manual.

In Pascal, the minus sign can be used to make a value negative. The plus sign can also be used to make a value positive, but is typically left out since values default to positive.

Do not attempt to use two operators side by side, like in:

```
some_real := 37.5 * -2;
```

This may make perfect sense to you, since you're trying to multiply by negative-2. However, Pascal will be confused — it won't know whether to multiply or subtract. You can avoid this by using parentheses to clarify:

```
some_real := 37.5 * (-2);
```

The computer follows an order of operations similar to the one that you follow when you do arithmetic. Multiplication and division (`* / div mod`) come before addition and subtraction (`+ -`), and parentheses always take precedence. So, for example, the value of: `3.5*(2+3)` will be `17.5`.

Pascal cannot perform standard arithmetic operations on Booleans. There is a special set of Boolean operations. Also, you should not perform arithmetic operations on characters.

◄ ▲ ►

Category: Object Pascal Introduction