Variables and Data Types

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

1D - Variables and Data Types (author: Tao Yue, state: *changed*)

Variables are similar to constants, but their values can be changed as the program runs. Variables must first be declared in Pascal before they can be used:

```
var
  IdentifierList1 : DataType1;
  IdentifierList2 : DataType2;
  IdentifierList3 : DataType3;
  ...
```

`IdentifierList` is a series of identifiers, separated by commas (`,`). All identifiers in the list are declared as being of the same data type.

The basic data types in Pascal include:

- integer
- real
- char
- boolean

Standard Pascal does not make provision for the string data type, but most modern compilers do. Experienced Pascal programmers also use pointers for dynamic memory allocation, objects for object-oriented programming, and many others, but this gets you started.

More information on Pascal data types:

- The **integer** data type can contain integers from `-32768` to `32767`. This is the signed range that can be stored in a 16-bit word, and is a legacy of the era when 16-bit CPUs were common. For backward compatibility purposes, a 32-bit signed integer is a longint and can hold a much greater range of values.
- The **real** data type has a range from $3.4x10^{-38}$ to $3.4x10^{38}$, in addition to the same range on the negative side. Real values are stored inside the computer similarly to scientific notation, with a mantissa and exponent, with some complications. In Pascal, you can express real values in your code in either fixed-point notation or in scientific notation, with the character `E` separating the mantissa from the exponent. Thus, `452.13` is the same as `4.5213e2`
- The **char** data type holds characters. Be sure to enclose them in single quotes, like so: `'a' 'B' '+'` Standard Pascal uses 8-bit characters, not 16-bits, so Unicode, which is used to represent all the world's language sets in one UNIfied CODE system, is not supported.
- The **WideChar** is a two-byte character (an element of a DBCS: Double Byte Character Set) and can hold a Unicode character. Note: some Unicode characters require two WideChars. See *UTF-16*.
- Free Pascal supports the Delphi implementation of the **PChar** type. PChar is defined as a pointer to a Char type, but allows additional operations. The PChar type can be understood best as the Pascal equivalent of a C-style null-terminated string, i.e. a variable of type PChar is a pointer that points to an array of type Char, which is ended by a null-character (#0). Free Pascal supports initializing of PChar typed constants, or a direct assignment. For example, the following pieces of code are equivalent:

```
program one;
var P : PChar;
begin
  P := 'This is a null-terminated string.';
  WriteLn (P);
end.
```

```
program two;
const P : PChar = 'This is a null-terminated string.';
begin
  WriteLn (P);
end.
```

- Free Pascal supports the **String** type as it is defined in Turbo Pascal: a sequence of characters with an optional size specification. It also supports ansistrings (with unlimited length) as in Delphi. And can be declared as:

```
variable_name : string;                  // if no length is given, it defaults to 255
variable_name : string[length];          // where:  1 < length <= 255
```

- The predefined type **ShortString** is defined as a string of size 255.
- **AnsiStrings** are strings that have no length limit. They are reference counted and are guaranteed to be null terminated. Internally, an ansistring is treated as a pointer: the actual content of the string is stored on the heap, as much memory as needed to store the string content is allocated.
- **Widestrings** (used to represent unicode character strings) are implemented in much the same way as ansistrings: reference counted, null-terminated arrays, only they are implemented as arrays of `WideChars` instead of regular `Chars`.
- The **boolean** data type can have only two values: **TRUE** and **FALSE**

An example of declaring several variables is:

```
var
  age, year, grade : integer;
  circumference : real;
  LetterGrade : char;
  DidYouFail : Boolean;
```

From the FPC manual

integer types

Type	Range	Bytes
Byte	0 .. 255	1
Shortint	-128 .. 127	1
Smallint	-32768 .. 32767	2
Word	0 .. 65535	2
Integer	smallint or longint	2 or 4
Cardinal	longword	4
Longint	-2147483648 .. 2147483647	4
Longword	0..4294967295	4
Int64	-9223372036854775808 .. 9223372036854775807	8
QWord	0 .. 18446744073709551615	8

Free Pascal does automatic type conversion in expressions where different kinds of integer types are used.

real types

Type	Range	Significant digits	Bytes
Real	platform dependent	???	4 or 8
Single	1.5E-45 .. 3.4E38	7-8	4
Double	5.0E-324 .. 1.7E308	15-16	8
Extended	1.9E-4932 .. 1.1E4932	19-20	10
Comp	-2E64+1 .. 2E63-1	19-20	8
Currency	-922337203685477.5808	922337203685477.5807	8

boolean types

Type	Bytes	Ord(True)
Boolean	1	1
ByteBool	1	Any nonzero value
WordBool	2	Any nonzero value
LongBool	4	Any nonzero value