32 C keywords

Basic data types

* Primary (primitive): Char, int , float, double
  + Arithmetic operations are applied to primary types
* Others: short, long, signed, unsigned, void

Define data types

* Typedef, struct, union, enum

Modifiers

* Const, auto, static, extern, volatile, register

Flow control

* If, else, switch, case, default, goto, for, while, do, break, continue

Function

* Return, sizeof

**Syntax**

* // comment
* /\* … \*/ block comment
* # preprocessor
* ; statement terminator
* , list separator
* () function parameter
* {} scope of a program block
* && and
* || or
* ! not

**Declaring functions**

Type function\_name(argument type list){

//function block

}

You can declare a function initially just by writing:

Type function\_name(argument);

* If it doesn’t have a type in front it’s a call not a declaration

**The char type**

* Char type is used to present characters as an integer by ASCII code. Ex A is coded as 65, a as 97

**Memory**

* Each addressable memory cells holds 8 bits (1 byte) when a data type has size bigger than 1, it needs a contiguous memory cells (called memory block) to store the value of the type
* The char type uses 1 byte, it uses one addressable memory cell
* Example:
* Character A is coded as 65. The binary representation of 65 is 100 0001. A is stored in memory as 0100 0001
* Big-endian: store the most significant byte in the lowest address cell
* Little-endian: store the least significant byte in the lowest address cell
* Ex. For the data 90AB12CD
* Big endian

|  |  |
| --- | --- |
| Address | Value |
| 1003 | CD |
| 1002 | 12 |
| 1001 | AB |
| 1000 | 90 |

* Little-endian

|  |  |
| --- | --- |
| Address | Value |
| 1003 | 90 |
| 1002 | AB |
| 1001 | 12 |
| 1000 | CD |

**Float and double types**

* Float type uses 4 bytes for a single precision floating point number
* Double type uses 8 bytes for double precision floating point numbers

**Variables**

* A variable is a name (identifier) used on source code to represent a data value of a certain type
* A variable is assigned a memory block with relative address at the compile time, as well as instructions to set and get the values to the memory block
* The memory block of a variable is instantiated with the absolute address at runtime

Brief description: a variable is an identifier of a data value in a program, it gets memory allocation at compile time and instanced at runtime.

* A variable must be declared with a type and name
* The variable declaration lets compiler to assign memory block with relative address, and generate instructions to allocate memory space for the variable at runtime
* A variable has to be initiated or assigned a value by assignment before it can be used
* C variable names must start with a letter, follow by letter, underscores and numbers, and case sensitive (use snake case)

**Variable scope**

* Each variable has a scope of the block where it is declared and be accessed by statements within the block. A variable has to be declared before it can be used
* Global variables are variables declared not in any block and can be used anywhere
* Local variables are ones declared within a block

**Sizeof**

* Sizeof is a unary operator used to calculate the sizes of data types
* Can be applied to all data types
* Returns the size of the variable, data type or expression in bytes
* Used to determine

**Constants**

* Preproccesor can be used to define constatns

Ex.

#define PI 3.1415926

* Now anytime you type PI it will use the value listed above