Cheatography

# C Programming Language Basics Cheat Sheet

by Randomgirlll13 via cheatography.com/186328/cs/39141/

## General Knowledge

| | |
|---|---|
| Syntax in C | All Statements in C must end with a semicolon ;<br>keywords and other code elements are case-sensitive |
| Escape Sequences | In C, within a string the \n will cut the string and move the part after to the next line. The \t will tab the string after, giving it more space |
| Single Line Comments | //This is a single line comment within c |
| Block Comments | /* *Block comments are nice for when you have a lot to say about a particular piece of code* */ |
| Compiling C Code | to compile c code, in a terminal you'll need to type in, minimally, gcc fileName.c<br>this will output something call a.out which will allow you to run the program<br>Alternatively, if you do gcc fileName.c -o fileName you can then run it using the denotation of filename instead |

## Conditionals

| | |
|---|---|
| if Statements | if(x == 3){<br>printf(x);<br>} |
| else-if Statements | if(x == 3){<br>printf(x);<br>} else if (x < 3) {<br>printf(x - 1);<br>} |
| else statements | if(x == 3){<br>printf(x);<br>} else if (x < 3) {<br>printf(x - 1);<br>} else {<br>print(0);<br>} |
| Ternary Operators: A condensed if-else statement | if (a < b) {<br>min = a;<br>} else {<br>min = b;<br>}<br><br>min = a < b ? a : b; ((This is the ternary Operator for the above if)<br>to briefly explain it, it essentially will set min to a if the a < b is true otherwise b, the second option, will be what min is set to |

## Conditionals (cont)

| | |
|---|---|
| Switch Statements: A Condensed series of cascading else statements, it tests a value and compares it against multiple cases. | switch (grade)<br>{<br>case 9:<br>printf("Freshman\n");<br>break;<br>default:<br>printf("Invalid\n");<br>break;<br>} |

## Functions

| | |
|---|---|
| Functions in C | A function is a block of reusable logic that may have a defined set of input and output |
| Built-In Function in C | The C Programming language comes with built-in standard library fuctions<br>- printf()<br>-rand()<br>**Note to be able to use these make sure to #include <stdio.h> |
| Calling Functions | int myNumber = incrementBy(5, 2);<br><br>A function is called by stating the function name followed by parentheses. One or more argument values can be places in the parentheses as the function requires. |

## Functions (cont)

| | |
|---|---|
| Storing a return value | int myNumber = incrementBy(5, 2);<br><br>A function's return value, or output, can be stored in a variable for future use. |
| Function Signature | |
| Return Type | |

## Variables

| | |
|---|---|
| Variable Names in C | In C, Variable names follow specific Rules:<br>Names can only be composed of upper and lower case letters, numbers, and underscores.<br>The first character must be a letter (upper or lower case).<br>No keywords are allowed as the full name (int is not allowed but int_count would work) |
| Data Types in C | The four main data types in c are: int, char, double, and float |
| Declaring Variables in C | int int_count = 4;<br>double priceApples; |
| Setting Variables in C | When you declare a variable, you do not need to set it right away, but you can set it right away if you want to |

## Variables (cont)

| | |
|---|---|
| Variable Casting in C | You can implicitly or explicitly cast to variables in C. However, implicitly casting may not have the same effect as explicitly casting |

## Loops and Errors

| | |
|---|---|
| While Loops, will iteratre until a condition is met | while (a < 10) {<br>a++;<br>} |
| do -While loops, while loops that initially execute the body once before checking conditions | do {<br>printf("not true!");<br>} (while 2 == 3); |
| for loops, iterates a set number of times | for (int i = 0; i <= 10; i++) {<br>printf("Hello!");<br>} |
| Loop Keywords | All Loops can Utilize Keywords like continue and break.<br>Continue will restart the loop without completing anything within the loop past the continue keyword.<br>break will completely stop the loop and continue on after it within the code. |

## Pointers and Memory

| | |
|---|---|
| What is a pointer? | A pointer is a variable that sotres the hexadecimal address of the variable it is pointing to within memory |
| Declaring pointers | type* pntr;<br>type *pntr; |
| Accessing Memory Address | A Memory Address of a vairiable is obtained using the reference operator (&).<br>*example &var |
| Derefe-rencing Pointers | A Pointer is dereference using the dereference operator (*)<br>*Example *pntr |
| Increm-enting and Decrem-enting Pointers | Pointers can be incremented and decremented using the + and - arithmetic operators |
| Accessing Arrays | Arrays can be accessed by using a pointer to the first element and icrementing and decrementing as necessary |

By **Randomgirlll13**

cheatography.com/randomgirlll13/

Not published yet.
Last updated 17th August, 2023.

Cheatography

# C Programming Language Basics Cheat Sheet

by Randomgirlll13 via cheatography.com/186328/cs/39141/

## Operators

| | |
|---|---|
| Mathmatical Symbols in C | Addition: +<br>Subtraction: -<br>Division: /<br>Multiplication: *<br>Incrementing: ++<br>Decrementing: --<br>Modulo: % |
| Assignment Operations in C | Assignment: =<br>Addition then Assignment: +=<br>Subtraction then Assignment: -=<br>Multiplication then Assignment: *=<br>Division then Assignment: /=<br>Modulo then Assignment: %= |
| Comparing Values in C | Both sides Equal?: ==<br>Two sides not equal?: !=<br>left lower then right?: <<br>left lower or equal to right?: <=<br>left greater than right?: ><br>left greater than or equal to right?: >= |

## Operators (cont)

| | |
|---|---|
| Logical Operators in C | and: && (Both sides true?)<br>or: \|\| (At least one side true?)<br>not: ! (true = false, false = true) |

## Arrays and Strings

| | |
|---|---|
| Creating Uninitialized Arrays | type arr[array_size];<br>char word[15]; |
| Creating initilized Arrays | type arr[] = {el1, el2};<br>char word[] = {'H', 'e', 'l', 'l', 'o', '\0'} |
| Accessing Array Elements | arr[idx];<br>word[0]; ('H') |
| First and Last Array Elements | firstElement = arr[0];<br>lastElement = arr[arraySize - 1]; |
| sizeof()<br>**Note** C does not have a built in way to actually find the size of an array, this is simply a handy trick to find the amount of elements an array should be able to hold. | int arr[17];<br>size_t sizeArr = sizeof(arr) / sizeof(arr[0]);<br>**Note size_t is an unsigned integer type used to represent the size of objects in bytes, its the return type of sizeof() |

## Arrays and Strings (cont)

| | |
|---|---|
| Invalid Array Access | While it is possible to access beyond an array, it will cause the program to behave unpredictably |
| Creating Multidimensional Arrays | unitializedMultArr = type arr[12][15]...[13];<br>initializedMultArr = type arr[][15][20]...[n] = {{elm1, elm2}, {elm3}} |
| Arrays are static | The length of a string cannot be modified as a string is a char array in C |
| Null Character | All Strings terminate with a null character ('\0') |
| The length of a string | find the length using the strlen() function |
| string concatenation | strcat() function can be used to concate two strings |
| strcpy() | a string can be copied into an empty char array using this function |