1. **Formatted I/O Statements**
   When we accept the data using scanf() function, it is formatted statement.
   It refers to an input data that has been arranged in a particular format.
   Example:
   123 3.14 GUJARAT

scanf();

In the C programming language, scanf is a function that reads formatted data from stdin (i.e, the standard input stream, which is usually the keyboard, unless redirected) and then writes the results into the arguments given.

Syntax:

scanf("Control String", Arg1, Arg2, Arg3… Argn);

Where

Control string is

- format specifier or control string. It contains filed specification which direct the interpretation of input data.
- Control string specify the field format in which the data is to be entered.
- Generally, Control String consist the conversion character % and data type character.

Arg1, Arg2, Arg3… Argn specifies the address location of the data.

The % Format Specifiers

The **%** specifiers that you can use in C are:

Usual variable type:

%c char single character
%d (%i) int signed integer
%e (%E) float or double exponential format
%f float or double signed decimal
%g (%G) float or double use %f or %e as required
%o int unsigned octal value
%p pointer address stored in pointer
%s array of char sequence of characters

%u int unsigned decimal
%x (%X) int unsigned hex value

Example:

```
void main()
{
   int a;
   float b;
   char c;
   clrscr();
   printf("Enter Data for integer, float and character--");
   scanf("%d %f %c",&a,&b,&c);
   printf("\nInteger a=%d",a);
   printf("\nFloat a=%f",b);
   printf("\nCharacter a=%c",c);
   getch();
}
```

printf();
Syntax:
printf("Control String",Arg1,Arg2,Arg3,….Argn);
Where Control String consist the one of the following:

- Characters that will be printed on the screen as they appear
- Format Specification that defines the output format for display its item
- Escape Sequence characters like \n,\t,\b etc…

Arg1, Arg2, Arg3… Argn are the variables or constant which we want to print.

The % Format Specifiers

The **%** specifiers that you can use in C are:

Usual variable type:

%c char single character
%d (%i) int signed integer
%e (%E) float or double exponential format
%f float or double signed decimal
%g (%G) float or double use %f or %e as required

%o int unsigned octal value
%p pointer address stored in pointer
%s array of char sequence of characters
%u int unsigned decimal
%x (%X) int unsigned hex value

Example:

```
void main()
{
    int a;
    float b;
    char c;
    clrscr();
    printf("Enter Data for integer, float and character--");
    scanf("%d %f %c",&a,&b,&c);
    printf("\nInteger a=%d",a);
    printf("\nFloat a=%f",b);
    printf("\nCharacter a=%c",c);
    getch();
}
```

Escape Sequence:

Escape Sequence is also known as Backslash Character(\).
It is composed of two or more characters starting with backslash \. For example: \n represents new line.

Escape Sequence	Meaning

\a	Alarm or Beep
\b	Backspace
\f	Form Feed
\n	New Line
\r	Carriage Return
\t	Tab (Horizontal)
\v	Vertical Tab
\\	Backslash
\'	Single Quote
\"	Double Quote
\?	Question Mark
\nnn	octal number
\xhh	hexadecimal number
\0	Null

Example:

```
#include<stdio.h>
void main()
{
    int number=50;
    printf("You\nare\nlearning\n\'c\' language\n\"Do you
know C language\"");
    getch();
}
```

Output:

```
You
are
learning
```

'c' language
"Do you know C language"

2. Unformatted I/O Statements
Unformatted input statements:
getchar()
This function is used to take input of a single character.
Syntax:
Var name=getchar();
Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
   char c;
   printf("Enter any character:");
   c=getchar();
   printf("You have entered %c",c);
   getch();
}
```

Output:

```
Enter any character: A
You have entered A
```

getch()
getch() function is a function in C programming language which waits for any character input from keyboard.
getche()
This function is used to take input of single character.
This function will automatically move the execution of program to the next line.
When we use getche() user will give only single character.
Var name=getche();
Example:

```
#include<stdio.h>
#include<conio.h>
void main()
```

```
{
   char c;
   printf("Enter any character:");
   c=getche();
   printf("You have entered %c",c);
   getch();
}
```

Output:

```
Enter any character: A
You have entered A
```

gets()

This function is used to take input of a String:

Syntax:

```
gets(String Variable);
#include<stdio.h>
#include<conio.h>
void main()
{
   char c[50];
   printf("Enter any String:");
   gets(c);
   printf("You have entered %s",c);
   getch();
}
```

Output:

```
Enter any string: GUJARAT
You have entered GUJARAT
```

Unformatted output statements:

putchar()

This function is used to take print a single character.

Syntax:

```
putchar(char variable);
```

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
```

```
{
   char c;
   c=getchar();
   putchar(c);
   getch();
}
```

Output:

```
A
A
```

puts()

This function is used to print a String:

Syntax:

```
puts(String Variable);
#include<stdio.h>
#include<conio.h>
void main()
{
   char c[50];
   printf("Enter any String:");
   gets(c);
   puts(c);
   getch();
}
```

Output:

```
Enter any string: GUJARAT
GUJARAT
```

3. Mathematical Functions

- C functions which are used to perform mathematical operations in a program are called Arithmetic functions.
- Functions are given below :

  abs(), floor(), round(), ceil(), sin(), cos(), cosh(), exp(), tan(), tanh(), sinh(), log(), log10(), sqrt(), pow() and trunc().

1. abs():

- ✓ abs ( ) function in C returns the absolute value of an integer. The absolute value of a number is always positive. Only integer values are supported in C.
- ✓ “stdlib.h” header file supports abs( ) function in C language. Syntax for abs( ) function in C is given below.

Syntax : int abs (int n);

Example:
```
#include <stdio.h>
#include <stdlib.h>
void  main()
{
  int m = abs(200);     // m is assigned to 200
  int n = abs(-400);    // n is assigned to -400

  printf("Absolute value of m = %d\n", m);
  printf("Absolute value of n = %d \n",n);
  getch();
}
```
Output:
```
Absolute value of m = 200
  Absolute value of n = 400
```

2. floor():

- ✓ floor( ) function in C returns the nearest integer value which is less than or equal to the floating point argument passed to this function.
- ✓ ”math.h” header file supports floor( ) function in C language. Syntax for floor( ) function in C is given below.

Syntax: double floor (double x);

Example:
```
#include <stdio.h>
```

```
#include <math.h>
 Void main()
{
      float i=5.1, j=5.9, k=-5.4, l=-6.9;
      printf("floor of  %f is  %f\n", i, floor(i));
      printf("floor of  %f is  %f\n", j, floor(j));
      printf("floor of  %f is  %f\n", k, floor(k));
      printf("floor of  %f is  %f\n", l, floor(l));
      getch();
}
```

Output:

```
floor of 5.100000 is 5.000000
floor of 5.900000 is 5.000000
floor of -5.400000 is -6.000000
floor of -6.900000 is -7.000000
```

3. **ceil():**

- ✓ ceil( ) function in C returns nearest integer value which is greater than or equal to the argument passed to this function.
- ✓ "math.h" header file supports ceil( ) function in C language. Syntax for ceil( ) function in C is given below.

Syntax: double ceil (double x);

Example:

```
#include <stdio.h>
#include <math.h>
void main()
{
  float i=5.4, j=5.6;
  printf("ceil of  %f is  %f\n", i, ceil(i));
  printf("ceil of  %f is  %f\n", j, ceil(j));
  getch();
}
```

Output:

```
ceil of 5.400000 is 6.000000
ceil of 5.600000 is 6.000000
```

4. sin(),cos(),tan(),sinh(),cosh(),tanh(),exp(),log(),log10():

- ✓ sin( ), cos( ) and tan( ) functions in C are used to calculate sine, cosine and tangent values.
- ✓ sinh( ), cosh( ) and tanh( ) functions are used to calculate hyperbolic sine, cosine and tangent values.
- ✓ exp( ) function is used to calculate the exponential “e” to the xth power. log( ) function is used to calculates natural logarithm and log10( ) function is used to calculates base 10 logarithm.
- ✓ ”math.h” header file supports all these functions in C language.

Example:

```
#include <stdio.h>
#include <math.h>
void main()
{
        float i = 0.314;
        float j = 0.25;
        float k = 6.25;

        float sin_value = sin(i);
        float cos_value = cos(i);
        float tan_value = tan(i);
        float sinh_value = sinh(j);
        float cosh_value = cosh(j);
        float tanh_value = tanh(j);
        float log_value = log(k);
        float log10_value = log10(k);
        float exp_value = exp(k);

printf("The value of sin(%f) : %f \n", i, sin_value);
```

```
    printf("The value of cos(%f) : %f \n", i, cos_value);
    printf("The value of tan(%f) : %f \n", i, tan_value);
    printf("The value of sinh(%f) : %f \n", j,
    sinh_value);
    printf("The value of cosh(%f) : %f \n", j,
    cosh_value);
    printf("The value of tanh(%f) : %f \n", j,
    tanh_value);
    printf("The value of log(%f) : %f \n", k, log_value);
    printf("The value of log10(%f) :
    %f\n",k,log10_value);
    printf("The value of exp(%f) : %f \n",k, exp_value);

     getch();
    }
```

Output:

```
The value of sin(0.314000) : 0.308866
The value of cos(0.314000) : 0.951106
The value of tan(0.314000) : 0.324744
The value of sinh(0.250000) : 0.252612
The value of cosh(0.250000) : 1.031413
The value of tanh(0.250000) : 0.244919
The value of log(6.250000) : 1.832582
The value of log10(6.250000) : 0.795880
The value of exp(6.250000) : 518.012817
```

6. sqrt() :

- ✓ sqrt( ) function in C is used to find the square root of the given number.
- ✓ "math.h" header file supports sqrt( ) function in C language. Syntax for sqrt( ) function in C is given below.

Syntax: double sqrt (double x);

Example:

```
#include <stdio.h>
#include <math.h>
void main()
{
   printf ("sqrt of 16 = %f\n", sqrt (16) );
   printf ("sqrt of  2 = %f\n", sqrt (2) );
   getch();
}
```

Output:

```
sqrt of 16 = 4.000000
sqrt of 2 = 1.414214
```

7. pow():

- ✓ pow( ) function in C is used to find the power of the given number.
- ✓ "math.h" header file supports pow( ) function in C language. Syntax for pow( ) function in C is given below.

Syntax: double pow (double base, double exponent);

Example:

```
#include <stdio.h>
#include <math.h>
void main()
{
        printf ("2 power 4 = %f\n", pow (2.0, 4.0) );
        printf ("5 power 3 = %f\n", pow (5, 3) );
        getch();
}
```

Output:

```
2 power 4 = 16.000000
5 power 3 = 125.000000
```

4. String Functions

There are numerous functions defined in "string.h" header file. Few commonly used string handling functions are discussed below:

Function	Use
strlen()	Calculates the length of string
strcpy()	Copies a string to another string
strcat()	Concatenates(joins) two strings
strcmp()	Compares two string
strlwr()	Converts string to lowercase
strupr()	Converts string to uppercase

strlen()
strlen() function calculates the length of string.
Syntax:
temp_variable = strlen(string_name);

here string_name is the string which we want length

Function strlen() returns the value of type integer.
Example:

```
#include <stdio.h>
#include <string.h>
void main(){
    char a[20]="Program";
    char b[20]={'P','r','o','g','r','a','m','\0'};
    char c[20];
    printf("Enter string: ");
    gets(c);
    printf("Length of string a=%d \n",strlen(a));
    //calculates the length of string before null charcter.
    printf("Length of string b=%d \n",strlen(b));
    printf("Length of string c=%d \n",strlen(c));
  }
```

Output:

```
Enter string: String
```

```
Length of string a=7
Length of string b=7
Length of string c=6
```

strcpy()

strcpy() copies the content of one string to the content of another string.

Syntax:

strcpy(destination,source);

Here, source and destination are both the name of the string. This statement, copies the content of string source to the content of string destination.

Example:

```
#include <stdio.h>
#include <string.h>
void main()
{
   char a[10],b[10];
   printf("Enter string: ");
   gets(a);
   strcpy(b,a);   //Content of string a is copied to string b.
   printf("Copied string: ");
   puts(b);
 }
```

Output:

```
Enter string: C Language
Copied string: C Language
```

strcat()

strcat() concatenates(joins) two strings.

It takes two arguments, i.e, two strings and resultant string is stored in the first string specified in the argument.

Syntax:

strcat(first_string,second_string);

Example:

```
void main ()
{
   char src[50], dest[50];

   strcpy(src,  "This is source");
   strcpy(dest, "This is destination");

   strcat(dest, src);

   printf("Final destination string : |%s|", dest);

  }
```

Output:

```
Final destination string : |This is destinationThis is source|
```

strcmp()

strcmp() compares two string and returns value 0, if the two strings are equal.

Function strcmp() takes two arguments, i.e, name of two string to compare

Syntax:

```
temp_varaible=strcmp(string1,string2);
```

Example:

```
#include <stdio.h>
#include <string.h>
void main()
{
  char str1[30],str2[30];
  printf("Enter first string: ");
  gets(str1);
  printf("Enter second string: ");
  gets(str2);
  if(strcmp(str1,str2)==0)
      printf("Both strings are equal");
  else
      printf("Strings are unequal");
}
```

Output:
```
Enter first string: Apple
Enter second string: Apple
Both strings are equal.
```

strlwr()

strlwr() function converts all the uppercase characters in that string to lowercase characters.

Syntax:
```
strlwr(string_name);
```
Example:
```
#include <stdio.h>
#include <string.h>
void main()
{
   char str1[]="LOWer Case";
   puts(strlwr(str1));   //converts to lowercase and displays
it.
}
```
Output:
```
lower case
```
strupr()

strupr() function converts all the lowercase characters in that string to uppercase characters.

Syntax:
```
strupr(string_name);
```

Example:
```
#include <stdio.h>
#include <string.h>
void main()
{
   char str1[]="Upper Case";
   puts(strlwr(str1));   //converts to uppercase and displays
it.
}
```

Output:

```
UPPER CASE
```

5. Conversion Functions

- The header file stdlib.h includes functions, used for different conversions.
- When we get input of a different type other than the type of variable in which the value is being stored, it warrants the need to convert that type into another type.
- These conversion functions take an argument of a type and return it after converting into another type. These functions and their description are given in the table below.

Typecast function	Description
atof()	atof( ) function converts string to float
atoi()	atoi( ) function converts string to int
atol()	atol( ) function converts string to long
itoa()	itoa( ) function converts int to string
ltoa()	ltoa( ) function converts long to string

1. **atof():**

- atof() function in C language converts string data type to float data type. Syntax for atof() function is given below.

```
double atof (const char* string);
```

```
#include <stdio.h>
#include <stdlib.h>
```

```
void main()
{
    char a[10] = "3.14";
    float pi = atof(a);
    printf("Value of pi = %f\n", pi);
}
```

Output:

```
Value of pi = 3.140000
```

2. atoi():

atoi() function in C language converts string data type to int data type. Syntax for atoi() function is given below.

int atoi (const char * str);

```
#include <stdio.h>
#include <stdlib.h>

void  main()
{
    char a[10] = "100";
    int value = atoi(a);
    printf("Value = %d\n", value);
}
```

Output:

```
Value = 100
```

3. atol():

atol() function in C language converts string data type to long data type. Syntax for atol() function is given below.

long int atol (const char * str);

```
#include <stdio.h>
#include <stdlib.h>

void main()
{
    char a[20] = "100000000000";
    long value = atol(a);
    printf("Value = %ld\n", value);
}
```

Output:

```
Value = 100000000000
```

4. **itoa():**

   itoa () function in C language converts int data type to string data type. Syntax for this function is given below.

char * itoa (int value, char * str, int base);

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void main()
{
    int a=54325;
    char buffer[20];
    itoa(a,buffer,2);   // here 2 means binary
    printf("Binary value = %s\n", buffer);

    itoa(a,buffer,10);   // here 10 means decimal
    printf("Decimal value = %s\n", buffer);

    itoa(a,buffer,16);   // here 16 means Hexadecimal
    printf("Hexadecimal value = %s\n", buffer);
    }
```

itoa () function Output:

```
Binary value = 1101010000110101
Decimal value = 54325
Hexadecimal value = D435
```

5. ltoa():

ltoa() function in C language converts long data type to string data type. Syntax for ltoa() function is given below.

char *ltoa(long N, char *str, int base);

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main()
{
   long  a=10000000;
   char buffer[50];
   ltoa(a,buffer,2);   // here 2 means binary
   printf("Binary value = %s\n", buffer);

   ltoa(a,buffer,10);   // here 10 means decimal
   printf("Decimal value = %s\n", buffer);

   ltoa(a,buffer,16);   // here 16 means Hexadecimal
   printf("Hexadecimal value = %s\n", buffer);
   return 0;
}
```

Output:

```
Binary value = 100110001001011010000000
Decimal value = 10000000
Hexadecimal value = 989680
```

FOP