**Math Series**

# C Programming

UG Sem-3 Major (Kalyani University)

Day - 02

# *Use of printf()*

In C Programming language, the printf() function is used for output. printf() function can take any number of arguments. First argument must be enclosed within the double quotes “hello” and every other argument should be separated by comma ( , ) within the double quotes.

**Syntax:**

printf("format specifier",argument_list);

The format string for output can be %d (integer), %c (character), %s (string), %f (float) %lf (double) and %x (hexadecimal) variable

# *Use of  scanf()*

In C, scanf() is a standard input function used to read formatted data from the standard input stream (stdin), which is usually the keyboard.

- It scans the input according to the specified format specifiers (like %d, %f, %s, etc.) and stores the values into the provided variable addresses.
- The scanf() function is defined in the <stdio.h> header file.

**Syntax:**

```
scanf("format specifier",argument_list);
```

# *Example of  scanf() and printf()*

```c
#include <stdio.h>

int main() {
    int n;

    // Reading an integer input
    scanf("%d", &n);
    printf("%d", n);
    return 0;
}
```

**Output**

```
10 (Enter by user)
10
```

## *Explanation*

`scanf("%d", &n)` reads an integer from the keyboard and stores it in variable `n`. Here, `%d` specifies an integer input, and `&n` gives the memory address where the value is stored.

In this program, the format specifier %d is used to print integers variables using printf. Here, it prints the values of n.

# *Escape characters or backslash characters:*

| | | |
|---|---|---|
| a) | \n | newline |
| b) | \r | carriage return |
| c) | \t | tab |
| d) | \v | vertical tab |
| e) | \b | backspace |
| f) | \f | form feed (page feed) |
| g) | \a | alert (beep) |
| h) | \’ | single quote(‘) |
| i) | \” | double quote(“) |
| j) | \? | Question mark (?) |
| k) | \\ | backslash (\) |

# *OPERATORS AND EXPRESSIONS:*

***Operators :*** *An operator is a Symbol that performs an operation. An operators acts some variables are called operands to get the desired result. Ex : a+b;*
*Where a,b are operands and + is the operator.*

***Types of Operator :***

*1) Arithmetic Operators.*
*2) Relational Operators.*
*3) Logical Operators.*
*4) Assignment Operators.*
*5). Unary Operators.*
*6) Conditional Operators.*
*7) Special Operators.*
*8) Bitwise Operators.*
*9) Shift Operators.*

# *Arithmetic Operators:*

*An arithmetic operator performs mathematical operations such as addition, subtraction and multiplication on numerical values (constants and variables).*

```c
#include <stdio.h>
void main()
{
int a = 9,b = 4, c;
c = a+b;
printf("a+b = %d \n",c);
c = a-b;
printf("a-b = %d \n",c);
c = a*b;
printf("a*b = %d \n",c);
c=a/b;
printf("a/b = %d \n",c);
c=a%b;
printf("Remainder when a divided by b = %d \n",c);
}
```

***Output***

*a+b = 13*

*a-b = 5*

*a*b = 36*

*a/b = 2*

*Remainder when a divided by b=1*

# *Relational Operators:*

*A relational operator checks the relationship between two operands. If the relation is true, it returns 1; if the relation is false, it returns value 0. Operands may be variables, constants or expressions. Relational operators are used in decision making and loops.*

| Operator | Meaning | Example | Return value |
|---|---|---|---|
| < | is less than | 2<9 | 1 |
| < = | is less than or equal to | 2 < = 2 | 1 |
| > | is greater than | 2 > 9 | 0 |
| > = | is greater than or equal to | 3 > = 2 | 1 |
| = = | is equal to | 2 = = 3 | 0 |
| != | is not equal to | 2!=2 | 0 |

# *Logical Operators:*

*These operators are used to combine the results of two or more conditions. An expression containing logical operator returns either 0 or 1 depending upon whether expression results true or false. Logical operators are commonly used in decision making in C programming.*

***Logical AND : If any one condition false the complete condition becomes false.***

***Logical OR : If any one condition true the complete condition becomes true.***

***Logical Not : This operator reverses the value of the expression it operates on i.e, it makes a true expression false and false expression true.***

# Example of Logical Operators:

| Operator | Meaning | Example | Return value |
|---|---|---|---|
| && | Logical AND | (9>2)&&(17>2) | 1 |
| \|\| | Logical OR | (9>2) \|\| (17 = = 7) | 1 |
| ! | Logical NOT | 29!=29 | 0 |

| Op1 | Op1 ! |
|---|---|
| true | false |
| false | true |

**Truth Table**

| Op1 | Op2 | Op1 && Op2 |
|---|---|---|
| true | true | true |
| true | false | false |
| false | true | false |
| false | false | false |

**Truth Table**

| Op1 | Op2 | Op1 // Op2 |
|---|---|---|
| true | true | true |
| true | false | true |
| false | true | true |
| false | false | false |

# *Assignment Operators:*

*Assignment operators are used to assign a value (or) an expression (or) a value of a variable to another variable.*

***Syntax :*** *variable name=expression (or) value (or) variable*
***Example :*** *x=10; y=a+b; z=p;*

***Note:*** *Here, the = symbol does not represent the 'equals to' sign; it signifies the assignment of a value. The 'equals to' comparison in C programming is denoted by ==.*

***Note:*** *int a=b=20; (Wrong);*

```
int a,b;
a=b=5;
```

# Compound Assignment Operators:

*C provides compound assignment operators to assign a value to variable in order to assign a new value to a variable after performing a specified operation.*

| Operator | Example | Meaning |
|---|---|---|
| + = | x + = y | x=x+y |
| - = | x - = y | x=x-y |
| * = | x * = y | x=x*y |
| / = | x / = y | x=x/y |
| % = | x % = y | X=x%y |