Quora:

<https://www.geeksforgeeks.org/how-to-prepare-for-acm-icpc/>

<https://www.quora.com/What-are-the-best-websites-a-programmer-should-visit>

<https://www.quora.com/Where-can-I-find-difficult-algorithm-data-structure-problems>

<http://assoc.cse.nitc.ac.in/wiki/Best_Learning_Material>

<http://courses.csail.mit.edu/iap/interview/materials.php>

<http://algomuse.appspot.com/>

<https://www.research.ibm.com/haifa/ponderthis/challenges/February2016.html>

<http://www.hiredintech.com/plans/>

<http://www.geeksforgeeks.org/understanding-extern-keyword-in-c/>

<http://www.geeksforgeeks.org/scope-rules-in-c/>

<http://www.geeksforgeeks.org/eof-and-feof-in-c/>

<http://www.geeksforgeeks.org/fopen-for-an-existing-file-in-write-mode/>

<http://www.geeksforgeeks.org/g-fact-82/> -fseek vs rewind

<http://stackoverflow.com/questions/17236242/usage-of-b-and-r-in-c>

<http://www.geeksforgeeks.org/output-of-a-program-set-2/>

<http://www.geeksforgeeks.org/gets-is-risky-to-use/>

<http://www.geeksforgeeks.org/memory-layout-of-c-program/>

<http://www.geeksforgeeks.org/calloc-versus-malloc/>

<http://www.geeksforgeeks.org/g-fact-80/>

<http://www.geeksforgeeks.org/understanding-register-keyword/>

<http://www.geeksforgeeks.org/understanding-extern-keyword-in-c/>

<http://www.geeksforgeeks.org/const-qualifier-in-c/>

<https://msdn.microsoft.com/en-us/library/888bfst6(VS.80).aspx>

<https://msdn.microsoft.com/en-us/library/w9hwbe3d(v=vs.80).aspx>

<http://geeksquiz.com/c-language-set-1-introduction/>

<https://en.wikipedia.org/wiki/Lexical_analysis>

<http://geeksquiz.com/c-arrays-question-6/>

<http://www.geeksforgeeks.org/why-c-treats-array-parameters-as-pointers/>

<http://www.geeksforgeeks.org/g-fact-95/>

<http://www.geeksforgeeks.org/g-fact-80/>

<http://www.geeksforgeeks.org/how-to-count-variable-numbers-of-arguments-in-c/>

<http://www.geeksforgeeks.org/storage-for-strings-in-c/>

<http://www.geeksforgeeks.org/swap-strings-in-c/>

<http://www.geeksforgeeks.org/complicated-declarations-in-c/>

<http://www.geeksforgeeks.org/find-a-pair-with-the-given-difference/>

<http://www.geeksforgeeks.org/using-sizof-operator-with-array-paratmeters/>

# Variables and Keywords in C

<http://geeksquiz.com/void-pointer-c/>

<http://geeksquiz.com/commonly-asked-c-programming-interview-questions-set-1/>

<http://www.geeksforgeeks.org/const-qualifier-in-c/>

<http://www.geeksforgeeks.org/g-fact-19-redeclaration-of-global-variable-in-c/>

C allows a global variable to be declared again when first declaration doesn’t initialize the variable.

<http://www.geeksforgeeks.org/complicated-declarations-in-c/>

<http://ieng9.ucsd.edu/~cs30x/rt_lt.rule.html>

<http://www.geeksforgeeks.org/how-linkers-resolve-multiply-defined-global-symbols/>

# Data Types in C

Data Type Memory (bytes) Range Format Specifier

short int 2 -32,768 to 32,767 %hd

unsigned short int 2 0 to 65,535 %hu

unsigned int 4 0 to 4,294,967,295 %u

int 4 -2,147,483,648 to 2,147,483,647 %d

long int 4 -2,147,483,648 to 2,147,483,647 %ld

unsigned long int 4 0 to 4,294,967,295 %lu

long long int 8 -(2^63) to (2^63)-1 %lld

unsigned long long int 8 0 to 18,446,744,073,709,551,615 %llu

signed char 1 -128 to 127 %c

unsigned char 1 0 to 255 %c

float 4 %f

double 8 %lf

long double 12 %Lf

<http://www.geeksforgeeks.org/g-fact-94/>

<http://www.geeksforgeeks.org/comparison-float-value-c/>

------------------------------------------------------------------------------

[[*Decimal to Floating-Point Conversions*](http://sandbox.mc.edu/~bennet/cs110/flt/dtof.html)] [[*Float to Decimal Conversion*](http://sandbox.mc.edu/~bennet/cs110/flt/ftod.html)]

## The Conversion Procedure

The rules for converting a decimal number into floating point are as follows:

1. Convert the absolute value of the number to binary, perhaps with a fractional part after the binary point. This can be done by converting the integral and fractional parts separately. The integral part is converted with the techniques examined previously. The fractional part can be converted by multiplication. This is basically the inverse of the division method: we repeatedly multiply by 2, and harvest each one bit as it appears left of the decimal.
2. Append × 20 to the end of the binary number (which does not change its value).
3. Normalize the number. Move the binary point so that it is one bit from the left. Adjust the exponent of two so that the value does not change.
4. Place the mantissa into the mantissa field of the number. Omit the leading one, and fill with zeros on the right.
5. Add the bias to the exponent of two, and place it in the exponent field. The bias is 2*k*−1 − 1, where *k* is the number of bits in the exponent field. For the eight-bit format, *k* = 3, so the bias is 23−1 − 1 = 3. For IEEE 32-bit, *k* = 8, so the bias is 28−1 − 1 = 127.
6. Set the sign bit, 1 for negative, 0 for positive, according to the sign of the original number.

* Convert 2.625 to our 8-bit floating point format.
  1. The integral part is easy, 210 = 102. For the fractional part:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0.625 | × 2 = | 1.25 | 1 | *Generate 1 and continue with the rest.* |
| 0.25 | × 2 = | 0.5 | 0 | *Generate 0 and continue.* |
| 0.5 | × 2 = | 1.0 | 1 | *Generate 1 and nothing remains.* |

* 1. So 0.62510 = 0.1012, and 2.62510 = 10.1012.
  2. Add an exponent part: 10.1012 = 10.1012 × 20.
  3. Normalize: 10.1012 × 20 = 1.01012 × 21.
  4. Mantissa: 0101
  5. Exponent: 1 + 3 = 4 = 1002.
  6. Sign bit is 0.

The result is 01000101. Represented as hex, that is 4516.

<http://www.geeksforgeeks.org/understanding-extern-keyword-in-c/>

extern:

By default, the declaration and definition of a C function have “extern” prepended with them. It means even though we don’t use extern with the declaration/definition of C functions, it is present there.

<http://geeksquiz.com/static-variables-in-c/>

In C, static variables can only be initialized using constant literals.

# Storage Classes:

<http://www.geeksforgeeks.org/g-fact-53/>

<http://www.geeksforgeeks.org/understanding-register-keyword/>

C doesn’t allow multiple storage class specifiers for a variable

<http://www.geeksforgeeks.org/understanding-volatile-qualifier-in-c/>

<http://www.geeksforgeeks.org/memory-layout-of-c-program/>

**Input/Output:**

<http://www.geeksforgeeks.org/scansets-in-c/>

<http://www.geeksforgeeks.org/g-fact-31/>

<http://geeksquiz.com/difference-getchar-getch-getc-getche/>

**Operators**:

<http://www.geeksforgeeks.org/c-operator-precedence-associativity/>

<http://www.geeksforgeeks.org/difference-between-p-p-and-p/>

<http://www.geeksforgeeks.org/a-comma-operator-question/>

<http://www.geeksforgeeks.org/comna-in-c-and-c/>

<http://www.geeksforgeeks.org/interesting-facts-bitwise-operators-c/>

<http://www.geeksforgeeks.org/evaluation-order-of-operands/>

result of comma operator as l-value is not valid in C

<http://www.geeksforgeeks.org/g-fact-78/>

<http://www.geeksforgeeks.org/g-fact-9/>

<http://geeksquiz.com/sizeof-operator-c/>

<http://www.geeksforgeeks.org/cc-ternary-operator-some-interesting-observations/>

<http://www.geeksforgeeks.org/sequence-points-in-c-set-1/>

A [sequence point](http://en.wikipedia.org/wiki/Sequence_point) defines any point in a computer program’s execution at which it is guaranteed that all side effects of previous evaluations will have been performed, and no side effects from subsequent evaluations have yet been performed

<http://www.geeksforgeeks.org/to-find-sum-of-two-numbers-without-using-any-operator/>

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Associativity** |
| ( ) [ ] . -> ++ -- | Parentheses (function call) (see Note 1) Brackets (array subscript) Member selection via object name Member selection via pointer Postfix increment/decrement (see Note 2) | left-to-right |
| ++ -- + - ! ~ (*type*) \* & sizeof | Prefix increment/decrement Unary plus/minus Logical negation/bitwise complement Cast (convert value to temporary value of *type*) Dereference Address (of operand) Determine size in bytes on this implementation | right-to-left |
| \*  /  % | Multiplication/division/modulus | left-to-right |
| +  - | Addition/subtraction | left-to-right |
| <<  >> | Bitwise shift left, Bitwise shift right | left-to-right |
| <  <= >  >= | Relational less than/less than or equal to Relational greater than/greater than or equal to | left-to-right |
| ==  != | Relational is equal to/is not equal to | left-to-right |
| & | Bitwise AND | left-to-right |
| ^ | Bitwise exclusive OR | left-to-right |
| | | Bitwise inclusive OR | left-to-right |
| && | Logical AND | left-to-right |
| | | | Logical OR | left-to-right |
| ? : | Ternary conditional | right-to-left |
| = +=  -= \*=  /= %=  &= ^=  |= <<=  >>= | Assignment Addition/subtraction assignment Multiplication/division assignment Modulus/bitwise AND assignment Bitwise exclusive/inclusive OR assignment Bitwise shift left/right assignment | right-to-left |
| , | Comma (separate expressions) | left-to-right |

**Preprocessor:**

<http://www.geeksforgeeks.org/the-offsetof-macro/>

<http://www.geeksforgeeks.org/interesting-facts-preprocessors-c/>

**Arrays:**

<http://www.geeksforgeeks.org/difference-pointer-array-c/>

<http://www.geeksforgeeks.org/using-sizof-operator-with-array-paratmeters/>

<http://www.geeksforgeeks.org/are-array-members-deeply-copied/>

<http://www.geeksforgeeks.org/write-one-line-functions-for-strcat-and-strcmp/>

strcat and strcmp as recursive

<http://www.geeksforgeeks.org/dynamically-allocate-2d-array-c/>

see 4th method

<http://www.geeksforgeeks.org/pass-2d-array-parameter-c/>

**Functions**:

<http://www.geeksforgeeks.org/importance-of-function-prototype-in-c/>

<http://www.geeksforgeeks.org/g-fact-95/>

In C, if a function is called before its declaration, the **compiler assumes return type of the function as int**.

<http://www.geeksforgeeks.org/functions-that-are-executed-before-and-after-main-in-c/>

**Control Statements:**

<http://www.geeksforgeeks.org/a-nested-loop-puzzle/>

<http://www.geeksforgeeks.org/interesting-facts-about-switch-statement-in-c/>

**Pointers:**

<http://www.geeksforgeeks.org/function-pointer-in-c/>

<http://www.geeksforgeeks.org/generic-linked-list-in-c-2/>

**others:**

getchar() and putchar() functions.

getword and getline functions

srand(time(0))-to get random number different on every run of program.

Num & num-1 remove one 1’s from number.-use for cal no of 1’s in a num