C Language Reference

|  |  |  |
| --- | --- | --- |
| **Comments** | Single line comments:  Multi-line comments: | |
| **Organisation of a C program** | Header files:      C files: | |
| **Preprocessor include statement** | Description:  User-defined headers (search path includes the current project)  System headers (search path includes operating system or vendor directories) | |
| **Data types** | Boolean (true or false)  Floating point (single and double precision)  Integers (signed and unsigned)  Characters  Strings | |
| **Declaring variables** | Variables must be declared before they can be used. The declaration tells the compiler the data type of the variable. | |
| **Variable assignment** | Notice that all statements end in a semicolon.  Basic assignment operator:  Increment / decrement operators:  Assignment with arithmetic: | |
| **Numbers** | Decimal numbers:  Hexadecimal numbers:  Octal numbers:  Binary numbers: (not standard C but supported by our compiler) | |
| **Logical operators** | Equal to:  Not equal to:  Less than: | Less than or equal to:  Greater than:  Greater than or equal to: |
| **Logical operators** | AND  OR  NOT | |
| **Arithmetic operators** | Add, subtract, multiply, divide | |
| **Bitwise operators** | AND  OR  NOT | XOR  Shift left  Shift right |
| **If statement** | The condition is always in parentheses.  If statement:  If-else structure:  If there’s only a single statement the braces can be omitted:  Omitting the braces means only the first statement after the if is included:  A value of zero is considered false. Any non-zero value is considered true. | |
| **While loop (precondition)** | The condition is tested before the loop runs. | |
| **Do-while loop (postcondition)** | The condition is tested after the loop has run once. | |
| **For loop** | Syntax:  Example:  Sequence of evaluation: | |
| **Break and continue** | Break statement:  Continue statement: | |
| **Function definitions**  Implements a particular function. | Syntax:  Example 1 (no arguments, no return value)  Example 2 (one argument, no return value)  Example 3 (no arguments, integer return value)  Example 4 (two arguments, double precision floating point return value) | |
| **Function prototypes / function declarations** | Declares that a function with this name and these arguments exists. The compiler needs the function prototype for compile-time checking of the data types. Usually placed in header files with the “.h” file extension.  Notice that the prototype is always followed by a semicolon whereas the definition is not.  Syntax:  Examples: | |
| **Pointers** | Declaring a pointer:  Taking the address of a variable:  Setting the address that a pointer points to:  Dereferencing a pointer, or accessing the memory that a pointer points to:  The NULL pointer: | |
| **Arrays** | Declaring arrays:  Indexing arrays:  Assigning arrays | |
| **Link between arrays and pointers** | Array variables are pointers. | |
| **Strings** |  | |
| **Structures** | Structures are variables with named sub-fields.  Defining a structure:  Declaring a structure variable:  Accessing structure fields  Pointers to structures   * Pointers to structures often arise for efficiency. It is faster to pass a pointer instead of copying all the fields inside a structure. | |
| **Preprocessor macro definitions** | Define a macro without giving its expansion (useful for #ifdef conditions, see below)  Define a macro with replacement text. Causes the pre-processor to replace every instance of the macro with the replacement text.  Undo a macro definition: | |
| **Preprocessor conditions** | If a macro is defined:  If a macro is not defined:  Else block: (optional) | |
| **Header file guard** | Ensures that a given header file is included only once so that the compiler does not see duplicated definitions.  Method 1: pre-processor definitions  Method 2: #pragma definitions on modern compilers | |
| **Const qualifier** | Const indicates a constant value. | |
| **Static qualifier** | Outside of functions  Inside of functions | |
| **Volatile** |  | |
| **Extern** |  | |
| **Typecasting** |  | |