Writing a C program

Do not use word processor applications to create program code as they store additional formatting information that prevents code compilation.

In C programs the code statements to be executed are contained within "functions", which are defined using this syntax format:

data-type function-name **( )** **{** ***statements-to-be-executed*** **}**

After a function has been called upon to execute the statements it contains, it can return a value to the caller. This value must be of the data type specified before the function name.

A program can contain one or many functions but must always have a function named "main". The **main()** function is the starting point of all C programs, and the C compiler will not compile the code unless it finds a **main()** function within the program.

Other functions in a program may be given any name you like using letters, digits, and the underscore character, but the name may not begin with a digit. Also, the C keywords, listed in the table on the front inner cover of this book, must be avoided.

Preprocessor instructions begin with a **#** hash character and must enclose standard library names within **< >** angled brackets.

The **( )** parentheses that follow the function name may, optionally, contain values to be used by that function. These take the form of a comma-separated list and are known as function "arguments" or "parameters".

The **{ }** curly brackets (braces) contain the statements to be executed whenever that function is called. Each statement must be terminated by a semi-colon, in the same way that English language sentences must be terminated by a period/full stop.

Traditionally, the first program to attempt when learning any programming language is that which simply generates the message "Hello World".

1. Open a plain text editor, such as Notepad, then type this line of code at the start of the page, exactly as it is listed

```
#include <stdio.h>
```

The program begins with an instruction to the C compiler to include information from the standard input/output **stdio.h** library file. This makes the functions contained within that library available for use within this program. The instruction is more properly called a "preprocessor instruction" or "preprocessor directive" and must always appear at the start of the page, before the actual program code is processed.

2 Two lines below the preprocessor instruction, add an empty main function

```
int main()
{

}
```

This function declaration specifies that an integer value, of the **int** data type, should be returned by the function upon completion.

Whitespace between the code is ignored by the C compiler but program code should always end with a newline character.

3 Between the braces, insert a line of code that calls upon one of the functions defined in the standard input/output library – made available by the preprocessor instruction

```
printf ( "Hello World!\n" ) ;
```

Here the **printf()** function specifies a single string argument between its parentheses. In C programming, strings must always be enclosed within double quotes. This string contains the text **Hello World** and the **\n** "newline" escape sequence that moves the print head to the left margin of the next line.

4 Between the braces, insert a final line of code to return a zero integer value, as required by the function declaration

```
return 0 ;
```

Traditionally, returning a value of zero after the execution of a program indicates to the operating system that the program executed correctly.

Each statement must be terminated by a semi-colon character.

5 Check that the program code looks exactly like the listing below, then add a final newline character (hit Return after the closing brace) and save the program as "hello.c"

```
#include <stdio.h>

int main()
{
  printf( "Hello World!\n" ) ;
  return 0 ;
}
```

The complete program in text format is now ready to be compiled into machine-readable byte format as an executable file.