**Science College for Women/ Second Class/System programming/ Lec1**

**Lecture: Asraa A.H.**

**HEADER FILES::**

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

 In C programming language, printf() function is used to print the “character, string, float, integer, octal and hexadecimal values” in to the output screen whereas scanf() function is used to read character, string, numeric data from keyboard .

 **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.

**TYPES IN C::**

 In C, there are three kinds of types that variables can have::

1.Primary, 2.Derived, and 3.pointers. Half of the game in getting things right in C is keeping yourself from being confused about types. This lecture tries to elaborate on this a little.

**PRIMARY TYPES::**

There are 6 Primary types in C:

 **char** :: 1 byte

 **short**  :: 2 bytes

 **int**  :: 4 bytes

 **long**  :: 4 or 8 bytes, depending on the system and compiler

 **float**  :: 4 bytes

 **double**  :: 8 bytes

 To get the exact size of a type or a variable on a particular platform, you can use the  **sizeof**  operator. The expressions sizeof(type) yields the storage size of the object or type in bytes. For example, **sizeof(long)** will return either 4 or 8, depending on how big a **long** is in your system.

 You can declare variable in one of three places: As **a global variable**, as a **procedure parameter**, and as a **local variable**.

**Example 1::**

#include <stdio.h>

void main()

{ int number;

 // printf() dislpays the formatted output

 printf("Enter an integer: ");

 // scanf() reads the formatted input and stores them

 scanf("%d", &number);

 // printf() displays the formatted output

 printf("You entered: %d", number);

 getch();

}

 Output



In this program, an integer variable number is declared.

 The printf() function displays Enter an integer: on the screen. Then, the scanf()function reads an integer data from the user and stores in variable number.

Finally, the value stored in the variable number is displayed on the screen using printf()function. & is used to assign the input value to the variable

and store it at that particular location

 You must be wondering what is 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 number |
| %f | Scan or print a floating point number |
| %c | To scan or print a character |
| %s | To scan or print a character string. |

**Example 2::**

#include <stdio.h>

int i;

void main(int argc)

{ long j;

 argc=10;

 j = sizeof(j);

 i = sizeof(int);

 printf("Argc: %d\n",argc );

 printf("Sizeof(long): %d\n", j);

 printf("i %d\n", i);

 getch();

}

 There are three primary int variables here: **i**, **j**, and **argc**. **I** is a global variable. **J** is a local variable, and **argc** is a parameter.

Output



**Example 3::**

#include <stdio.h>

Void main()

{ char c;

 printf("Enter a character: ");

 // Reads character input from the user

 scanf("%c", &c);

 // %d displays the integer value of a character

 // %c displays the actual character

 printf("ASCII value of %c = %d", c, c);

 getch();

}

**Output**

****

 In this program, user is asked to enter a character which is stored in variable c. The ASCII value of that character is stored in variable c rather than that variable itself.

When %d format string is used, 71 (ASCII value of 'G') is displayed.

When %c format string is used, 'G' itself is displayed