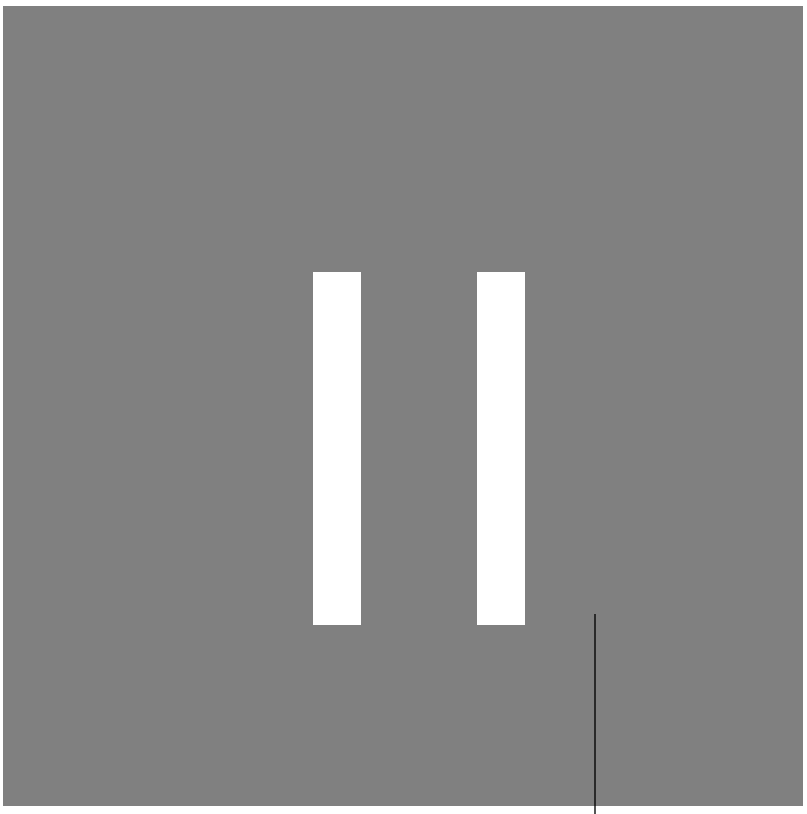

E X T E N S I O N

Introducing C to Pascal Programmers

C is not a "very high level" language . . . and is not specialized to any particular area of application. But its absence of restrictions and its generality make it more convenient and effective for many tasks than supposedly more powerful languages.

Brian W. Kernighan and Dennis M. Ritchie
The C Programming Language, Preface, 1978

A primary motivation for . . . Pascal was the need for a powerful and flexible language that could be reasonably efficiently implemented on most computers.

Kathleen Jensen and Niklaus Wirth
Pascal User Manual and Report, p. 165, 1975

II.1 Introduction

This Web extension is meant for readers familiar with Pascal but not C. It is not intended as a tutorial on how to write C programs, but as a way of allowing the reader familiar with Pascal to better understand the small amount of C code that appears in this book. Given the level of examples in this book, these differences are primarily syntactic.

II.2 Variable Declarations

There is no standard Pascal type for unsigned integers or double precision floating point. Also, Pascal ignores capitalization in variable names while

capitalization counts in C. For example, `Apple` and `apple` are different variables in C. Here are the corresponding standard types:

Type	C declaration	Pascal declaration
Integer	`int`	`integer`
Single precision floating point	`float`	`real`
Unsigned integer	`unsigned int`	?
Double precision floating point	`double`	?

II.3 Assignment Statements

The primary difference is that C uses the "=" while Pascal uses ":=" to indicate an assignment. Here are C examples from the book with their equivalents in Pascal.

Examples in C	Corresponding Pascal code
`a = b + c;`	`a := b + c;`
`d = a - e;`	`d := a - e;`
`f = (g + h) - (i + j);`	`f := (g + h) - (i + j);`
`g = h + P[i];`	`g := h + P[i];`
`P[i] = h + P[i];`	`P[i] := h + P[i];`

In addition to standard arithmetic operators (+,-,*,/), C has some logical operators sometimes found as library routines in other languages.

Logical operations	C operator
Shift Left	<<
Shift Right	>>
AND	&
OR	|
XOR	^
NOT	~

Operators << and >> are logical operations only on unsigned integers in C.

II.4 Relational Expressions and Conditional Statements

There is more difference in the *if* statements of the two programming languages, both in the statements themselves and in the expressions that commonly occur. C leaves off the keyword "`then`" in the traditional *if* statement, and since "=" is used to mean assignment, new symbols are used to mean relational equality. The table below shows the mapping of the relational operators in both languages:

Operation	C	Pascal
Equal	==	=
Not equal	!=	<>
Less than	<	<
Less than or equal	<=	<=
Greater than	>	>
Greater than or equal	>=	>=

Here are two examples of *if* statements.

C	Pascal
`if (i == j) f = g + h;` `  else f = g - h;`	`if i = j then f := g + h` `  else f := g - h;`
`if (i == j) goto L1;`	`if i = j then goto 1;`
`f = g + h;`	`f := g + h;`
`L1:f = f - i;`	`1: f := f - i;`

C replaces the `begin end` of Pascal's compound statements with `{ }`.

The *case* statement in Pascal is quite similar to the *switch* statement in C. Each switching alternative in C starts with the keyword "`case`" and ends with the keyword "`break`." Here are equivalent statements:

C	Pascal

`switch (k) {`
`  case 0: f = i + j; break;`
`  case 1: f = g + h; break;`
`  case 2: f = g - h; break;`
`  case 3: f = i - j; break;`
`};`
	`case k of`
	`  0: f := i + j;`
	`  1: f := g + h;`
	`  2: f := g - h;`
	`  3: f := i - j;`
	`end;`

II.5 Loops

The *while* loops are almost identical in the two languages. Here is an example:

C	Pascal
`while (save[i] == k)` `  i = i + j;`	`while save[i] = k do` `  i := i + j;`

Of course, *while* loops can be constructed from `gotos`:

C	Pascal

`Loop:  g = g + P[i];`
`  i = i + j;`
`  if (i != h) goto Loop;`
	`2:  g := g + P[i];`
	`  i := i + j;`
	`  if i <> h then goto 2;`

The *for* statement may be the most unusual. In keeping with the philosophy of no restrictions, the initialization, exit test, and per loop operation can be any statements. They appear in three pieces in that order in the *for* statement:

C	Pascal
`for (i = 0; i < n; i = i + 1)` `  {...}`	`for i := 0 to n - 1 do` `  begin ... end`

In addition to the relational operators, there are logical relational operators to connect conditions. Here they are in the two languages:

Operation	C	Pascal
And	&&	and
Or	||	or
Not	!	not

This example shows some of the power of the *for* statement; the compound exit test requires nested statements in Pascal.

C	Pascal

`for (j = i - 1; j >= 0`
`  && v[j] > v[j + 1]; j = j - 1)`
`  { ... }`
	`  for j := i - 1 downto 0 do`
	`  if v[j] > v[j + 1] then`
	`  begin ... end`
	`  else goto 3;`
	`3:  ...`

II.6 Examples to Put it All Together

Procedures in C and Pascal are quite similar. The primary difference is that arrays are normally declared as types, with the type name used to declare array variables:

C	Pascal

`swap (int v[], int k)`
 	
`{`
`  int temp;`
 	
`  temp  = v[k];`
`  v[k]  = v[k + 1];`
`  v[k + 1] = temp;`
`};`
	`type`
	`names = array [0..19] of integer;`
	`...`
	`procedure swap (var v: names;`
	`  k: integer);`
	` var`
	`  temp: integer;`
	`begin`
	`  temp  := v[k];`
	`  v[k]  := v[k + 1];`
	`  v[k + 1] := temp`
	`end;`

Here is a longer example::

C	Pascal

`sort (int v[], int n)`
 	
`{`
`int i, j;`
 	
`for (i = 0; i < n; i = i + 1)`
`for (j = i-1; j >= 0 &&`
`v[j] > v[j + 1]; j = j - 1)`
`swap(v,j);`
 	
`};`
	`procedure sort (var v: names;`
	`n: integer);`
	`var`
	`i, j: integer;`
	`begin`
	`for i := 0 to n - 1 do`
	`for j := i – 1 downto 0 do`
	`if v[j] > v[j + 1] then`
	`swap(v,j)`
	`else goto 4;`
	`4: ;`
	`end;`

To Probe Further

Kernighan, Brian W., and Dennis M. Ritchie [1988]. *The C Programming Language*, 2nd edition, Prentice Hall, Englewood Cliffs, N.J.

This classic text is so widely used it's known as "K&R"; be sure to get the second edition. The first section, which is a tutorial, is a good introduction to C for someone who knows how to program.

Tondo, Clovis L., and Scott E. Gimpel [1989]. *The C Answer Book: Solutions to the Exercises in the C Programming Language*, 2nd edition, Prentice Hall, Englewood Cliffs, N.J.

The second edition of this book has the answers to the exercises in the second edition of K&R.

II.7 Exercises

II.1 [5] Write a Pascal version of the C program for summing shown in Appendix A, Figure A.5 on page A-8.

II.2 [5] Write a Pascal version of the first C procedure to set an array to zero, Clear1, shown in Chapter 3, Figure 3.27 on page 171.

II.3 [10] Write a Pascal version of the second C procedure to set an array to zero, Clear2, shown in Chapter 3, Figure 3.27 on page 171.