Output using printf

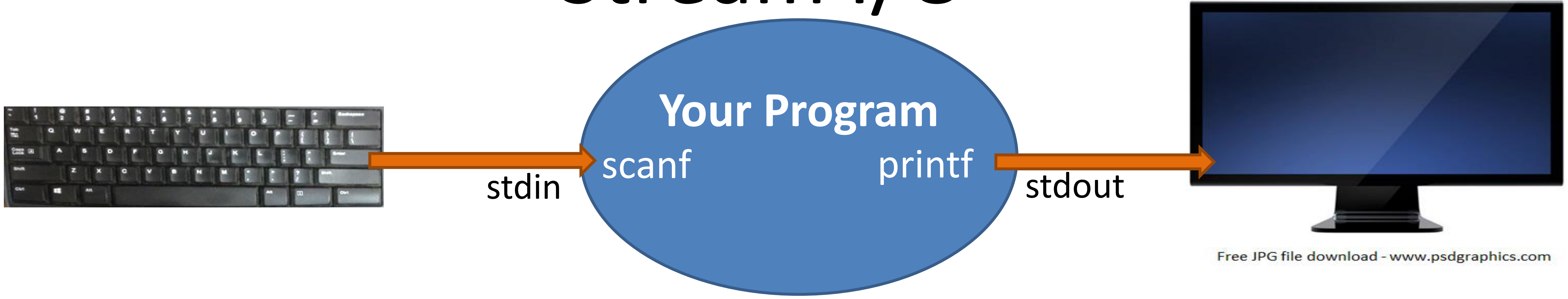
Dan McElroy

Aug 2018

Should you know printf?

The **printf** output formatter was developed for the C-language. **printf** is used in Java and many other languages and can still be used in C++. It is important to know **printf** even if you are learning C++.

Stream I/O



A common way for a program to output to the display is to use **printf** which stand for print-formatted.

```
printf ("The number squared is %f\n", a*a);
```

Use `\n` to move the cursor to the next line on the display. The backslash character \ is called the 'escape' character. It gives the next character a special meaning. Make sure you enter the backslash \ and not the slash /

NOTE: Operating systems like Linux can use pipes and redirection to cause stdin and stdout to use other devices or even other programs

How Does **printf** work?

```
printf (control, arg1, arg2, ...);
```

printf converts and prints text from the control string and the arguments referred to in the control string. There can be zero or more arguments. For example:

```
int age = 25;
char name[ ] = "Joe";
printf ("Greetings\n");
printf ("Hello %s, you are %d years old.\n", name, age);
```

The output will be:

```
Greetings
Hello Joe, you are 25 years old.
```

How Does **printf** work?

printf with
no arguments

```
int age = 25;
char name[] = "Joe";
printf ("Greetings\n");
printf ("Hello %s, you are %d years old.\n", name, age);
```

The output will be:

```
Greetings
Hello Joe, you are 25 years old.
```

The \n causes the cursor to go to the next line.

The **printf** Format Specifiers

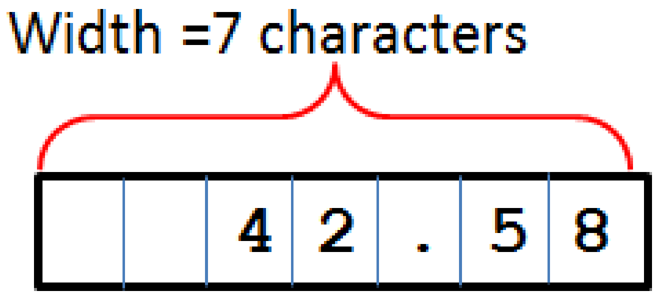
Control	Description
%d	Decimal integer. The argument should be an integer.
%o	Octal integer. The argument should be an integer.
%x	Hexadecimal integer. The argument should be an integer.
%X	Hexadecimal integer. A-F is displayed in upper case
%c	The argument should be the address of a character.
%s	Character string. The argument should a character array
%e	Floating point number in engineering format. The argument should be a float. Example 5632 displays as 5.632E3
%f	Floating point number. The argument should be a float.
%lf	Long-float. The argument should be a double.
%g	Use %e or %f which ever is shorter. Non-significant zeros are not printed
%%	If the character after the % is not a control character, print it. %% prints %

printf Field Width and Precision

The printf control specifiers can have an optional field width and/or precision listed. Examples:

```
double length = 42.578;
printf ("%7.2lf", length);
```

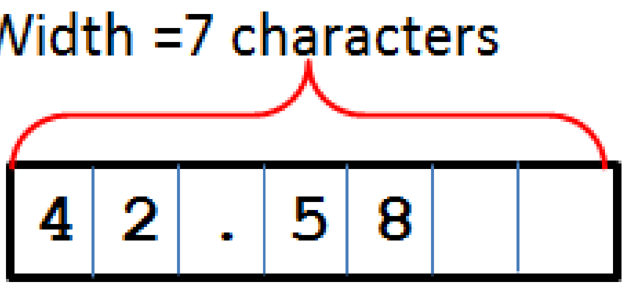
small-L



printf uses a total of 7 character positions with 2 digits past the decimal and is right-justified

```
double length = 42.578;
printf ("%-7.2lf", length);
```

Left-justified

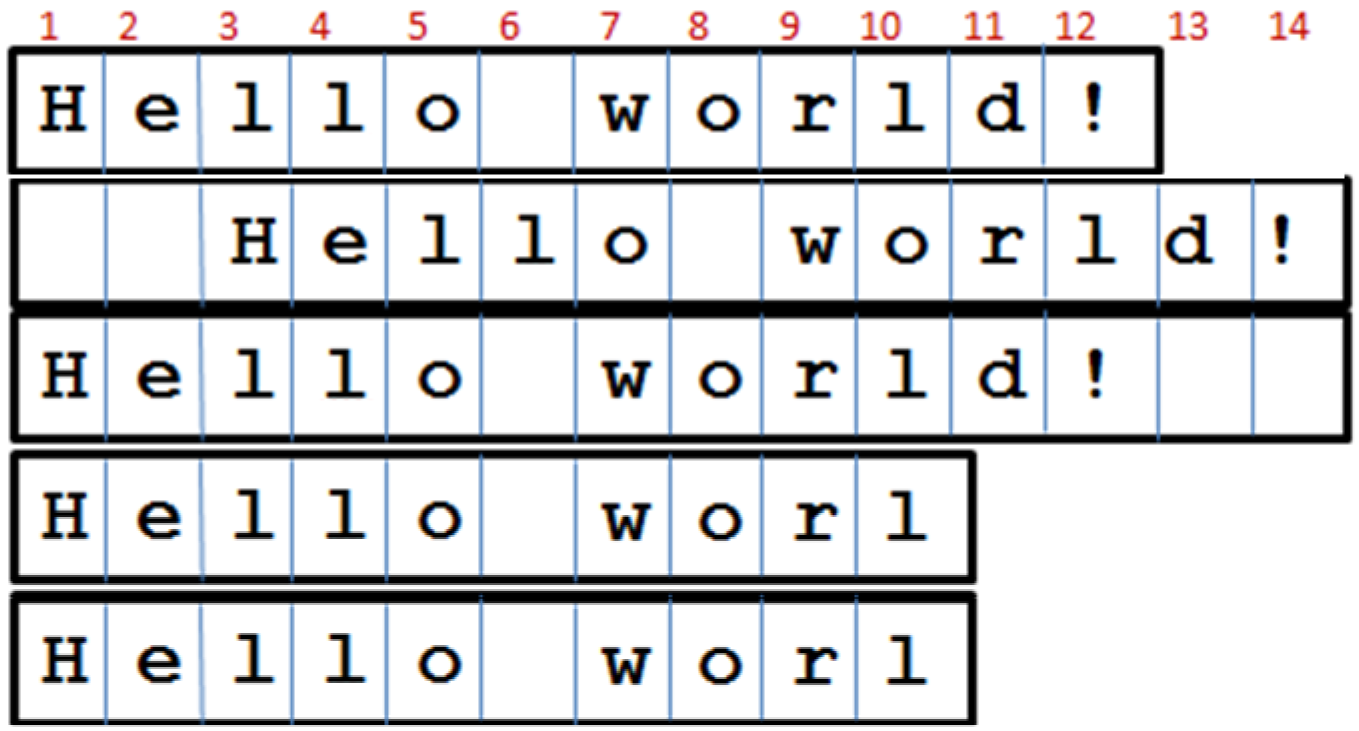


printf Field Width and Precision

Examples with %s:

```
char msg[ ] = "Hello world!";

printf ("%s", msg);
printf ("%14s", msg);
printf ("%-14s", msg);
printf ("%7.10s", msg);
printf ("%-7.10s", msg);
```



```
Hello world!
  Hello world!
Hello world!  
Hello worl
Hello worl
```

For strings, the numbers after the % and before the s in the format string specify the minimum and maximum number of characters to 'print'. In the last two examples, since only 10 of the 12 characters in the string are printed, the outputs are the same because there are no spare spaces to be utilized.

WARNING

printf uses the control string to determine the number and data type for the arguments that follow. printf gets confused and prints nonsense answers if there are not enough arguments, they are the wrong type, or the arguments are not listed in the same order as the control specifiers!