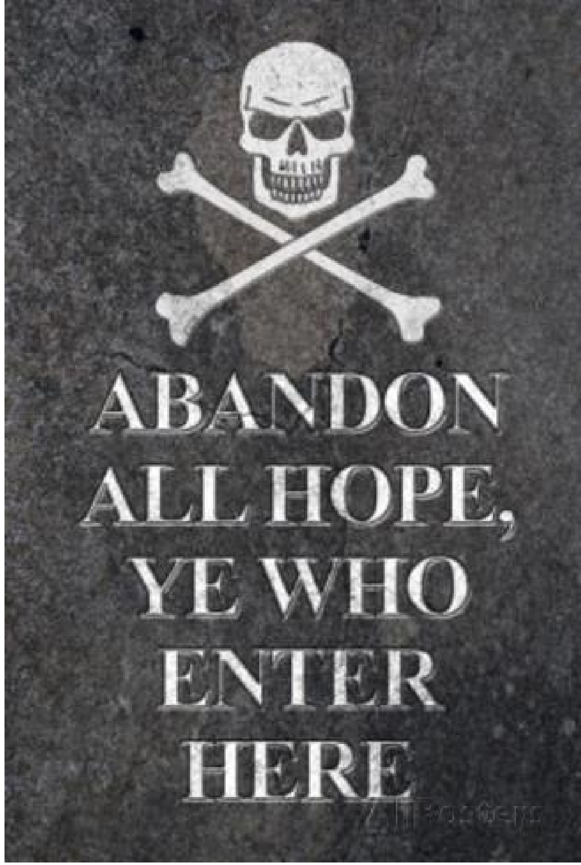
Print Formatting

Escape Characters

- Allow different characters or actions to be done within the printf() statement

- We've already seen one…

  >> \n ➔ newline (equivalent to hitting the enter key)



Escape Characters

- Carriage return (\r): Popular with typewriters (but has limited functions in C)
    - Moves cursor to the beginning of the line you are TYPING ON

Example:

```
>> printf("test1\r123\n");
```

Output: *123t1*

Escape Characters

- Backspace (\b): Popular with typewriters (but has limited functions in C)
  - Moves cursor back one space

Example:

```
>> printf(“test1\b23\n”);
```

Output: *test23*

Escape Characters

- Formfeed (\f)
  - Interpreted different ways on different
  - On your VM, it goes down exactly one line (spaces included)

```
>> printf(“test123\fHello World\n”);
```

```
Output: test123
               Hello World
```

Escape Characters

- Alert (\a)
  - Makes a beep sound from the computer (depend on your machines...)

```
>> printf("\a\a\a\a");
```

Output: */*Nothing to the screen, but will make 4 beeps)*

Escape Characters

- Tabs
  - Horizontal tab (\t)
  - Vertical tab (\v)
    - Does the exact same thing as formfeed (/f)

```
>> printf("Hello\tWorld\n");
```

```
Output: Hello	World
```

Escape Characters

- The backslash, single, and double quotes are all meaningful characters in *printf*
    - What if we need to print out a quote or a backslash?

>> printf("Steve Jobs once said "Everyone should learn to code. It teaches you how to think".\n");

Output: ???

Escape Characters

- What if we need to print out a quote or a backslash?
  - Use an additional backslash!
    - Quote: \’ *or* \”
    - Backslash: \\

>> printf(“Steve Jobs once said \“Everyone should learn to code. It teaches you how to think\”.\n”);

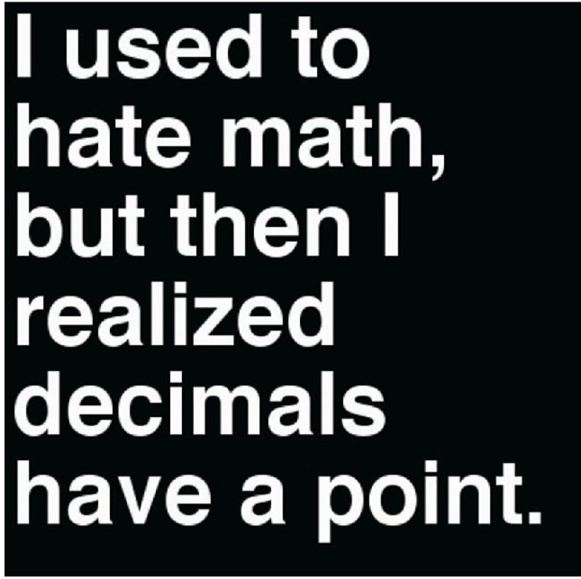
Number formatting

- By default, %f prints out a decimal number to 6 six decimal places

  ```
  >> float foo = 6

  >> printf(“Foo is %f\n”, foo);
  ```

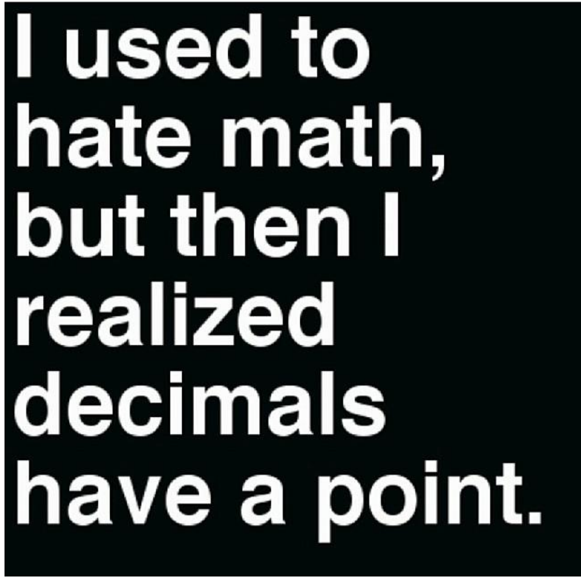
  Output: Foo is 6.000000



I used to hate math, but then I realized decimals have a point.

Number formatting

- By default, %f prints out a decimal number to 6 six decimal places
- To change the default, we can add *formatters*
    - Go in between the % and the *f*



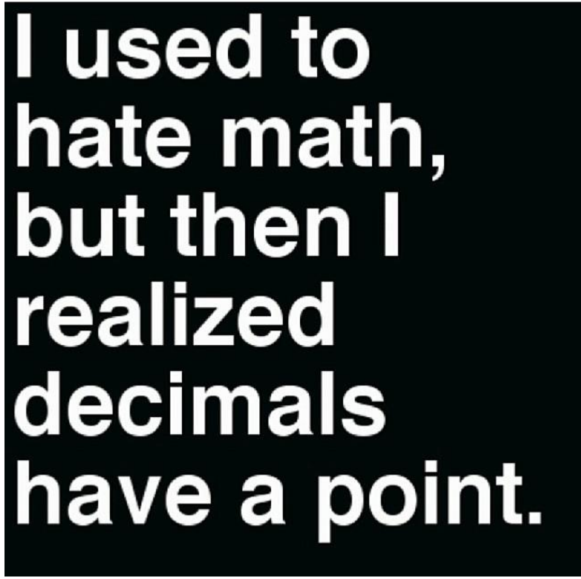
I used to hate math, but then I realized decimals have a point.

Number formatting

- To change the default, we can add *formatters*
    - Written as a decimal number: numbers after the decimal refer to the number of decimals that should be shown

```
>> printf(“Foo is %.4f\n”);
```

```
Output: Food is 6.0000
```



I used to hate math, but then I realized decimals have a point.

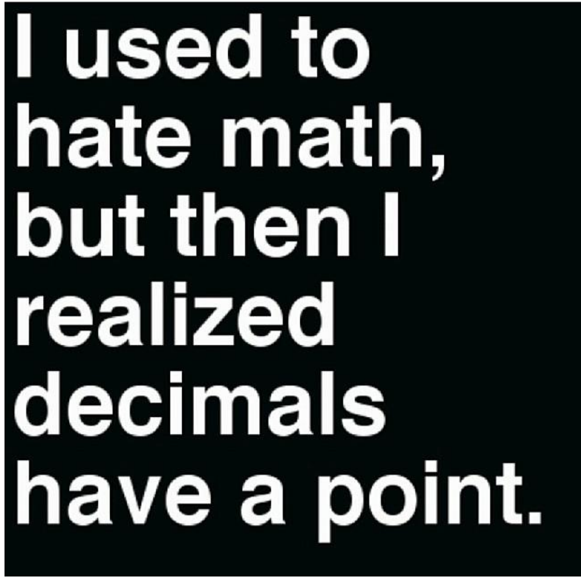
Number formatting

- To change the default, we can add *formatters*
  - Written as a decimal number: numbers BEFORE the decimal refer to the total number of spaces used by the number

```
>> printf(“Foo is %5.2f\n”);
```

```
Output: Foo is _6.00
```

The underscore is not printed: it is printed as a space character. Combined, the space, the six, the decimal point, and the 2 zeros make up 5 total spaces printed out.



I used to hate math, but then I realized decimals have a point.