

CL1002 – Programming Fundamentals Lab



Lab # 04

Arithmetic Operators & Escape Sequences in C

Instructor: Engr. Muhammad Usman

Email: usman.rafiq@nu.edu.pk

Department of Computer Science,
National University of Computer and Emerging Sciences FAST Peshawar

Escape Sequences

Character combinations consisting of a backslash (\) followed by a letter or by a combination of digits are called "escape sequences." To represent a newline character, single quotation mark, or certain other characters in a character constant, you must use escape sequences. An escape sequence is regarded as a single character and is therefore valid as a character constant. Escape sequences are used to format our output. The following escape sequences can be used to print out special characters.

Escape Sequence	Description
\n	Newline
\t	Horizontal tab
\v	Vertical tab
\\	Backslash
\"	Double quote

To insert a line break, a new-line character shall be inserted at the exact position the line should be broken. In C, a new-line character can be specified as \n (i.e., a backslash character followed by a lowercase n).

Example 1

```
1 #include <stdio.h>
2 int main()
3 {
4     printf("First Sentence\n");
5     printf("Second Sentence\n");
6     printf("Third Sentence\n");
7
8     return 0;
9 }
```

This produces the following output:

```
usman@usman-7g-series: ~/pf/lab4
usman~/pf/lab4$ gcc ex1_escape.c -o ex1_escape.out
usman~/pf/lab4$ ./ex1_escape.out
First Sentence
Second Sentence
Third Sentence
usman~/pf/lab4$
```

Example 2

Following program shows the use of Newline Escape Sequence (`\n`).

```
1 #include <stdio.h>
2 int main()
3 {
4
5 printf("This\nis\na\ntest\n\nHe said, How are you?\n");
6
7 return 0;
8 }
```

Output

```
usman@usman-7g-series: ~...
usman~/pf/lab4$ gcc ex2.c -o ex2
usman~/pf/lab4$ ./ex2
This
is
a
test

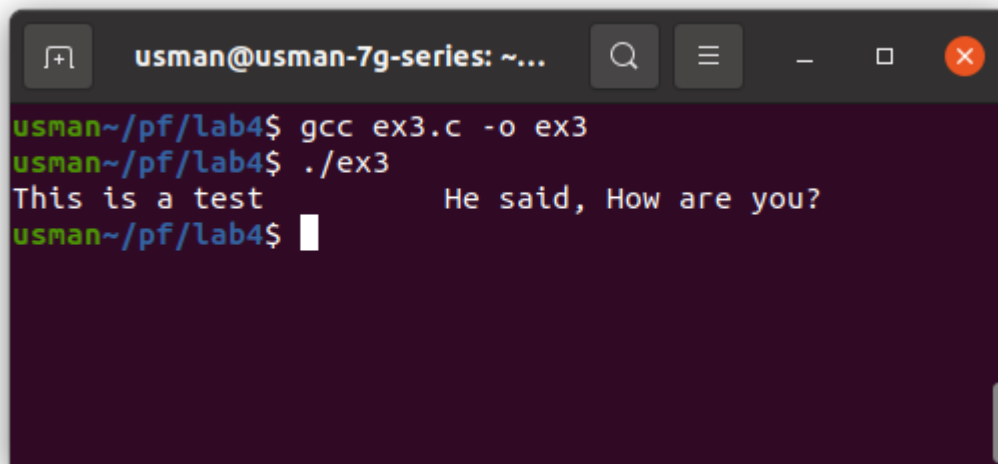
He said, How are you?
usman~/pf/lab4$
usman~/pf/lab4$
```

Example 3

This program shows the use of Horizontal tab Escape Sequence (`\t`).

```
1#include <stdio.h>
2int main()
3{
4
5printf("This is a test\t\tHe said, How are you?\n");
6
7return 0;
8}
```

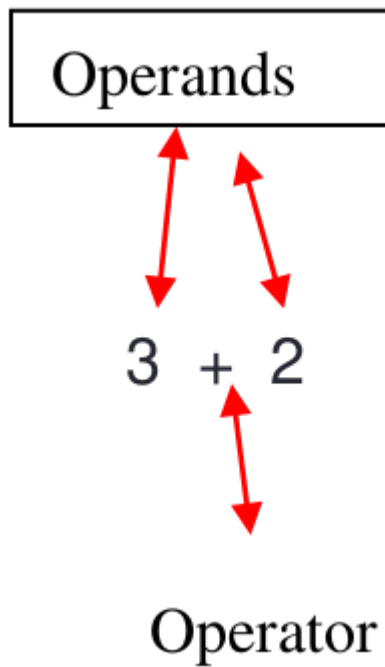
Output

A terminal window with a dark purple background. The title bar shows 'usman@usman-7g-series: ~...'. The prompt is 'usman~/pf/lab4\$'. The user enters 'gcc ex3.c -o ex3' and presses enter. The prompt is 'usman~/pf/lab4\$'. The user enters './ex3' and presses enter. The output is 'This is a test' followed by two horizontal tabs and then 'He said, How are you?'. The prompt is 'usman~/pf/lab4\$' with a cursor.

Now try escape sequences `\\`, `\v`, `\"` yourself.

Operators

Operators are special symbols that carry out arithmetic or logical computation. The value that the operator operates on is called the operand.



Here, + is the operator that performs addition. 2 and 3 are the operands and 5 is the output of the operation.

```
int sum1 = 100 + 50;    // 150 (100 + 50)
int sum2 = sum1 + 250;  // 400 (150 + 250)
int sum3 = sum2 + sum2; // 800 (400 + 400)
```

Arithmetic Operators

Arithmetic operators are used to perform mathematical operations like addition, subtraction, multiplication etc.

Operator	Meaning	Example
+	Add two operands	$x + y$
-	Subtract right operand from the left	$x - y$
*	Multiply two operands	$x * y$

/	Divide left operand by the right one	x / y
%	Modulus - remainder of the division of left operand by the right	$x \% y$ (remainder of x/y)

Assignment operators

Assignment operators are used to assign values to variables.

`int a = 5` is a simple assignment operator that assigns the value 5 on the right to the variable a on the left.

Operator	Example	Equivalent to
=	<code>x = 5</code>	<code>x = 5</code>
+=	<code>x += 5</code>	<code>x = x + 5</code>
-=	<code>x -= 5</code>	<code>x = x - 5</code>
*=	<code>x *= 5</code>	<code>x = x * 5</code>
/=	<code>x /= 5</code>	<code>x = x / 5</code>
%=	<code>x %= 5</code>	<code>x = x % 5</code>

Example 4

```
#include<stdio.h>
int main()
{
int a=11;
int b=3;
int remainder = a%b;
printf("\nRemainder: %d", remainder);
return 0;
}
```

Output:

Remainder: 2

Example 5 | Basic Calculator:

```
#include<stdio.h>
int main()
{
    int a=2;
    int b=3;
    int sum=a+b;
    int diff=a-b;
    int product=a*b;
    int division=a/b;
    int mod=a%b;
    printf("a = %d b = %d", a, b);
    printf("\nSum: %d", sum);
    printf("\nDiff: %d", diff);
    printf("\nProduct: %d", product);
    printf("\nDivision: %d", division);
    printf("\nModulus: %d", mod);
return 0;
}
```

Output

a = 2 b = 3

Sum: 5

Diff: -1

Product: 6

Division: 0

Modulus: 2