## KCA102- Lecture
## Input and Output

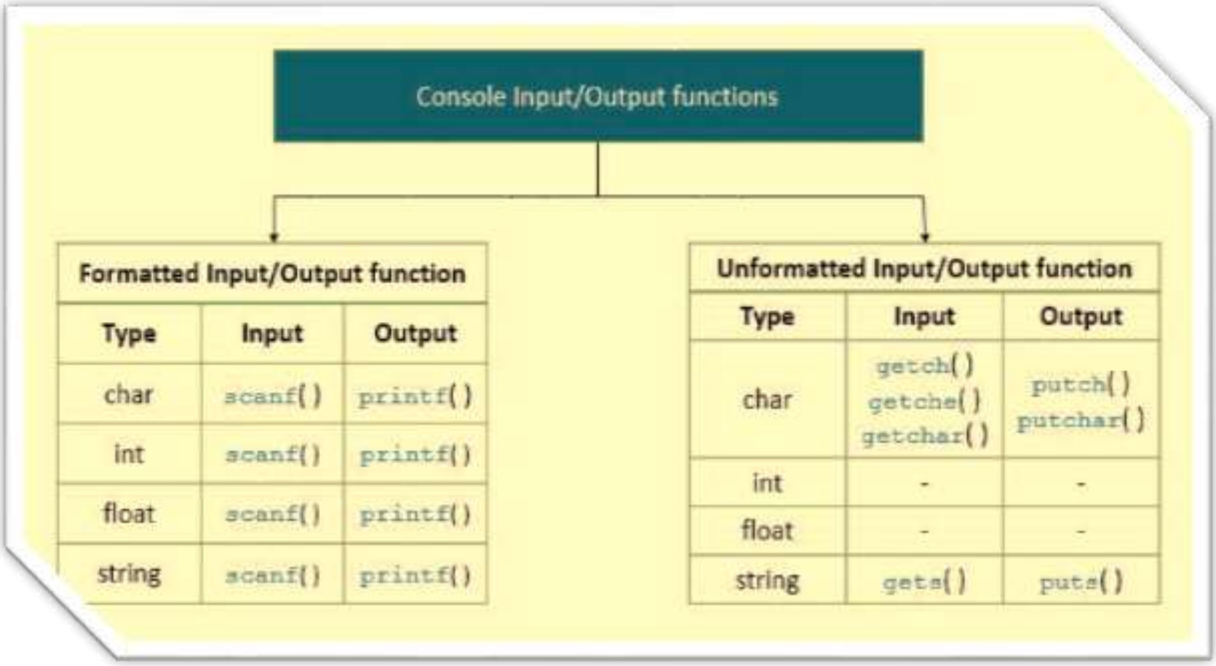
Input Output in C:

1. **Standard /Console I/O (standard Input-Output devices)**
2. **File I/O (A file store data on disk, when your program reading or writing from or to the file called File I/O)**

**Console Input-Output functions:** Console simply means screen and keyboard. There are two types of a console I/O functions:

- **Formatted input-output function** Standard Library function
- **Unformatted input-output function**

The major difference is that formatted function allows us to format the input from the keyboard and the output to be displayed on the screen.



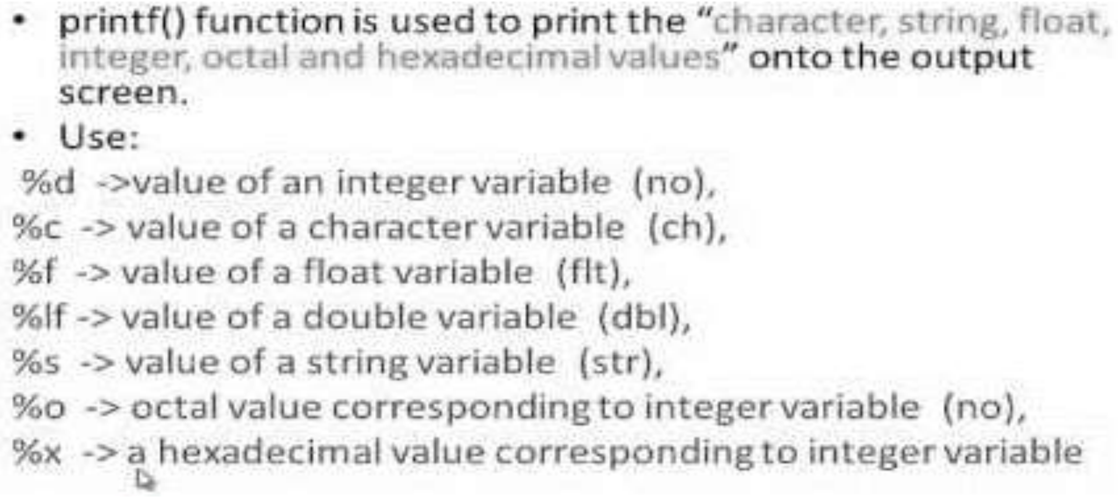
Console Input/Output functions

| Formatted Input/Output function | | |
|---|---|---|
| Type | Input | Output |
| char | scanf() | printf() |
| int | scanf() | printf() |
| float | scanf() | printf() |
| string | scanf() | printf() |

| Unformatted Input/Output function | | |
|---|---|---|
| Type | Input | Output |
| char | getch()<br>getche()<br>getchar() | putch()<br>putchar() |
| int | - | - |
| float | - | - |
| string | gets() | puts() |

### Formatted Output

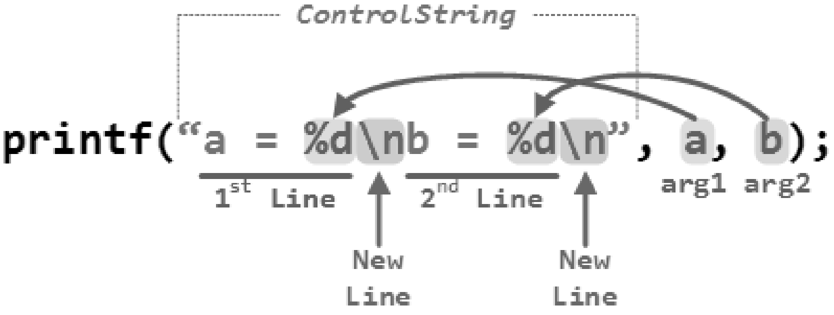
printf() is the standard library function that is used for precise output formatting. It describes the output format which consists of conversion specifiers, precisions, flags, field widths and literal characters.



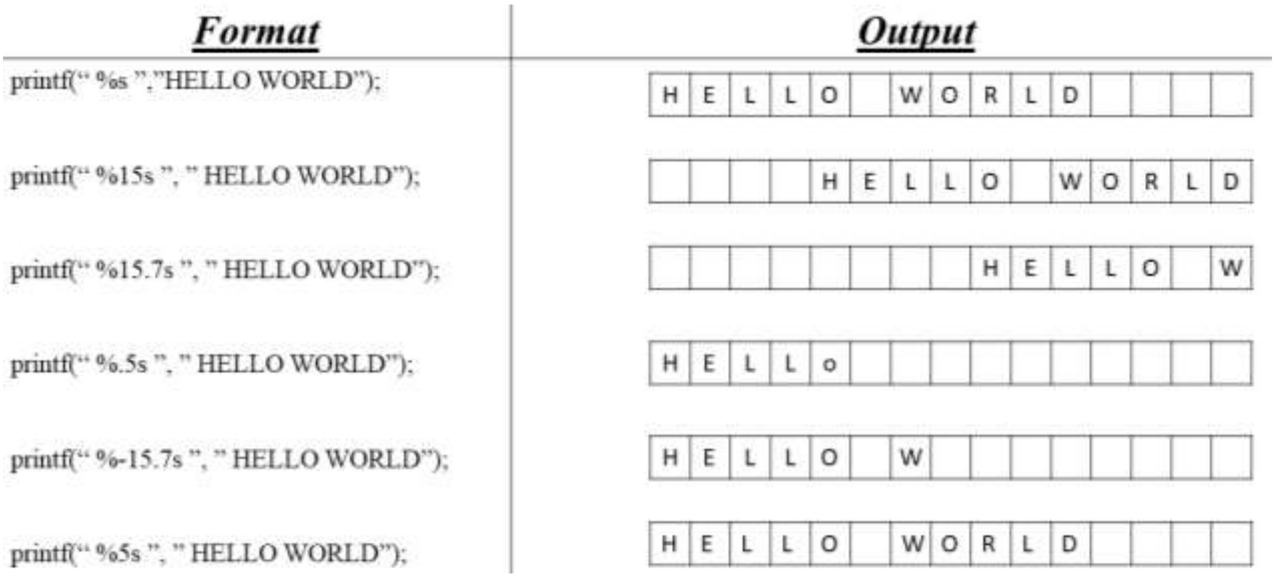
- printf() function is used to print the "character, string, float, integer, octal and hexadecimal values" onto the output screen.
- Use:

%d ->value of an integer variable (no),
%c -> value of a character variable (ch),
%f -> value of a float variable (flt),
%lf -> value of a double variable (dbl),
%s -> value of a string variable (str),
%o -> octal value corresponding to integer variable (no),
%x -> a hexadecimal value corresponding to integer variable

**Syntax: printf(format-control-string, other-arguments );**



```
printf("a = %d\nb = %d\n", a, b);
```



| Format | Output | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| printf(“ %s ”,”HELLO WORLD”); | H | E | L | L | O | | W | O | R | L | D | | | | |
| printf(“ %15s ”, ” HELLO WORLD”); | | | | | H | E | L | L | O | | W | O | R | L | D |
| printf(“ %15.7s ”, ” HELLO WORLD”); | | | | | | | | | H | E | L | L | O | | W |
| printf(“ %.5s ”, ” HELLO WORLD”); | H | E | L | L | o | | | | | | | | | | |
| printf(“ %-15.7s ”, ” HELLO WORLD”); | H | E | L | L | O | | W | | | | | | | | |
| printf(“ %5s ”, ” HELLO WORLD”); | H | E | L | L | O | | W | O | R | L | D | | | | |

**Example**

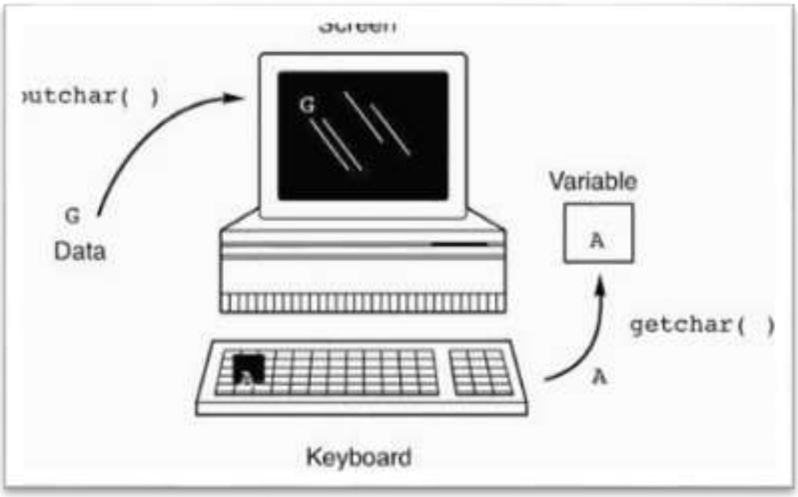
```
void main()
{ int num = 31;
  printf("%o\n", num);
}
void main()
{  int year = 2020, currency = 2020;
float num = 29.99;
printf("%d\n", year);
printf("%i\n", currency);
printf("%f\n", num);
printf("%e\n", num);
   }
```

**Formatted Input**

scanf() is a standard library function used for formatted input from standard input.

- Need to Include stdio.h file in program pre-processor
- reads and converts characters from the standards input depending on the format specification string and stores the input in memory locations represented by the other arguments (num1, num2,….).
  **Syntax:** scanf (format, num1, num2,……);
  **Example:** scanf(“%d”, &Roll No);

**Unformatted I/O : Standard Library function used to read single character**



**getchar()**

- **buffered single character input function used read the from standard input device**
- **character input is echo**

The function getchar() reads the character from the standard input while getc() reads from the input stream/file.

int getchar(void);

**Example**

```
#include <stdio.h>
               void main()
               { char firstInit, lastInit
               printf("Enetr your Name first and last two initials?\n");
               firstInit = getchar();
               getchar(); // Discards the newline
               lastInit = getchar();
               getchar(); //
               }

#include <stdio.h>
               int main() {
                 char val;
                 val = getchar();
                 printf("Enter the character : \n");
                 printf("Entered character : %c", val);
                 return 0;
               }
```

**getch(),**

- **un-buffered (not require carriage return (enter key) to terminate the reading)**
- **single character input function used read the from standard input device**
- **character input is not echo (will not be visible on screen**

The function getch() is a non-standard function. It is declared in “conio.h” header file. It is not a part of C standard library. It immediately returns the entered character without even waiting for the enter key. Here is the syntax of getch() in C language,

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

```
   val = getch();
   printf("Enter the character : ");
   printf("Entered character : %c", val);
   return 0;
}
```

**getche()**

- **un-buffered single character input function used read the from standard input device**
- **character input is echo**

The getche() function is also a non-standard function and declared in “conio.h” header file. It reads a single character from the keyboard and returns it immediately without even waiting for enter key. Here is the syntax of getche() in C language,

int getche(void);

```
#include <stdio.h>
#include<conio.h>
int main()
{
   char val;
   val = getche();
   printf("Enter the character : ");
   printf("Entered character : %c", val);
   return 0;
}
```

**putch(): This function** displays any alphanumeric characters to the standard output device. It displays only one character at a time.

**Putch(char);**

**putchar():** putchar() function is used to write a character on standard output/screen. In a C program, we can use putchar() function

putchar(char);

```
#include <stdio.h>
 int main(void)
 {
   char ch ;
   printf("Input a character:");
        ch = getch();
     putch(ch);
   }
```

**Refernces**

1. Hanly J. R. and Koffman E. B.,“Problem Solving and Program Design in C”, Pearson Education.
2. Schildt H., “C- The Complete Reference”, McGraw-Hill.
3. Kanetkar Y., “Let Us C”, BPB Publications.
4. Gottfried B., “Schaum’s Outlines- Programming in C”, McGraw-Hill Publications.
5. Kochan S.G., “Programming in C”, Addison-Wesley.
6. Dey P. and Ghosh M., “Computer Fundamentals and Programming in C”, Oxford University Press.
7. Goyal K. K., Sharma M. K. and Thapliyal M. P. “Concept of Computer and C Programming”, University Science Press.
8. Goyal K. K. and Pandey H.M., Trouble Free C”, University Science Press