5 Selection Statements

`_Bool` *type*

C99 provides a Boolean type named `_Bool`; C89 has no Boolean type.

6 Loops

`for` *statements*

In C99, the first expression in a `for` statement can be replaced by a declaration, allowing the statement to declare its own control variable(s).

7 Basic Types

`long long` *integer types*

C99 provides two additional standard integer types, `long long int` and `unsigned long long int`.

extended integer types

In addition to the standard integer types, C99 allows implementation-defined extended signed and unsigned integer types.

`long long` *integer constants*

C99 provides a way to indicate that an integer constant has type `long long int` or `unsigned long long int`.

types of integer constants

C99's rules for determining the type of an integer constant are different from those in C89.

hexadecimal floating constants

C99 provides a way to write floating constants in hexadecimal.

implicit conversions

The rules for implicit conversions in C99 are somewhat different from the rules in C89, primarily because of C99's additional basic types.

8 Arrays

designated initializers

C99 supports designated initializers, which can be used to initialize arrays, structures, and unions.

variable-length arrays

In C99, the length of an array may be specified by an expression that's not constant, provided that the array doesn't have static storage duration and its declaration doesn't contain an initializer.

9 Functions

no default return type

If the return type of a function is omitted in C89, the function is presumed to return a value of type `int`. In C99, it's illegal to omit the return type of a function.

mixed declarations and statements

In C89, declarations must precede statements within a block (including the body of a function). In C99, declarations and statements can be mixed, as long as each variable is declared prior to the first statement that uses the variable.