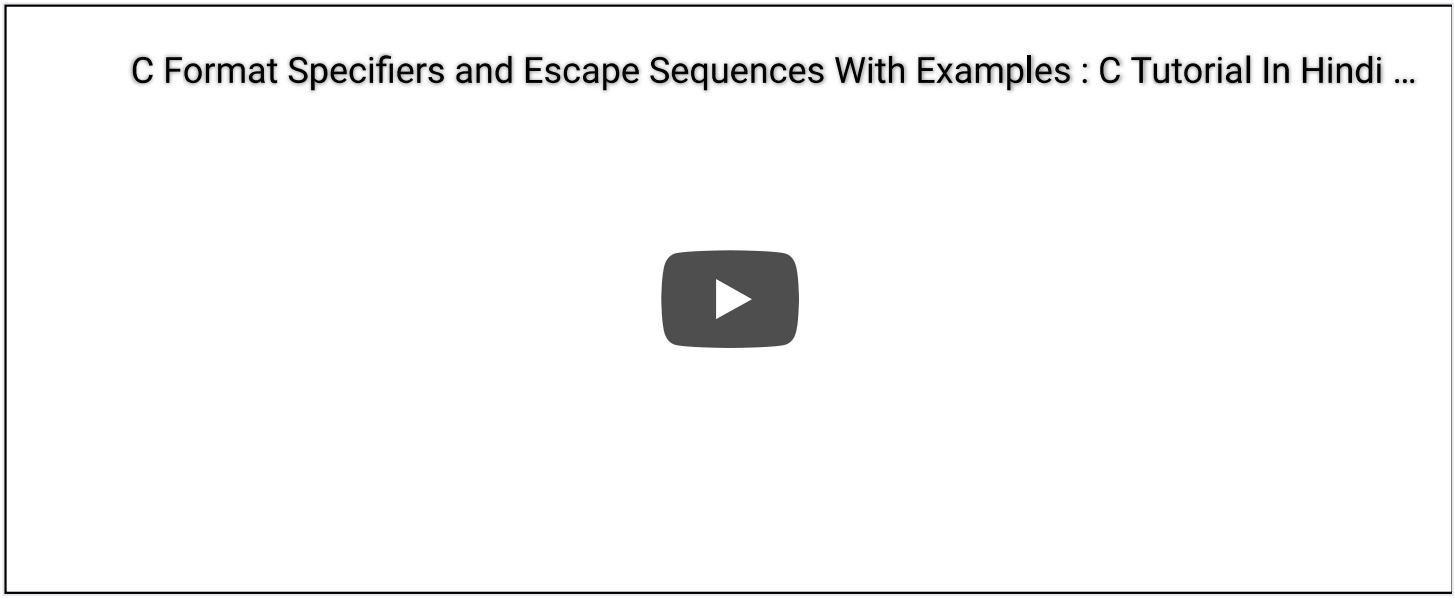
CodeWithHarry

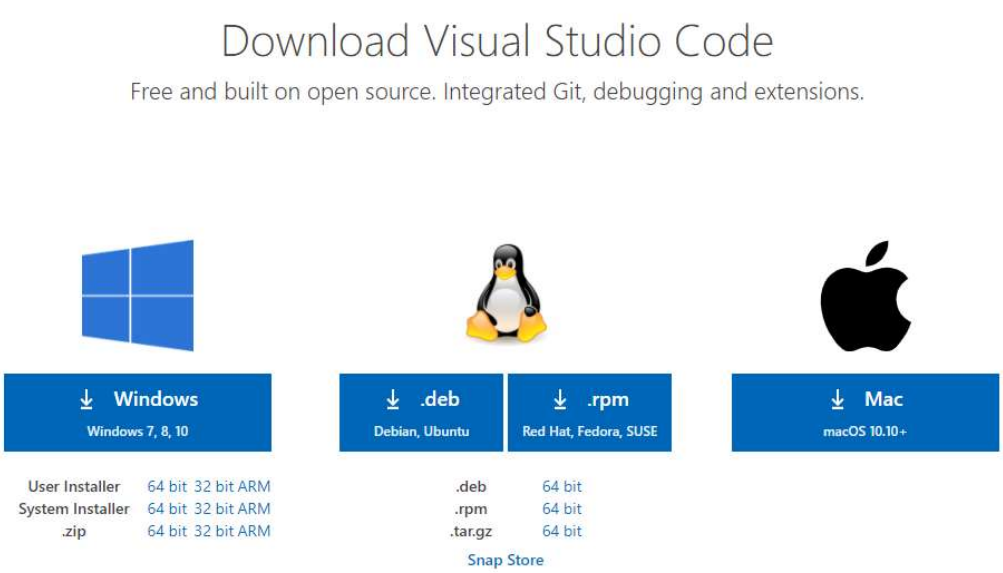


Course: C Language Tutorials For Beginners

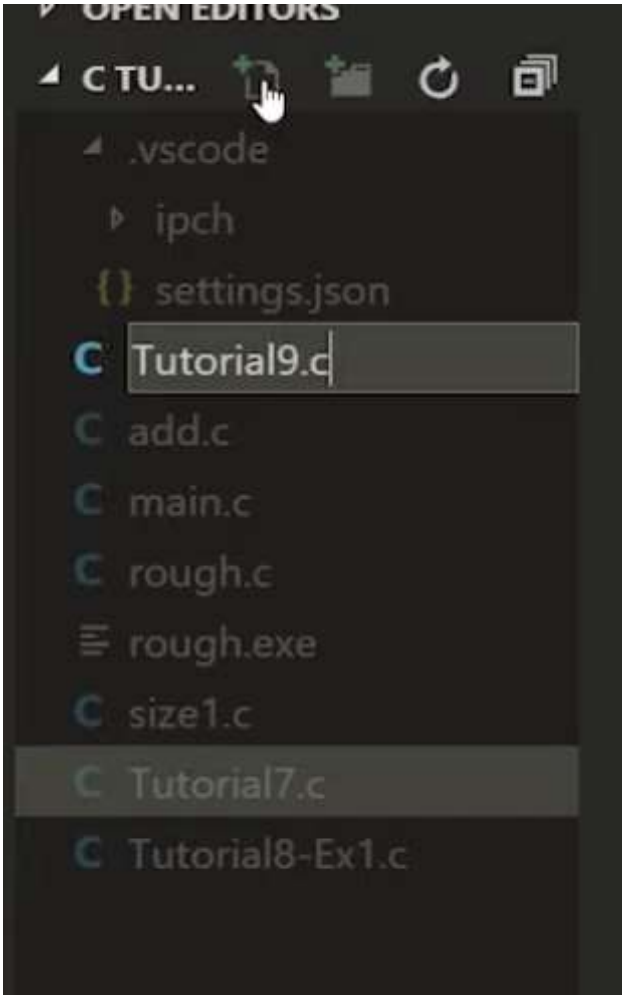
Overview Q&A Downloads Announcements

C Format Specifiers and Escape Sequences With Examples : C Tutorial In Hindi #9

To run a **C program**, we need an IDE's like Visual Studio Code or Code blocks. For this series, we are using Virtual Studio Code (VS Code). ***Visual Studio Code*** is a fast source code editor and provides the tools that a developer needs for efficient programming. To download VS Code, click on ***Download Visual Studio Code*** . For guidance, check the tutorial ***Install & Configure VS Code***



Now open VS Code and create a file with name **“tutorial9.c”**



In today's tutorial, we will learn about the format specifiers and escape characters as it is one of the most important concepts in C programming. Whenever we write a code in C, we have to use the format specifiers to define the variable type in input and output, and escape characters to format the output. So before exploring more C language concepts, first it will be better to have a strong grip on this concepts.

Format specifier in C:-

The format specifier in C programming is used for input and output purposes. Through this, we tell the compiler what type of variable we are using for input using scanf() or printing using printf(). Some examples are %d, %c, %f, etc.
The %c and %d used in the printf( ) are called format specifiers. **Format specifier** tells printf( ) to print the value, for %c, a character will printed, and for %d, a decimal will be printed.Here is a list of format specifiers.

Format Specifier	Type
%c	Used to print the character
%d	Used to print the signed integer
%f	Used to print the float values
%i	Used to print the unsigned integer
%l	Used to long integer
%lf	Used to print the double integer
%lu	Used to print the unsigned int or unsigned long integer

Format Specifier	Type
%s	Used to print the String
%u	Used to print the unsigned integer

Following are the few examples of format specifier:

- To print the integer value:

```
printf("\n The value of integer is %d", d);
```

- To print the float value:

```
printf("The value of float is %f", f);
```

- To print the character:

```
printf("\n The value of character is %c", c);
```

- To print the string:

```
printf("\n The value of string is %s", s);
```

What is Escape Sequence in C?

Many programming languages supports the concept of Escape Sequence. An escape sequence is a sequence of characters which are used in formatting the output. They are not displayed on the screen while printing. Each character has its own specific function like \t is used to insert a tab and \n is used to add newline.

Types of Escape Sequence in C

Escape Sequence	Description
\t	Inserts a tab
\b	Inserts a backspace
\n	Inserts a newline.
\r	Inserts a carriage return.
\f	Inserts a form feed.
\'	Inserts a single quote character.
\"	Inserts a double quote character.
****	Inserts a backslash character.

Following are the some examples of escape sequence:

- Print character backslash(\) using printf function

```
printf("\n C programming \m/ ");
```

- Prints a newline before and after the text

```
printf("\n This is my C program\n");
```

- Use \" to print double quote and \' for single quote

```
printf("\n Welcome to \"The C Programming tutorial\"");
```

```
printf("\n Welcome to \'C programming series \'") ;
```

- To provide tab space between two words

```
printf("Hello \t Viewers");
```

- To add vertical tab character.

```
printf("Hello Viewers");

printf("\v Welcome to C Programming");
```

Code as described/written in the video

```
#include <stdio.h>
#define PI 3.14
/* this is a multiline comment


this is ignored by my compiler

*/
int main()
```

```
{
    int a = 8;
    const float b = 7.333;
    // PI = 7.33; //cannot do this since PI is a constant
    printf(" tab character \t\t my backslash  %f", PI);
    // b = 7.22; //cannot do this since b is a constant
    // printf("Hello World\n");
    // printf("The value of a is %d and the value of b is %2.4f\n", a, b);
    // printf("%18.4f this",b);

    return 0;
}
```

Previous Next

CodeWithHarry