Structured Programming
CSE 103

Professor Dr. Mohammad Abu Yousuf

Input and Output

Printf () Function

- ***printf()*** is a library function declared in <stdio.h>
- Output data can be written from the computer onto a standard output device using the library function ***printf()*** .
- That is, the ***printf()*** function moves data from the computer's memory to the standard output device, whereas the scanf function enters data from the standard input device and stores it in the computer's memory.
- In general terms, the ***printf()*** function is written as

  printf (control string, arg1, arg2, . . . , argn).

  where control string or format specifier refers to a string that contains formatting information, and arg7, arg2, . . . , argn are arguments that represent the individual output data items.

Printf () Function

- The control string consists of individual groups of characters, with one character group for each output data item. Each character group must begin with a percent sign (%). In its simplest form, an individual character group will consist of the percent sign (%) , followed by a ***conversion character*** indicating the type of the corresponding data item

Printf () Function

Commonly used conversion character for data output

Conversion Character	*Meaning*
c	Data item is displayed as a single character
d	Data item is displayed as a signed decimal integer
e	Data item is displayed as a floating-point value with an exponent
f	Data item is displayed as a floating-point value without an exponent
g	Data item is displayed as a floating-point value using either e-type or f-type conversion, depending on value. Trailing zeros and trailing decimal point will not be displayed.
i	Data item is displayed as a signed decimal integer
o	Data item is displayed as an octal integer, without a leading zero
s	Data item is displayed as a string
u	Data item is displayed as an unsigned decimal integer
x	Data item is displayed as a hexadecimal integer, without the leading 0x

Printf () Function

Control string or format specifiers:

%c	The character format specifier.
%d	The integer format specifier.
%i	The integer format specifier (same as %d).
%f	The floating-point format specifier.
%o	The unsigned octal format specifier.
%s	The string format specifier.
%u	The unsigned integer format specifier.
%x	The unsigned hexadecimal format specifier.
%%	Outputs a percent sign.

Printf () Function

```
#include <stdio.h>
// program prints hello world
int main() {
    printf ("Hello world!");
    return 0;
}
```

Output: Hello world!

```
#include <stdio.h>
// program prints a number of type int
int main() {
    int number = 4;
    printf ("Number is %d", number);
    return 0;
}
```

Output: Number is 4

```
#include <stdio.h>

main()
{
    char item[20];
    int partno;
    float cost;

    . . . . .

    printf("%s %d %f", item, partno, cost);

    . . . . .

}
```

- Within the *printf* function, the control string is **"%s %d %f. It contains three character groups.**
- The first character group, **%s, indicates that the first argument (item) represents a string.**
- The second character group, %d, indicates that the second argument (partno) represents a decimal integer value, and
- The third character group, %f, indicates that the third argument (cost) represents a floating-point value.

Let's Try !!!

- A C program contains the following variable declarations.

  float a = 2.5, b = 0.0005, c = 3000.;

  Show the output resulting from each of the following ***printf*** statements.

  (a) printf ("%f %f %f",a, b, c);

  (b) printf('%3f %3f %3f ", a, b, c);

  (c) printf("%8f %8f %8f", a, b, c);

  (d) printf("%8.4f %8.4f %8.4f", a, b, c);

  (e) printf("%8.3f %8.3f %8.3f", a, b, c);

Scanf ()

- *scanf()* is a library function declared in <stdio.h>
- Input data can be entered into the computer from a standard input device by means of the C library function ***scanf().***
- This function can be used to enter any combination of numerical values, single characters and strings.
- The ***scanf*** function is written as

  scanf(contro1 string, argl, arg2, . . . , argn)

- where ***control string*** refers to a string containing certain required formatting information, and ***argl, arg2, . . . argn*** are arguments that represent the individual input data items. *The arguments are actually pointers that indicate where the data items are stored in the computer's memory.*

Scanf ()

- The control string consists of individual groups of characters, with one character group for each input data item. Each character group must begin with a percent sign (%).
- In its simplest form, a single character group will consist of the percent sign, followed by a conversion character which indicates the type of the corresponding data item.
- ***Each variable name must be preceded by an ampersand (&).***

Scanf ()

Commonly Used Conversion Characters for Data Input

Conversion Character	*Meaning*
c	data item is a single character
d	data item is a decimal integer
e	data item is a floating-point value
f	data item is a floating-point value
g	data item is a floating-point value
h	data item is a short integer
i	data item is a decimal, hexadecimal or octal integer
o	data item is an octal integer
s	data item is a string followed by a whitespace character (the null character \0 will automatically be added at the end)
u	data item is an unsigned decimal integer
x	data item is a hexadecimal integer
[. . .]	data item is a string which may include whitespace characters (see explanation below)

```
1.#include<stdio.h>
2.int main(){
3.    int number;
4.    printf("enter a number:");
5.    scanf("%d",&number);
6.   printf("cube of number is:%d ",number*number*number);
7.   return 0;
8.}
```

```
Output:
enter a number: 5
cube of number is:125
```

- The **scanf("%d",&number)** statement reads integer number from the console and stores the given value in number variable.

- The **printf("cube of number is:%d ", number * number * number)** statement prints the cube of number on the console.

More on scanf()

- Consider the C program

```
#include <stdio.h>

main()
{
    int a, b, c;

    . . . . .

    scanf("%3d %3d %3d", &a, &b, &c);

    . . . . .
}
```

More on scanf()

- When the program is executed, three integer quantities will be entered from the standard input device (the keyboard).
- Suppose the input data items are entered as **1 2 3.** Then the following assignments will result: a = 1 , b = 2 , c = 3
- If the data had been entered **as**

  123 456 789

  Then the assignments would be

  a = 123, b = 456, c = 789
- Now suppose that the data had been entered **as**

  123456789

  Then the assignments would be

  a = 123, b = 456, c = 789

  as before, since the first three digits would be assigned to a, the next three digits to b, and the last three digits to c.

More on scanf()

- Finally, suppose that the data had been entered **as**

  1234 5678 9

  The resulting assignments would now be

  a = 123, b = 4, c = 567

  The remaining two digits (8 and 9) would be ignored, unless they were read by a subsequent scanf statement.

Example of input and output in C language that prints addition of 2 numbers.

```
1.#include<stdio.h>
2.int main(){
3.    int x=0,y=0,result=0;
4.    printf("enter first number:");
5.    scanf("%d",&x);
6.    printf("enter second number:");
7.    scanf("%d",&y);
8.    result=x+y;
9.    printf("sum of 2 numbers:%d ",result);
10.   return 0;
11.}
```

Output:
enter first number:9
enter second number:9
sum of 2 numbers:18

Thank you