

Printing

Last updated 8/31/21

Printing

- Most embedded systems do not print out any results
 - We will use printing to:
 - Aid in our programming practice
 - Debug our programs
- The output from a `printf()` function is sent to the standard output “stream” (`stdout`)
 - `stdout` in our case will be the Tera Term window
- The `printf()` function is extremely costly in terms of CPU cycles
 - Special care must be taken to prevent it from impacting the timing of our programs

Printing Text

- Printing text (a string constant)

command: `printf()`

argument: “text to display”

optional: “\n” prints a line feed (a new line is started)

“\t” prints a tab

Examples:

`printf(“Hello World”);` - prints: **Hello World**

`printf(“Foo, boo, soo\n”);` - prints: **Foo, boo, soo** and a new line is started

`printf(“\n\n”);` - prints 2 blank lines and a 3rd new line is started

Printing Text

- Special Characters

- Special characters need to be treated differently because they are used by the printf() function to do special things

\ “

To print one of these characters they must be preceded by a backslash \

Examples:

printf(“Hello \ World”); - prints: Hello \ World

printf(“Foo, \”boo\”, soo\n”); - prints: Foo, “boo”, soo and a new line

% is special – precede it with another %

printf(“12%%\n\n”); - prints: 12% and 2 new lines

Printing Variables

- Printing a variable

This is the location where a variable of **type** will be printed - **%** is a special character to indicate the "type" follows next

This is the name of the **variable** to print

command: `printf()`

Argument(s): "text to display **%type** more text", **variable**

optional: "`\n`" prints a line feed (a new line is started)

`\t` prints a tab

type: **i** → int, **f** → float, **c** → character

```
int ave;
```

```
ave = 12;
```

```
printf("The average is %i", ave);
```

- prints: **The average is 12**

```
printf("%f is the average\n", ave);
```


- prints: **12.00000 is the average** and a new line is started

Printing Variables

- Each variable in a single print statement needs its own format descriptor
- Each variable is separated by a comma

```
int count;  
float ave;  
count = 21;  
ave = 12.2;
```

```
printf("The average is %f, with %i scores", ave, count);
```



- prints: The average is 12.200000, with 21 scores