Chapter 4:
Input Output in c

C printf and scanf functions:printf() and scanf() functions are inbuilt library functions in C programming language which are available in C library by default. These functions are declared and related macros are defined in “stdio.h” which is a header file in C language.

- We have to include “stdio.h” file as shown in below C program to make use of these printf() and scanf() library functions in C language.

1. PRINTF() FUNCTION IN C LANGUAGE:

- In C programming language, printf() function is used to print the (“character, string, float, integer, octal and hexadecimal values”) onto the output screen.
- We use printf() function with %d format specifier to display the value of an integer variable.
- Similarly %c is used to display character, %f for float variable, %s for string variable, %lf for double and %x for hexadecimal variable.
- To generate a newline,we use “\n” in C printf() statement.

Note:

- C language is case sensitive. For example, printf() and scanf() are different from Printf() and Scanf(). All characters in printf() and scanf() functions must be in lower case.

EXAMPLE PROGRAM FOR C PRINTF() FUNCTION:

```c
#include <stdio.h>
int main()
{
   char ch = 'A';
   char str[20] = "fresh2refresh.com";
   float flt = 10.234;
   int no = 150;
   double dbl = 20.123456;
   printf("Character is %c \n", ch);
   printf("String is %s \n" , str);
   printf("Float value is %f \n", flt);
   printf("Integer value is %d\n" , no);
   printf("Double value is %lf \n", dbl);
   printf("Octal value is %o \n", no);
   printf("Hexadecimal value is %x \n", no);
   return 0;
}
```

OUTPUT:

```
Character is A
String is fresh2refresh.com
Float value is 10.234000
Integer value is 150
Double value is 20.123456
Octal value is 226
Hexadecimal value is 96
```

You can see the output with the same data which are placed within the double quotes of printf statement in the program except.

- %d got replaced by value of an integer variable (no),
- %c got replaced by value of a character variable (ch),
- %f got replaced by value of a float variable (flt),
- %lf got replaced by value of a double variable (dbl),
- %s got replaced by value of a string variable (str),
- %o got replaced by a octal value corresponding to integer variable (no),
- %x got replaced by a hexadecimal value corresponding to integer variable
- \n got replaced by a newline.

2. SCANF FUNCTION IN C LANGUAGE:

- In C programming language, scanf() function is used to read character, string, numeric data from keyboard
- Consider below example program where user enters a character. This value is assigned to the variable “ch” and then displayed.
- Then, user enters a string and this value is assigned to the variable “str” and then displayed.

EXAMPLE PROGRAM FOR C PRINTF AND SCANF FUNCTIONS IN C PROGRAMMING LANGUAGE:

```c
#include <stdio.h>
int main()
{
   char ch;
   char str[100];
   printf("Enter any character \n");
   scanf("%c", &ch);
   printf("Entered character is %c \n", ch);
   printf("Enter any string ( upto 100 character ) \n");
   scanf("%s", &str);
   printf("Entered string is %s \n", str);
}
```

OUTPUT :

```
Enter any character
a
Entered character is a
Enter any string ( upto 100 character )
hai
Entered string is hai
```

- The format specifier %d is used in scanf() statement. So that, the value entered is received as an integer and %s for string.
- Ampersand is used before variable name “ch” in scanf() statement as &ch.
- It is just like in a pointer which is used to point to the variable. For more information about how pointer works, please click here.

KEY POINTS TO REMEMBER IN C PRINTF() AND SCANF():

1. printf() is used to display the output and scanf() is used to read the inputs.
2. printf() and scanf() functions are declared in “stdio.h” header file in C library.
3. All syntax in C language including printf() and scanf() functions are case sensitive.

C gets() and puts() functions

The gets() and puts() are declared in the header file stdio.h. Both the functions are involved in the input/output operations of the strings.

C gets() function

The gets() function enables the user to enter some characters followed by the enter key. All the characters entered by the user get stored in a character array. The null character is added to the array to make it a string. The gets() allows the user to enter the space-separated strings. It returns the string entered by the user.

Declaration

```
char[] gets(char[]);
```

Reading string using gets()

```
#include<stdio.h>
void main ()
{
   char s[30];
   printf("Enter the string? ");
   gets(s);
   printf("You entered %s",s);
}
```

Output

```
Enter the string?
javatpoint is the best
You entered javatpoint is the best
```

The gets() function is risky to use since it doesn't perform any array bound checking and keep reading the characters until the new line (enter) is encountered. It suffers from buffer overflow, which can be avoided by using fgets(). The fgets() makes sure that not more than the maximum limit of characters are read. Consider the following example.

```
#include<stdio.h>
void main()
{
  char str[20];
  printf("Enter the string? ");
  fgets(str, 20, stdin);
  printf("%s", str);
}
```

Output

```
Enter the string? javatpoint is the best website
javatpoint is the b
```

C puts() function

The puts() function is very much similar to printf() function. The puts() function is used to print the string on the console which is previously read by using gets() or scanf() function. The puts() function returns an integer value representing the number of characters being printed on the console. Since, it prints an additional newline character with the string, which moves the cursor to the new line on the console, the integer value returned by puts() will always be equal to the number of characters present in the string plus 1.

Declaration

```
int puts(char[])
```

Let's see an example to read a string using gets() and print it on the console using puts().

```
#include<stdio.h>
#include <string.h>
int main(){
char name[50];
printf("Enter your name: ");
gets(name); //reads string from user
printf("Your name is: ");
puts(name);  //displays string
return 0;
}
```

Output:

```
Enter your name: Sonoo Jaiswal
Your name is: Sonoo Jaiswal
```

C library function - putchar()

The C library function **int putchar(int char)** writes a character (an unsigned char) specified by the argument char to stdout.

Declaration

Following is the declaration for putchar() function.

```
int putchar(int char)
```

Parameters

char − This is the character to be written. This is passed as its int promotion.

Return Value

This function returns the character written as an unsigned char cast to an int or EOF on error.

Example

The following example shows the usage of putchar() function.

```
#include <stdio.h>
int main () {
   char ch;

   for(ch = 'A' ; ch <= 'Z' ; ch++) {
      putchar(ch);
   }

   return(0);}
```

Let us compile and run the above program that will produce the following result −

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

C library function - getchar()

Description

The C library function **int getchar(void)** gets a character (an unsigned char) from stdin. This is equivalent to **getc** with stdin as its argument.

Declaration

Following is the declaration for getchar() function.

```
int getchar(void)
```

Parameters

- **NA**

Return Value

This function returns the character read as an unsigned char cast to an int or EOF on end of file or error.

Example

The following example shows the usage of getchar() function.

```c
#include <stdio.h>
int main () {
   char c;

   printf("Enter character: ");
   c = getchar();

   printf("Character entered: ");
   putchar(c);

   return(0);}
```

Let us compile and run the above program that will produce the following result −

```
Enter character: a
Character entered: a
```