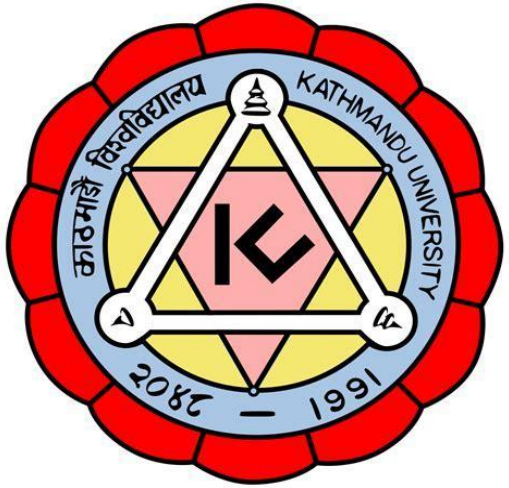
KATHMANDU UNIVERSITY SCHOOL OF MANAGEMENT

BBIS

COM 102 : 3 Credit Hours



5. Writing a Program in C

Outlines

- Simple Program
- Input Statement
- Output Statement
- Features of stdio.h

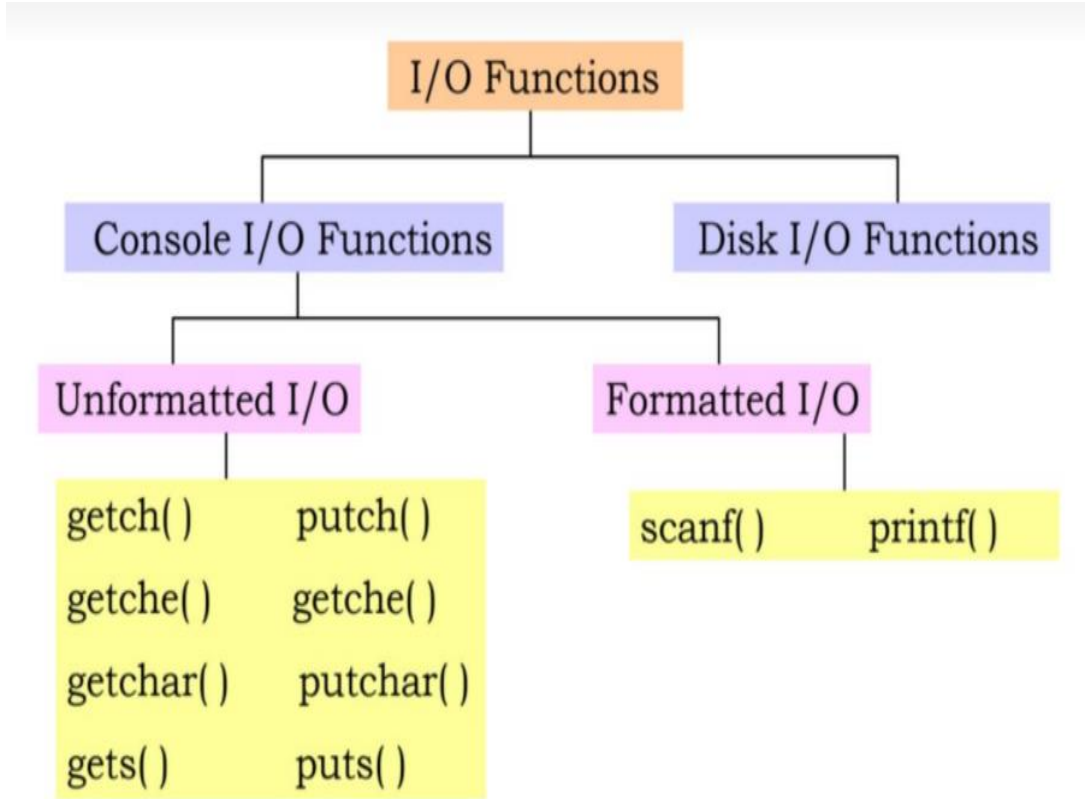
Data input and Output

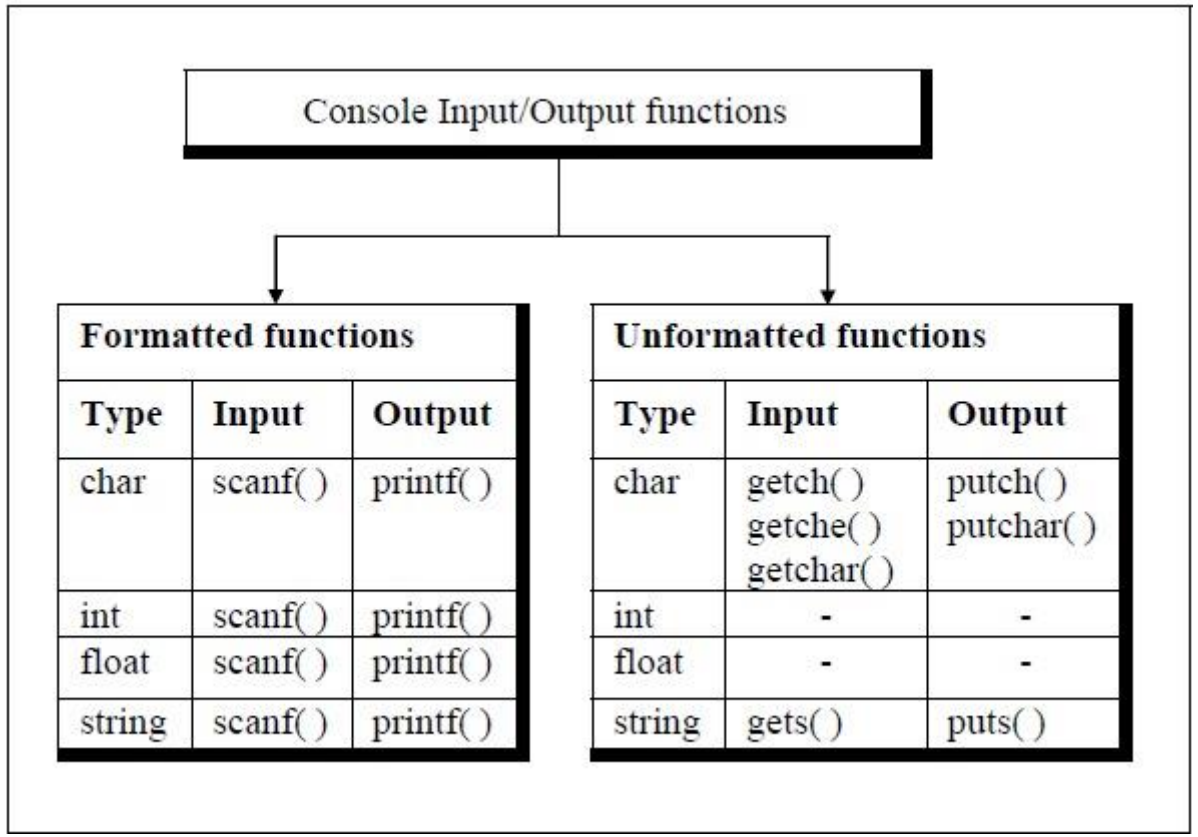
- How to display information on-screen with the printf(), putchar() and puts() library functions?
- How to format the information that is displayed on-screen?
- How to read data from the keyboard with the scanf() library function?

- A program without any input or output has no meaning.
  - Input --> Process --> Output

Displaying Information On-Screen

- We want most of ours programs to display information on-screen.
- The two most frequently used ways to do this are with C's library functions i.e, printf() and puts().
- C programming language provides many built-in functions to read any given input and to display data on screen when there is a need to output the result.
- All these built-in functions are present in C header files, we will also specify the name of header files in which a particular function is defined while discussing about it.
- The stdio.h or standard input output library in C that has methods for input and output.





Formatted functions

Type	Input	Output
char	scanf( )	printf( )
int	scanf( )	printf( )
float	scanf( )	printf( )
string	scanf( )	printf( )

Unformatted functions

Type	Input	Output
char	getch( ) getche( ) getchar( )	putch( ) putchar( )
int	-	-
float	-	-
string	gets( )	puts( )

Figure 1: Types of I/O functions

Formatted Input and Output Functions

- Formatted input and output functions allow the input to be read from the keyboard or the output to be displayed on screen.
- These i/o can be formatted according to our requirements.
  - Input function: e.g. scanf()
  - Output function: e.g. printf()

A Simple Program for Input and Output

```
#include<stdio.h>
void main()
{
// defining a variable
int i;
//displaying message on the screen asking the user to input a value
printf("Please enter a value...");
//reading the value entered by the user
scanf("%d", &i);
//displaying the number as output
printf( "\nYou entered: %d", i);
}
```

…

- We have earlier discussed the purpose of %d inside the scanf() or printf() functions.
- It is known as format string and this informs the scanf() function, what type of input to expect and
- in printf() it is used to give a heads up to the compiler, what type of output to expect.

Format String	Meaning
%d	Scan or print an integer as signed decimal number
%f	Scan or print a floating point number
%c	To scan or print a character
%s	To scan or print a character string. The scanning ends at whitespace.

..

- can also limit the number of digits or characters that can be input or output,
  - by adding a number with the format string specifier, like "%1d" or "%3s",
  - the first one means a single numeric digit and the second one means 3 characters,
  - hence if you try to input 42, while scanf() has "%1d",it will take only 4 as input. Same is the case for output.

Consider the following type of data:

- 50, 13.45, Ram
- int, float, char variables

This is possible using the scanf() function.

- scanf stands for scan formatted.
- The built-in function scanf() can be used to enter input data into the computer from a standard input device.

Syntax: scanf()

The general form of scanf is,

scanf("control string" , &arg1, &arg2,..... &argn);

– control string format in which data is to be entered.

– arg1,arg2... location where the data is stored preceded by ampersand (&)

The control string consists of individual groups of data formats, with one group for each input data item.

- Each data format must begin with a percentage sign.

General form of control string:

[whitespace character][ordinary character]%[field width] conversion character

  – Whitespace characters [optional]
  – Ordinary characters [Optional]
  – Field width [Optional]

Input/output program: whitespace character

```
// scanf simple example
#include<stdio.h>
void main()
{
    int n1;
    char ch;
    printf("Enter a number: ");
    scanf("%d%c",&n1,&ch);
    printf("Enter a character: ");
    scanf("%c",&ch);
    printf("\n Number: %d \t Character: %c",n1,ch);
}
```

Input/Output Program:Ordinary characters

```
// Ordinary characters
#include<stdio.h>
void main()
{
        int day, year, month;
        printf("enter day month year in DD-MM-YYYY format");
        scanf("%d-%d-%d",&day,&month,&year);
        printf("The entered date is: %d - %d - %d",day,month,year );
}
```

Input/Output Program: Field width

```
//Field width example
#include<stdio.h>
void main()
{
        int d;
        printf("Enter Max 5 numbers");
        scanf("%5d",&d);
        printf("Entered Numbers: %d",d);
}
```

Input/output program: String

```
//Input String
#include<stdio.h>
void main()
{
char string[10];
printf("Enter Your Name");
scanf("%s",string);
printf("Your Name is %s", string);
}
```

Classwork ...

- WAP to ask the birthdate of your friend in the format YYYY/MM/DD and display it in the screen

More on Scanf()

- %[character]
  - only characters specified within the brackets are allowed in the input string.
- %[^character]
  - the character specified after the caret are not allowed

```
#include<stdio.h>
void main()
{
char string[10];
printf("Enter Your Name in uppercase:");
scanf("%[A-Z]",string);
printf("Your Name is %s",string);
}
```

```
#include<stdio.h>
void main()
{
char string[10];
printf("Enter Your Name:");
scanf("%[^\n]",string);
printf("Your Name is %s",string);
}
```

Formatted Output: printf() …

- Formatted output refers to the output of data that has been arranged in a particular format.
- printf() is a built in function which is used to output data from the computer onto a standard device i.e. screen
- General form:
  - printf(“control string”, arg1,arg2,.....,arg n)

The control string consists of four types of items:

– characters that will be printed on the screen as they appear.
– format specifications that define the output format for display of each item
– escape sequence characters such as \n, \t etc.
– any combination of characters, format specifications and escape sequences.

printf() ...

The control string has the form: **%[flag] [field width][.precision] conversion character**

Flags [optional]

“ – “ indicates data item to be left-justified

“+” indicates a positive or negative sign to precede

“0” indicates leading 0’s to appear instead of leading blanks

Field width[optional]

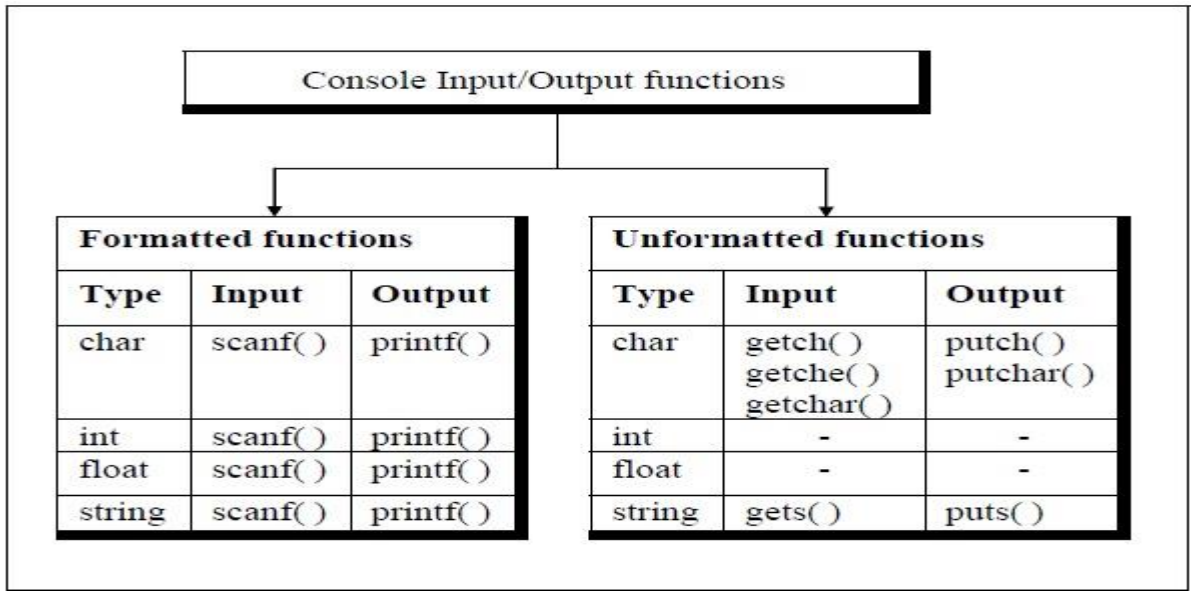
– Same as before

Precision [optional]

– The operation of precision field depends on the type of conversion. It must start with a period (.).

Unformatted i/o Functions

- Doesn’t allow user to read or display data in desired format.
- These library functions basically deals with a single character or a string of characters.
- The functions getchar(), putchar(), gets(),puts(),getch(),getche(),putch() are considered as unformatted functions.



Formatted functions		
Type	**Input**	**Output**
char	scanf( )	printf( )
int	scanf( )	printf( )
float	scanf( )	printf( )
string	scanf( )	printf( )

Unformatted functions		
Type	**Input**	**Output**
char	getch( ) getche( ) getchar( )	putch( ) putchar( )
int	-	-
float	-	-
string	gets( )	puts( )

getchar()

– Reads a character from a standard input device.

– It takes the form:

character_variable = getchar();

– character_variable is a valid C char type variable.

– Since getchar() is used on the right hand side of an assignment statement, the character value of getchar is in turn assigned to the variable name on the left.

e.g. char name;

```
name = getchar();
```

When this statement is encountered, the computer waits until a key is pressed and then assigns this character to character_variable.

putchar()

- Displays a character to the standard output device.
- It takes the form:
  - putchar(character_variable);
- where character_variable is a char type variable containing a character

Program: getchar() ... putchar()

```
#include<stdio.h>
void main()
{
        char gender;
        printf("Enter gender M or F:" );
        gender=getchar();
        printf("Your Gender is: ");
        putchar(gender);
}
```

Classwork…

- Write a program to ask the initial of your friend’s name and print it in the screen using unformatted i/o functions.

getch() and getche()

– Reads single character the instant it is typed without waiting for the enter key to be hit.

– getch() doesn’t print the character entered.

– getche() displays the character when entered.

– **General form**

```
character_variable=getch();
character_variable=getche();
```

– In both functions, the character typed is assigned to the char type variable character_variable.

putch()

– The function putch() prints a character onto the screen.

– General form

putch(character_variable);

– character variable is a char type.

NOTE: These three functions: getch(), getche() and putch() are defined under the standard library functions conio.h header file which is mostly used by MS-DOS compilers like Turbo C.

Program: getch()... getche() ... putch()

```
#include<stdio.h>
#include<conio.h>
void main()
{
	char ch1, ch2;

	printf("Enter 1st character: ");
	ch1=getch();

	printf("\n Enter 2nd character");
	ch2=getche();

	printf("\n first character: ");
	putch(ch1);

	printf("\n Second character: ");
	putch(ch2);
}
```

gets() ... puts()

gets()

– Used to read string of text, containing whitespaces, until a new line character is encountered.

– General form

```
gets(string_variable);
```

puts()

– Used to display the string onto the terminal

– **General form:**

```
puts(string_variable);
```

Program: gets() ... puts()

```
#include<stdio.h>
void main() {
    char name[20];
    printf("Enter your name:");
    gets(name);
    printf("Your Name is: ");
    puts(name);
}
```

Difference between scanf() and gets()

The main difference between these two functions is that scanf() stops reading characters when it encounters a space, but gets() reads space as character too.

Classwork

Write a program to get a sentence as an user input and print it in the screen using gets() and puts() functions.

Any Queries???