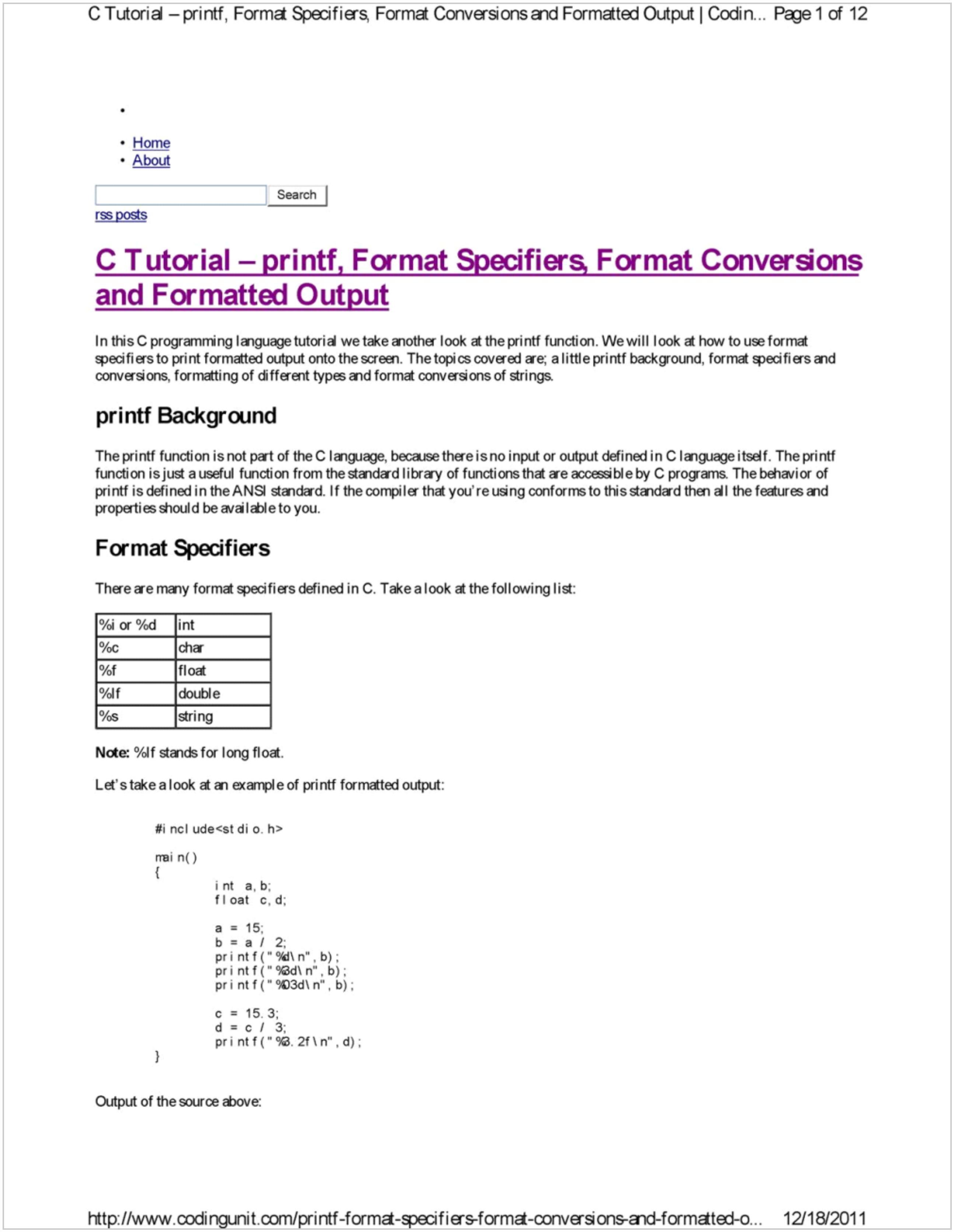
Sunday, December 18, 2011
9:05 AM



C Tutorial – printf, Format Specifiers, Format Conversions and Formatted Output

In this C programming language tutorial we take another look at the printf function. We will look at how to use format specifiers to print formatted output onto the screen. The topics covered are; a little printf background, format specifiers and conversions, formatting of different types and format conversions of strings.

printf Background

The printf function is not part of the C language, because there is no input or output defined in C language itself. The printf function is just a useful function from the standard library of functions that are accessible by C programs. The behavior of printf is defined in the ANSI standard. If the compiler that you're using conforms to this standard then all the features and properties should be available to you.

Format Specifiers

There are many format specifiers defined in C. Take a look at the following list:

%i or %d	int
%c	char
%f	float
%lf	double
%s	string

Note: %lf stands for long float.

Let's take a look at an example of printf formatted output:

```
#include<stdio.h>

main()
{
	int  a,b;
	float  c,d;

	a = 15;
	b = a / 2;
	printf("%d\n",b);
	printf("%3d\n",b);
	printf("%03d\n",b);

	c = 15.3;
	d = c / 3;
	printf("%3.2f\n",d);
}
```

Output of the source above:

```
7
  7
007
5.10
```

As you can see in the first printf statement we print a decimal. In the second printf statement we print the same decimal, but we use a width (%3d) to say that we want three digits (positions) reserved for the output.
The result is that two "space characters" are placed before printing the character. In the third printf statement we say almost the same as the previous one. Print the output with a width of three digits, but fill the space with 0.

In the fourth printf statement we want to print a float. In this printf statement we want to print three position before the decimal point (called width) and two positions behind the decimal point (called precision).

The \n used in the printf statements is called an escape sequence. In this case it represents a newline character. After printing something to the screen you usually want to print something on the next line. If there is no \n then a next printf command will print the string on the same line. Commonly used escape sequences are:

- \n (newline)
- \t (tab)
- \v (vertical tab)
- \f (new page)
- \b (backspace)
- \r (carriage return)
- \n (newline)

Let's take another look at a printf formatted output in a more application like example:

```
#include<stdio.h>

main()
{
        int Fahrenheit;

        for (Fahrenheit = 0; Fahrenheit <= 300; Fahrenheit = Fahrenheit + 20)
                printf("%3d %06.3f\n", Fahrenheit, (5.0/9.0)*(Fahrenheit-32));
}
```

Output of the source above:

```
  0 -17.778
 20 -6.667
 40 04.444
 60 15.556
 80 26.667
100 37.778
120 48.889
140 60.000
160 71.111
180 82.222
200 93.333
220 104.444
240 115.556
260 126.667
280 137.778
300 148.889
```

As you can see we print the Fahrenheit temperature with a width of 3 positions. The Celsius temperature is printed with a width of 6 positions and a precision of 3 positions after the decimal point. Let's recap:

- %d (print as a decimal integer)
- %6d (print as a decimal integer with a width of at least 6 wide)
- %f (print as a floating point)

- %4f (print as a floating point with a width of at least 4 wide)
- %.4f (print as a floating point with a precision of four characters after the decimal point)
- %3.2f (print as a floating point at least 3 wide and a precision of 2)

Formatting other Types

Until now we only used integers and floats, but there are more types you can use. Take a look at the following example:

```
#include<stdio.h>

main()
{
        printf("The color: %s\n", "blue");
        printf("First number: %d\n", 12345);
        printf("Second number: %04d\n", 25);
        printf("Third number: %i\n", 1234);
        printf("Float number: %3.2f\n", 3.14159);
        printf("Hexadecimal: %x\n", 255);
        printf("Octal: %o\n", 255);
        printf("Unsigned value: %u\n", 150);
        printf("Just print the percentage sign %%\n", 10);
}
```

Output of the source example:

```
The color: blue
First number: 12345
Second number: 0025
Third number: 1234
Float number: 3.14
Hexadecimal: ff
Octal: 377
Unsigned value: 150
Just print the percentage sign %
```

Note: In the last printf statement only the percentage sign is printed.

The number 10 in this statement doesn't matter; it's not used in the output. So if you want to print a percentage number you would use something like this: printf("%2d%%\n", 10); (The output will be 10%)

Formatting Strings

By now you have seen most of the format conversion possible, but there is one type that is a little different
and that are string format conversions. Take a look at the following example:

```
#include<stdio.h>

main()
{
        printf(":%s:\n", "Hello, world!");
        printf(":%15s:\n", "Hello, world!");
        printf(":%.10s:\n", "Hello, world!");
        printf(":%-10s:\n", "Hello, world!");
        printf(":%-15s:\n", "Hello, world!");
        printf(":%.15s:\n", "Hello, world!");
        printf(":%15.10s:\n", "Hello, world!");
        printf(":%-15.10s:\n", "Hello, world!");
}
```

The output of the example above:

```
:Hello, world!:
:  Hello, world!:
:Hello, wor:
:Hello, world!:
:Hello, world!   :
:Hello, world!:
:     Hello, wor:
:Hello, wor     :
```

As you can see, the string format conversion reacts very different from number format conversions.

- The printf(":%s:\n", "Hello, world!"); statement prints the string (nothing special happens.)
- The printf(":%15s:\n", "Hello, world!"); statement prints the string, but print 15 characters. If the string is smaller the "empty" positions will be filled with "whitespace."
- The printf(":%.10s:\n", "Hello, world!"); statement prints the string, but print only 10 characters of the string.
- The printf(":%-10s:\n", "Hello, world!"); statement prints the string, but prints at least 10 characters. If the string is smaller "whitespace" is added at the end. (See next example.)
- The printf(":%-15s:\n", "Hello, world!"); statement prints the string, but prints at least 15 characters. The string in this case is shorter than the defined 15 character, thus "whitespace" is added at the end (defined by the minus sign.)
- The printf(":%.15s:\n", "Hello, world!"); statement prints the string, but print only 15 characters of the string. In this case the string is shorter than 15, thus the whole string is printed.
- The printf(":%15.10s:\n", "Hello, world!"); statement prints the string, but print 15 characters.
  If the string is smaller the "empty" positions will be filled with "whitespace." But it will only print a maximum of 10 characters, thus only part of new string (old string plus the whitespace positions) is printed.
- The printf(":%-15.10s:\n", "Hello, world!"); statement prints the string, but it does the exact same thing as the previous statement, accept the "whitespace" is added at the end.

A little warning!
The printf function uses its first argument to determine how many arguments will follow and of what types they are. If you don't use enough arguments or if they are of the wrong type than printf will get confuses, with as a result wrong answers.

That's all for this C tutorial. Just make some examples of your own, they are easy to make. This is the only way to learn and see how the format conversions reacts.

This entry was posted in C Tutorials. You can follow any responses to this entry through the RSS 2.0 feed. You can leave a response, or trackback from your own site. Tweet This! **or use** ShareThis **to share this post with others.**

There are currently 61 responses to "C Tutorial – printf, Format Specifiers, Format Conversions and Formatted Output"

Why not let us know what you think by adding your own comment!

1. *Joe* on November 11th, 2009:

   this tutorial is very good!!!!!

2. *Rohit Rajpoot* on November 28th, 2009:

   Yes....
   This tutorial is really very helpful!!!!
   I like this...

3. *Deepak* on December 9th, 2009:

   This tutorial made my job easy
   Thanks a lot

4. *kaustav* on December 13th, 2009:

   beautiful tutorial. thanx.

5. *Aaditya kumar* on January 23rd, 2010:

i get confuse in formatting,this description is enough to show how these are working and helpful in formatting even digit or string.

6. *Matt* on February 3rd, 2010:

   I think you have an error near the bottom of the tutorial when summarizing string formatting…

   'The printf(":%10s:\n", "Hello, world!"); statement prints the string, but print 15 characters. If the string is smaller the "empty" positions will be filled with "whitespace." '

   This should say:

   'The printf(":%15s:\n", "Hello, world!"); statement prints the string, but print 15 characters. If the string is smaller the "empty" positions will be filled with "whitespace." '

7. admin on February 3rd, 2010:

   Thx, I have corrected the typo!

8. *Armando* on February 12th, 2010:

   I have a problem when printing large floating point numbers.

   I tried the following solution but it doesn't work.

   ```
   #include <stdio.h>
   main()
   {
   int i;
   double arr[3] =
   {12345678.000, 98765345.333, 456793332.300};
   for (i = 0; i < 3; i++)
   printf ("%g\n", arr[i]);
   }
   ```

   The output is this:
   ```
   1.23457e+07
   9.87653e+07
   4.56793e+08
   ```

   What I need is this:
   ```
   12345678
   98765345.333
   456793332.3
   ```

   Could anyone help me?

9. Social Media Graphics on February 12th, 2010:

   @Armando – just change printf ("%g\n", arr[i]); to printf ("%f\n", arr[i]);

10. *Armando* on February 17th, 2010:

    @Social Media Graphics thanks for your suggestion.

    But what I really need is to print the decimal significant if any, otherwise only the integer part.

11. *cheeloon* on March 4th, 2010:

    Very good and detail explanation. Thx alot.

12. *Joe* on March 12th, 2010:

if i have 1294300000000000.000001,
how do i get 0.000001?
i only want the decimal places….
does anybody know?

13. *prashanth* on April 7th, 2010:

    your tutorial was very helpful to me……..thank a lot……

14. *David* on April 8th, 2010:

    Joe: to get the decimal part of a float (or double), try this:

    float x=1294300000000000.000001;

    float decimal = x – (int)x;

    the conversion to an int should truncate the decimal off. However, I should warn that decimal might not be exactly .000001 here because that is actually impossible to represent in binary, because it is in fact in binary a repeating "decimal." You might also get nicer results using larger data types like doubles and longs.

15. *Shashika* on July 7th, 2010:

    Thanks For Your Very Helpful Tutorial

16. *selva* on July 8th, 2010:

    thank u ….. superb explanation …. it made me clear in formatting strings…..

17. *pramit* on July 17th, 2010:

    answer to armando problem on 12 th feb.

    just write
    printf("%.8g\n",arr[i]);
    printf("%.11g\n",arr[i]);
    printf("%.10g\n",arr[i]);

    note:u have to write like above,but not in loop bcoz this only serve ur requirement.

    waiting for ur another problem.

18. *Tushar Srivastava* on July 31st, 2010:

    Good stuff for beginners but you should include some examples of %U format specifier with its explanation. Best of Luck

19. *Holo* on August 19th, 2010:

    Very good exmples – I am using this in awk
    but How do I print 10,123,456 – comma after each thousand. I saw some where else to use "%'d" but that is causing problem. ANy ideas.
    Thanks

20. *Matt Bates* on August 24th, 2010:

    i am trying to print two strings that are stored in char arrays. I can't get them onto the same line.

    char firstName[50];

    char lastName[50];

    printf("Your name is %s", firstName, lastName);

it prints out:

Your name is 'firstName'
'lastName'

how do i get them onto the same line??? In other words, how to I add to the printf function?

21. *admin* on August 25th, 2010:

@ Matt Bates – The source code you have given should only print 'firstName', because you only use one %s in your printf statement. So I don't know why it's also printing 'lastName' (maybe an extra printf statement that you've not put in your example.) But below you'll find an example that should do what you want:

```
#include
void main() {
char *firstName;
char *lastName;

firstName = "John";
lastName = "Doe";
printf("Your name is %s %s", firstName, lastName);
}
```

Good luck!

22. *tallen387* on September 23rd, 2010:

How do you print a number in binary format (similar to %o, %d, %x)?

Thanks!

23. *admin* on September 24th, 2010:

There is no format (similar to %d) to print a integer in binary format, you have to build it yourself, a int2bin if you will. Something like this will do:

```
#include<stdio.h>

void bin(int i) {
int a=0;
if(i!=0) {
a=i;
bin(i>>1);
printf(" %d ", a&0x01);
}
}

int main() {
int b=13;
bin(b);
return 0;
}
```

24. *sankar v* on November 11th, 2010:

thanks a lot. i have exam today and this helped me a lot. i had no idea about the printf thing

25. *Aggie* on December 5th, 2010:

thanks! Helped me pick it up quickly, better than wikipedia

26. *tanushri* on December 14th, 2010:

i like information abt printf

27. *Deon Joubert* on January 19th, 2011:

You could also use the "itoa" function (from stdlib.h) with a radix of 2 to do the conversion from an integer to a string that represents the binary value of the integer.

28. *Deon Joubert* on January 19th, 2011:

    @tallen387
    My previous comment is for you

29. *aaron* on January 24th, 2011:

    Hello,

    I need to print 10 numbers with one number per line and then beside each number display the running total of digits in the numbers.

    example:

    num digits
    3 1
    78 3
    890 6
    1000 10
    etc..

    I know a couple of methods to do this and they work. But I am asked to use the "%n conversion specifier" or the 'n' format specifier(not even sure what it is actually known as). I guess this would be within the printf as I print out the numbers. But I have tried to search around to find out how this conversion specifier works but have figured out barely anything. I am lost as to how this method would work in counting and displaying the running total of digits.

    If somebody could point me in the right direction I would greatly appreciate it.

    Thank you.

30. *nancy* on March 1st, 2011:

    vry useful tutorial

31. *sha* on March 25th, 2011:

    Great Tutorial. very informative…..

32. *Faisal* on April 13th, 2011:

    Really helpful and informative tutorial. I hope every one likes it.

33. *Orel* on May 6th, 2011:

    Best c printf tutorial so far.. 😀 great job!

34. *aayush* on May 9th, 2011:

    awesome man!!!!!!!! nice tut..

35. *Pisio* on May 11th, 2011:

    Hi all,
    I need to print a floating number (for example 1009.23) without the decimal point (in this case 100923), just to put it in a formatted positional output.
    Can I use printf in some way to do this?
    Thank you!

36. *ranjitha g* on May 20th, 2011:

    awesome…this tut helped me lot.

37. *Rahul Singh Chouhan* on May 27th, 2011:

    i need to know how to print bits or bytes character value in c

38. *praga* on June 7th, 2011:

    hi dude s this is praga ….i need to know exactly whats the reason that we enclose the words to be displayed….meant the words in printf to be enclosed in double quotes

39. *praga* on June 7th, 2011:

    this tut is gr8

40. *Printf() function for debugging process* on June 13th, 2011:

    [...] Have you checked out Keil's User Guide: Cx51 User's Guide – printf() A couple of good tutorials: C Tutorial – printf, Format Specifiers, Format Conversions and Formatted Output Secrets of "printf" The Keil compiler sends printf() output to STDOUT, which I [...]

41. *Anchal* on June 16th, 2011:

    nice explanation given!!!!!

42. *sahtihi* on June 29th, 2011:

    Very clear and detailed explanation is given

43. *Ganesh Auti,Pune* on July 4th, 2011:

    very nice tut…

44. *Aravind* on July 5th, 2011:

    Simple stuff but helps a lot in real life programming.
    Thanks for the information.

45. *Brindha* on July 15th, 2011:

    Excellent explanations….
    It helps a lot..
    Thank u…

46. *shirin k* on July 17th, 2011:

    nice tutorial…more about the format specifiers

47. *Rajesh Joshi* on July 18th, 2011:

    One of best information uploaded!

48. *smit* on July 30th, 2011:

    This a realy useful site for student

49. *navneeth* on August 4th, 2011:

    THANKS FOR THE INFORMATION.THIS WAS REALLY USEFUL TO ME

50. *jesen* on August 26th, 2011:

    Could you help me please
    I need show a message as a BOLD .
    But I don't know how to do this…
    Please help..

51. *thita nayak* on September 6th, 2011:

    thanx c tutorial………..u help me alot…………

52. *Arivoli* on September 15th, 2011:

    Your examples give me a clear idea about this topic..
    Thanks a lot…

53. *kaviraj* on September 19th, 2011:

    best c-tutorials website…. I ever visit…
    Facebook.com/kaviraj.kalsi

54. *avinash* on September 26th, 2011:

    print the value of variable without using format specifier in c

55. *dheeraj* on October 7th, 2011:

    thanks

56. *Abd* on October 9th, 2011:

    Thsnks. Helped me figure out the specifier for double!

57. *amrutanshu* on October 18th, 2011:

    awesome man….it rocks…^

58. *Csaba* on November 8th, 2011:

    Just what I needed, with great examples! Thank you!

59. *Usman Akram Punjab University Lahore* on November 19th, 2011:

    This website helps me a lot. thanx

60. *sunny* on November 19th, 2011:

    this is wat i want thanks…

61. *Neeba* on November 24th, 2011:

    good one.

Leave a Reply:

Name (required)

Mail (will not be published) (required)

Website

Submit Comment

« go backup to content »

C Tutorials

- The History of the C Language
- C Tutorial – Compilers (GNU and Visual Studio)
- First C program, Hello World
- C Tutorial – variables and constants
- C Tutorial – The if and switch statement
- C Tutorial – for loop, while loop, break and continue
- C Tutorial – Arrays and Multi-Dimensional Arrays
- C Tutorial – Functions and Global/Local variables
- C Tutorial – More on Functions
- C Tutorial – How to use Pointers
- C Tutorial – More on Pointers
- C Tutorial – strings and string Library Functions
- C Tutorial – printf, Format Specifiers, Format Conversions and Formatted Output
- C Tutorial – structures, unions, typedef
- C Tutorial – File I/O (using text files)
- C Tutorial – The functions malloc and free
- C Tutorial – Binary File I/O
- C Tutorial – Deleting and Renaming a File
- C Tutorial – Copying a File
- C Tutorial – Command Line Parameter Parsing
- How to use Time and Date in C
- Writing Memory to a File and Reading Memory from a File in C
- How to make a Calendar in C
- C Tutorial – Searching for Strings in a Text File
- C Tutorial – Number of Seconds in Decade and Visa-Versa
- C Tutorial – A Star pyramid and String triangle using for loops

Latest Posts

- PHP Tutorial – File Handling
- C Tutorial – A Star pyramid and String triangle using for loops
- C Tutorial – Number of Seconds in Decade and Visa-Versa
- C++ Binary Operator Overloading Greater or Less than
- PHP Tutorial – include and require
- Rewrite of one C++ Tutorial
- Unary and Binary Operator Table

- C Tutorial – Searching for Strings in a Text File
- C Reference String Operation: strstr()
- Move from NextDawn.nl to CodingUnit.com is done!