Same message ...

... in *any* language?

The PYTimes

Today's whether: if, elif, or else!

Three-eyed troubles:
GradeScope, Python, & VSCode...

Aliens Attack! Picobot programmer Z. Dodds was subject of a bizarre encounter yesterday with **three-eyed aliens**. The trinocular tourists, it seems, were conducting experiments that would help them understand "**how humans think**."

It seems the aliens used a shrinking ray, which let them enter the programmer's head in order to see what was happening. A witness reports **deeply disappointed voices** emanating from within.

To escape the attack, Dodds had to turn the ray on himself – as he shrank, the aliens quickly flew off, departing so fast that he was **unable to use the reverse ray** before they left. "No worries," Dodds mused – in fact, this might help me tomorrow... "

see three-eyed alien attack, p. 42

Composite sketch of one of the attackers drawn from three-eyewitness accounts

Homework #1
Due Mon., 9/16

0) Reading + response

1) **Lab**: *data*

2) **Lab**: *functions*

3) The *fun* in *fun*ctions!

(ExCr) *Pig Latin, et alia*

Automatic translation: if it's possible for human languages... perhaps for CS, too?

CS5 *Favorites*!

In-person help: *Tutoring hours…*

Monday 8pm-10pm or 11pm *Grutors: if you want to stay until 11pm, great! But, please don't stay later than that… !*

CS5 at HMC's LAC (up to 5)	**CS5 *away* from HMC (up to 5)** Be sure to include WHERE you'll be!	**CS 42 (up to 2)**	**CS 60 (up to 2)**
Alicia Pentico (apentico@g.hmc.edu; green)	Frankie Konner (Pitzer Grovehouse) <fkonner@students.pitzer.edu>	Christina Lau (cllau@g.hmc.edu)	Elena Ehrlich (eehrlich@g.hmc.edu)
Aely Aronoff (aaronoff@hmc.edu)	Graham Brady (pitzer grove house) (gbrady@students.pitzer.edu)	David Mindlin	Harris McCullers (harrismccullers@gmail.com) ***I am willing to switch, email me**
	8-11pm		
Jenna Kahn (Green) (jmkahn@hmc.edu)	Kate Emery (New Hall Kitchen) (kemery5437@scrippscollege.edu)		
Jerry Liang (jyliang@hmc.edu)	Jacob Adolphe (Chall lounge probably somewhere by the TV) <jadolphe21@cmc.edu>		
Isaiah Fujii Bresnihan			

I don't know how many Scripps CS5 grutors there are this semester, but it has come to my attention that there are not enough CS5 tutoring opportunities on Scripps right now. I had my first Scripps hours yesterday, which at the time were the only tutoring hours available on Scripps, and I had a lot of people show up (especially considering the fact that it was 3 or 4 days before the first deadline of the

Katherine T.

Yes, we have hours at Scripps!

In-person help: *Tutoring hours…*

Monday 8pm-10pm or 11pm *Grutors: if you want to stay until 11pm, great! But, please don't stay later than that… !*

CS5 at HMC'
Alicia Pentico (apentico@g
Aely Aronoff (
Jenna Kahn ((jmkahn@hm
Jerry Liang (j
Isaiah Fujii B

In-person help: *Tutoring hours…*


Ready for
Picobot!


Picobot
tutoring
gets real!

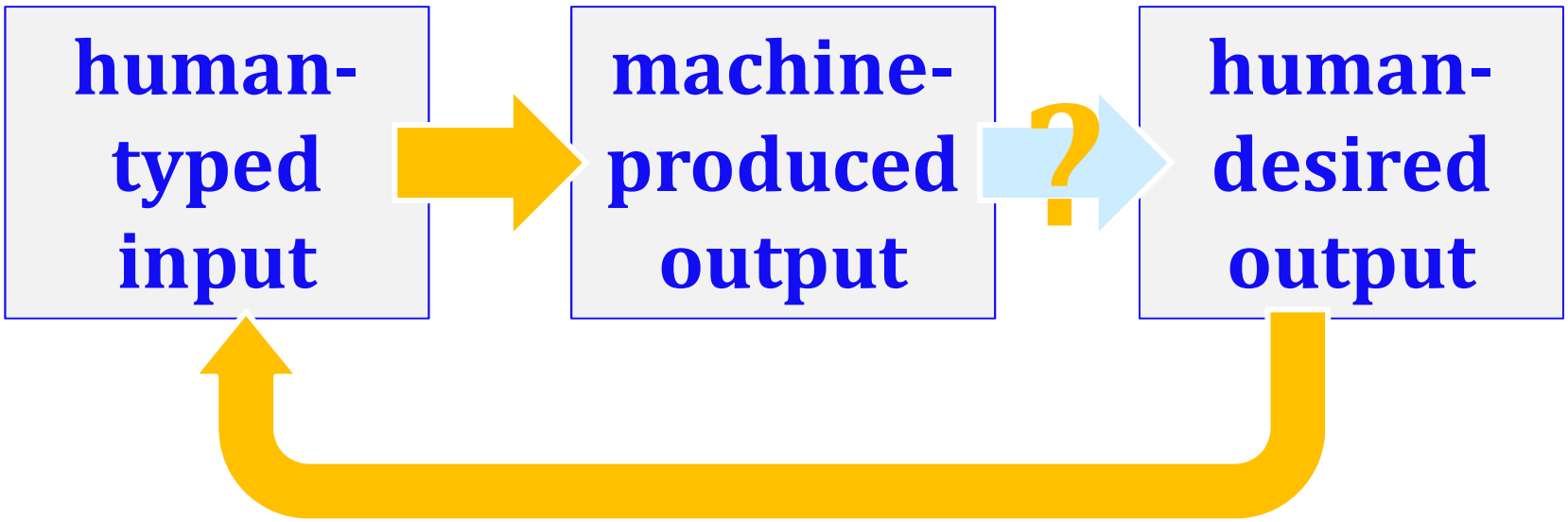
Homework 0... *The adventure begins!*

Lab!

Poptarts!

The ***challenge*** of programming…

syntax	semantics	intent
How it looks	What it does	What it should do



learning a language ~ ***syntax***

unavoidable, but not the point

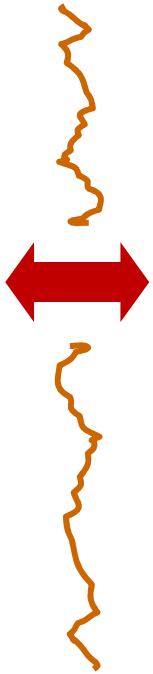
… but learning CS ~ ***semantics*** 

learning how machines ***think***!

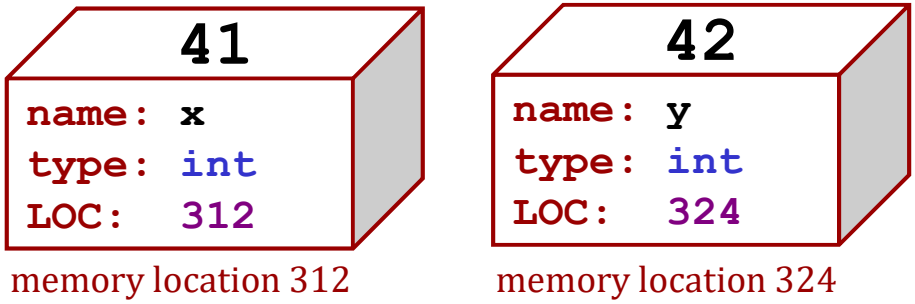
Inside the machine…

What's behind the scenes (processing + memory):

Computation



Data Storage

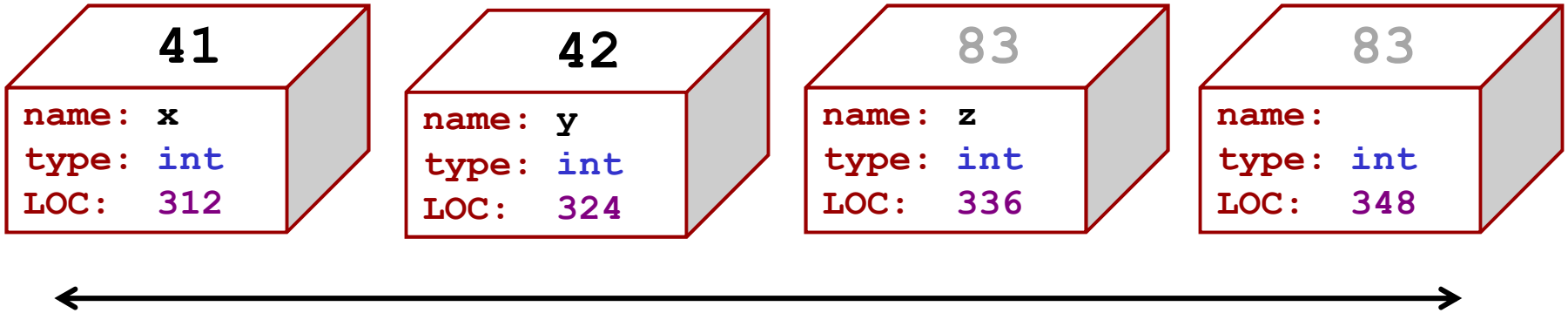


memory location 312 memory location 324

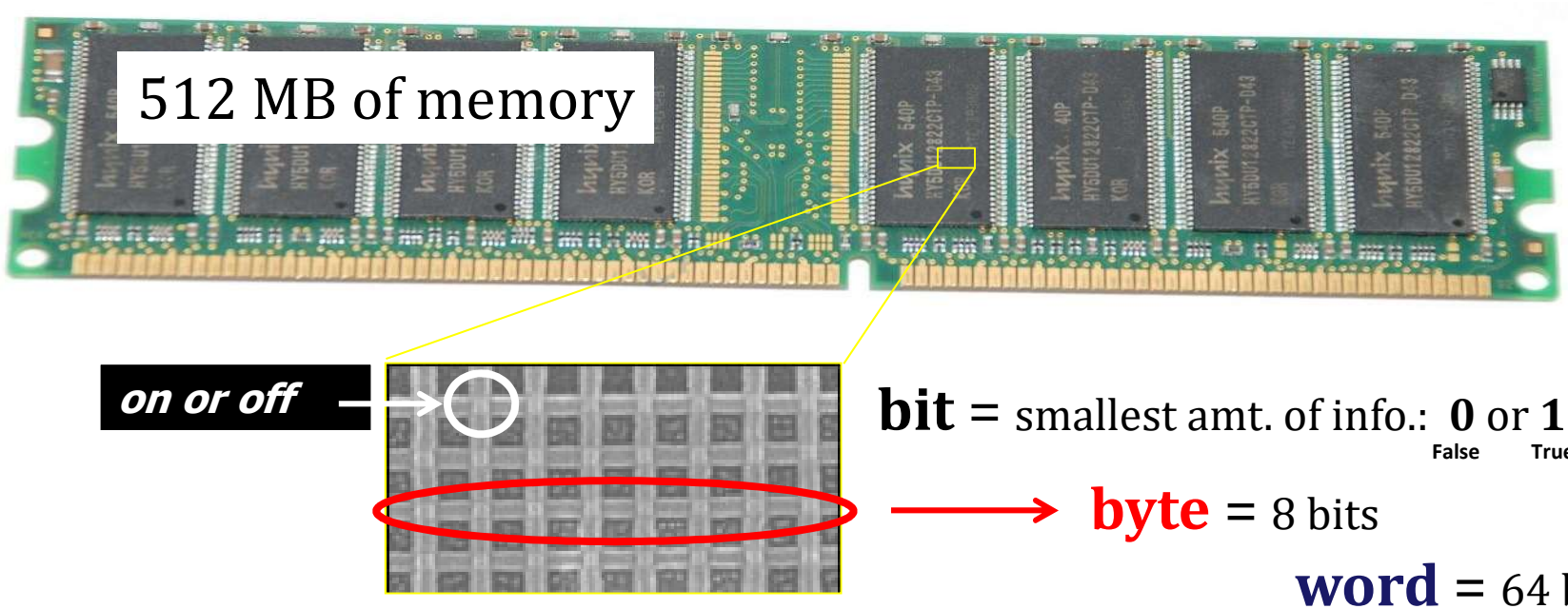
variables ~ ***boxes***

Memory!

Random Access Memory



a big list of boxes, each with a name, type, location, and value



bit = smallest amt. of info.: **0** or **1**
False True

byte = 8 bits

word = 64 bits

All languages use ***datatypes***

Type	Example	What *is* it?
`float`	`3.14` or `3.0`	numeric values with a fractional part, *even if the fractional part is .0*
`int`	`42` or `10**100`	integers – Python has *infinite precision ints!*
`bool`	`True` or `False`	the T/F results from a test or comparison: **==, !=, <, >, <=, >=**

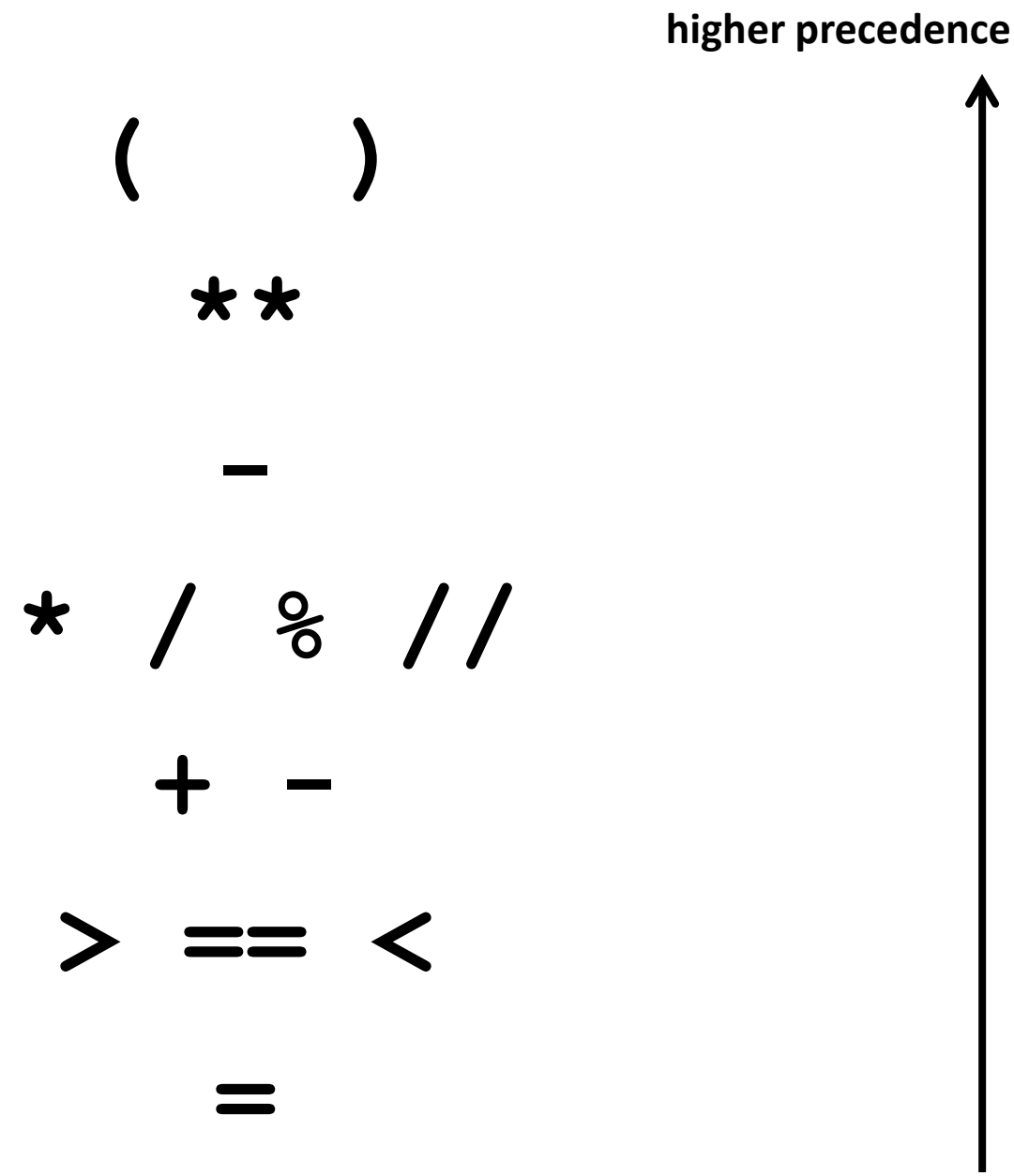
"Boolean values"

"Boolean operators"

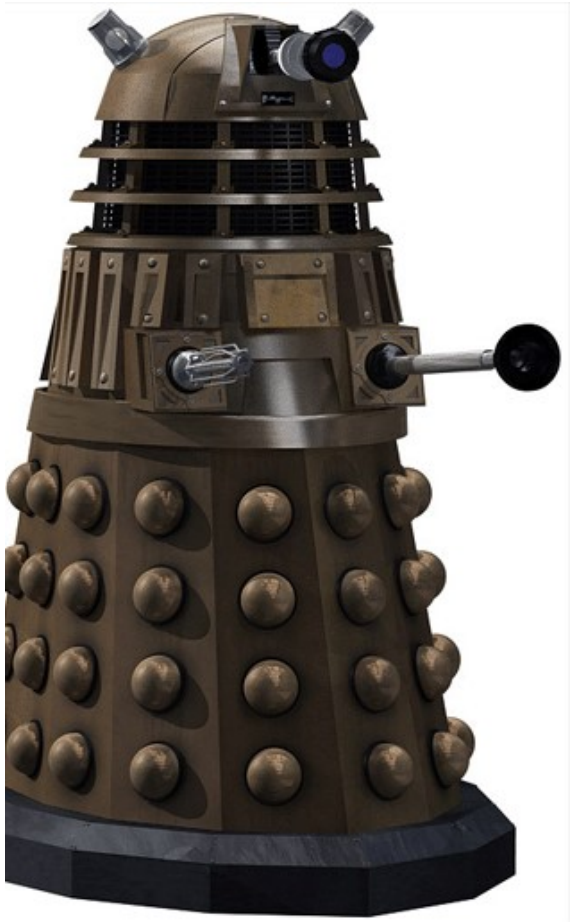
George Boole

type(x)

Operate!



O-per-ate!



higher precedence

( )

**

-

* / % //

+ -

> == <

=

Python operators

higher precedence

parens	( )
power	**
negate	-
times, mod, divide	* / % //
add, subtract	+ -
compare	> == <
assign	=

It's not worth remembering all these `%+/*` things!
I'd recommend parentheses over precedence.

`%` the ***mod*** operator

```
7 % 3
```

```
9 % 3
```

```
8 % 3
```

```
30 % 7
```

`x%y` is the ***remainder*** when **`x`** is divided by **`y`**

For what values of **`x`** are these **`True`**?

- `x%2 == 0`
- `x%2 == 1`
- `x%4 == 0` ← If x is a year, what happens on these years!?
- `x%4 == 3` ← What happens on these years, football-wise!?

`//` *integer* division

```
7 // 3
```

```
8 // 3
```

```
9 // 3
```

```
30 // 7
```

`x//y` is **`x/y`**, *rounded-down* to an integer

`//` *integer* division

```
7 // 3
```

```
8 // 3
```

```
9 // 3
```

```
30 // 7
```

`x//y` is **`x/y`**, *rounded-down* to an integer

Why?

Decomposition of 30 into 7's:

```
30 ==  (4)*7  +  (2)
```

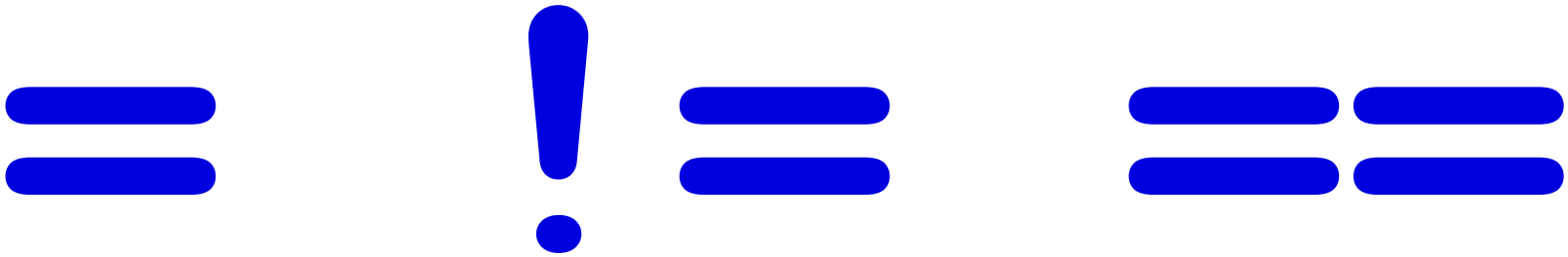
Decomposition of x into y's:

```
x == (x//y)*y + (x%y)
```

# of full y's in x

remainder after "taking" all of the full y's in x

the "equals" operators



This is true – *but what is it saying!?*

the "equals" operators

=

SET equals

!=

isn't equal to

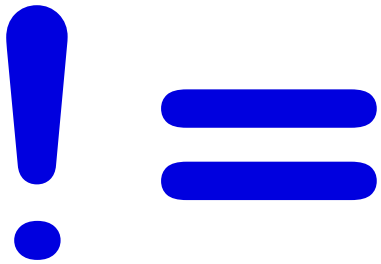
==

TEST equals

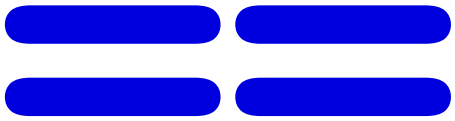
I want === !

the "equals" operators

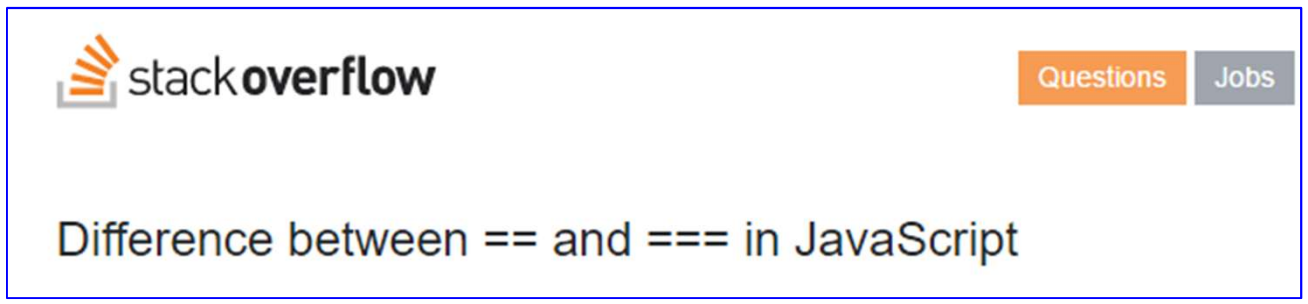
SET equals



isn't equal to



TEST equals



I want === !

how = works

"Quiz"

_____
name(s)

Run these lines

```
x = 41
y = x + 1
z = x + y
```

What are **x**, **y**, and **z** at this time?

x	y	z

Then run this line

```
x = x + y
```

What are **x**, **y**, and **z** at this time?

x	y	z

Extra!

```
a = 11//2
b = a%3
c = b** a+b *a
```

What are the values of **a**, **b**, and **c** after the 3 lines, at left, run?

a	b	c

Inside the machine…

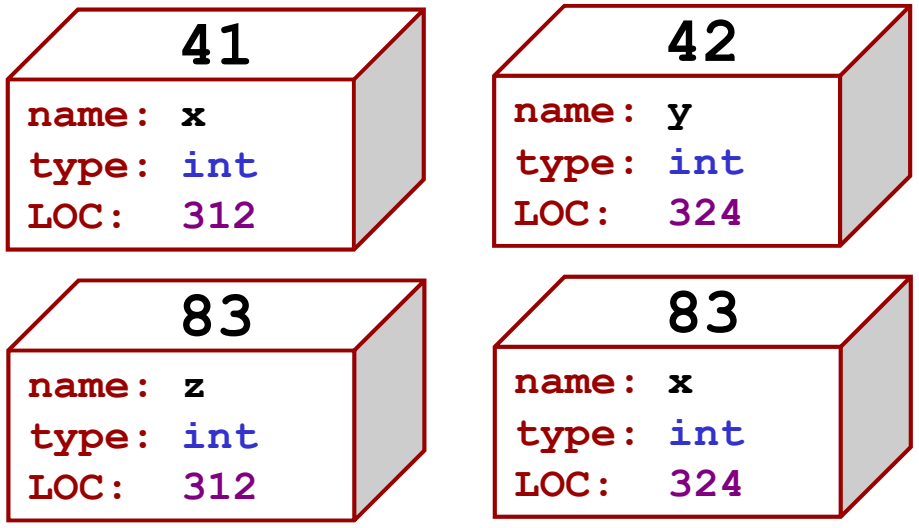
What's happening in python:

```
x = 41
y = x + 1
z = x + y
x = x + y
```

What's happening behind the scenes (in memory):

Computation

Memory (Data Storage)



id, del

how = works

"Quiz"

– try these on the back page first...

Run these lines

```
x = 41
y = x + 1
z = x + y
```

What are **x**, **y**, and **z** at this time?

x	y	z
41	42	83

Then run this line

```
x = x + y
```

What are **x**, **y**, and **z** at this time?

x	y	z
83	42	83

Extra!

```
a = 11//2
b = a%3
c = b** a+b *a
```

What are the values of **a**, **b**, and **c** after the 3 lines, at left, run?

a	b	c
5	2	??

how = works

"Quiz"

– try these on the back page first...

Run these lines

```
x = 41
y = x + 1
z = x + y
```

What are **x**, **y**, and **z** at this time?

x	y	z
41	42	83

Then run this line

```
x = x + y
```

What are **x**, **y**, and **z** at this time?

x	y	z
83	42	83

Extra!

```
a = 11//2
b = a%3
c = b** a+b *a
```

What are the values of **a**, **b**, and **c** after the 3 lines, at left, run?

a	b	c
5	2	42

Popular culture [edit]

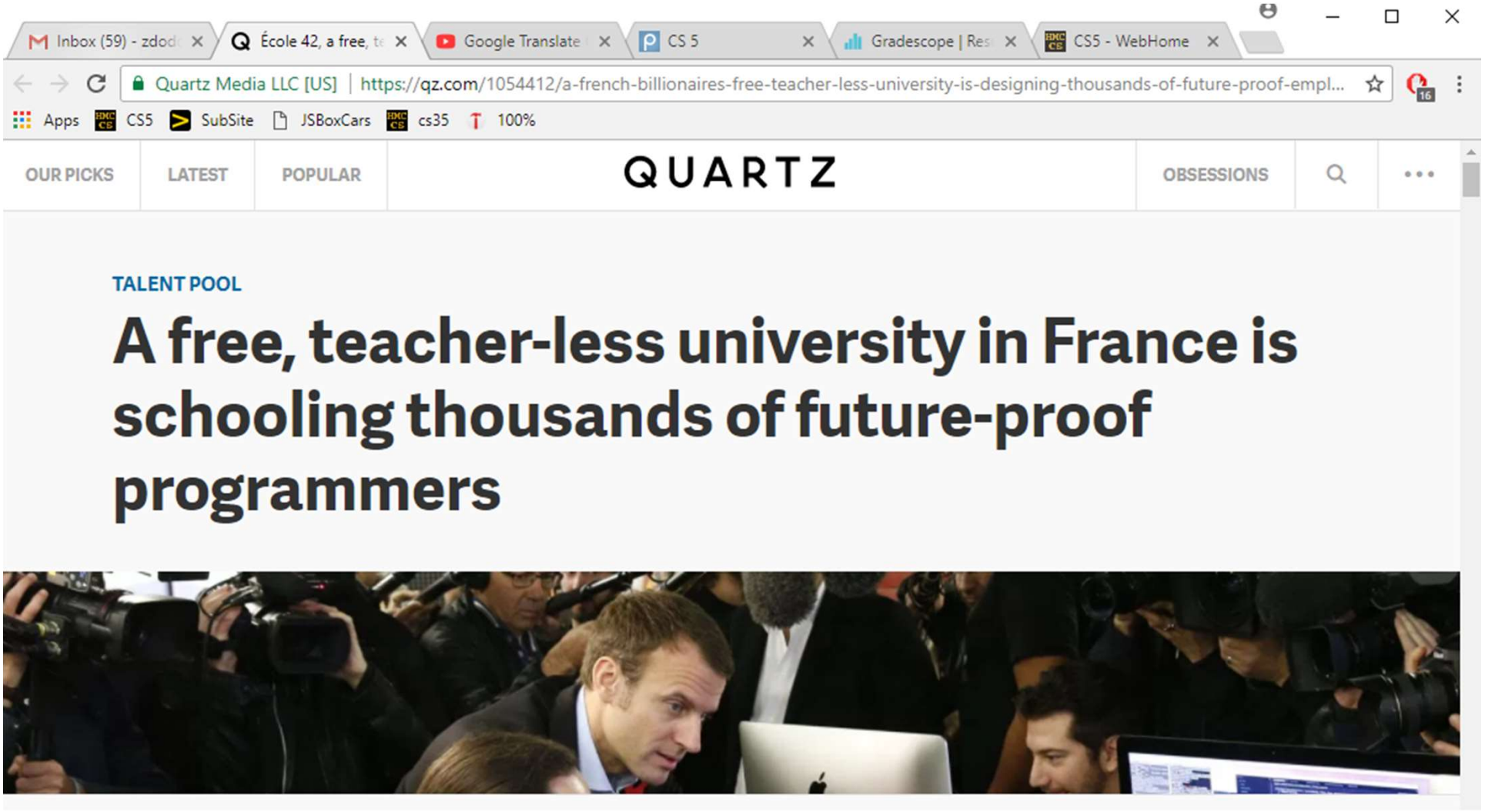
The Hitchhiker's Guide to the Galaxy [edit]

The Answer to the Ultimate Question of Life, The Universe, and Everything.

The number 42 is, in *The Hitchhiker's Guide to the Galaxy* by Douglas Adams, the "Answer to the Ultimate Question of Life, the Universe, and Everything", calculated by an enormous supercomputer named Deep Thought over a period of 7.5 million years. Unfortunately, no one knows what the question is. Thus, to calculate the Ultimate Question, a special computer the size of a small planet was built from organic components and named "Earth". The Ultimate Question "What do you get when you multiply six by nine"[17] was found by Arthur Dent and Ford Prefect in the second book of the series, *The Restaurant at the End of the Universe*. This appeared first in the radio play and later in the novelization of *The Hitchhiker's Guide to the Galaxy*. The fact that Adams named the episodes of the radio play "fits", the same archaic title for a chapter or section used by Lewis Carroll in "The Hunting of the Snark", suggests that Adams was influenced by Carroll's fascination with and frequent use of the number. The fourth book in the series, the novel *So Long, and Thanks for All the Fish*, contains 42 chapters. According to the novel *Mostly Harmless*, 42 is the street address of Stavromula Beta. In 1994 Adams created the *42 Puzzle*, a game based on the number 42.

among *many* 42 references...

mostly in cs5...!



among *many* 42 references...

mostly in cs5...!

Are numbers enough for ***everything***?

Yes and no…

You need ***lists*** of numbers, as well!

and ***strings*** - lists of characters - too.

Both of these are Python ***sequences…*** ➡

strings: *textual* data

strings

```
s = 'scripps'
c = 'college'
```

type...

```
type(s)
```

len

```
len(s)
```

add!

```
s + c
```

multiply!!

```
2*s + 3*c
```

strings: *textual* data

Given
```
s1 = 'ha'
s2 = 't'
```

What are `s1 + s2`

`2*s1 + s2 + 2*(s1+s2)`

What did you say!?!

strings: *textual* data

Given
```
s1 = 'ha'
s2 = 't'
```

What are `s1 + s2`

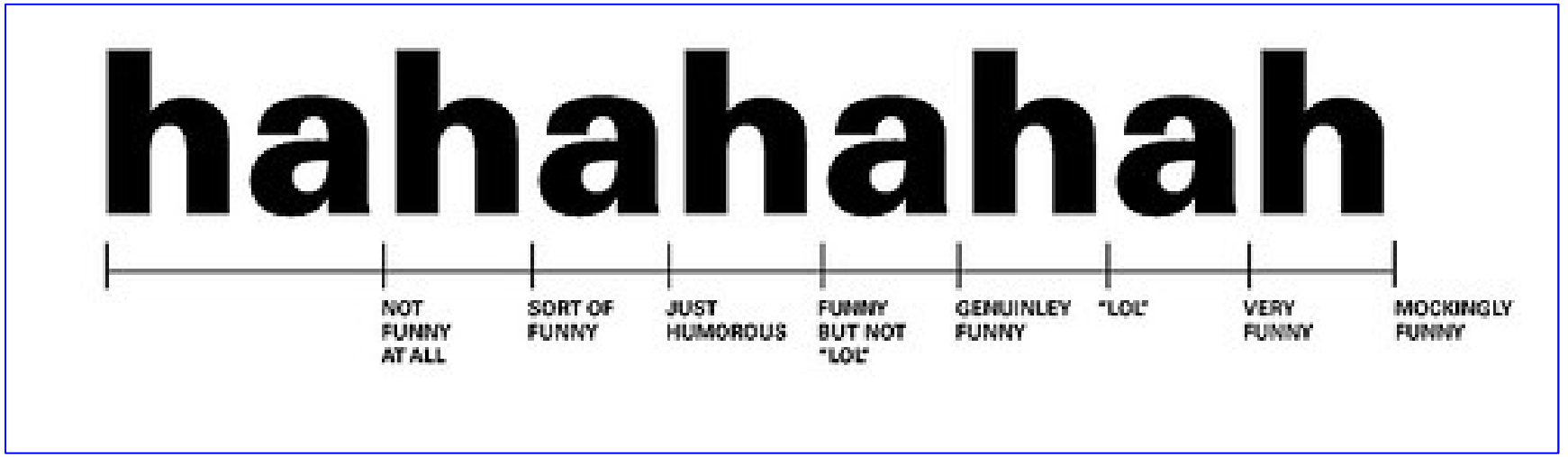
hat

```
2*s1 + s2 + 2*(s1+s2)
```

hahathathat

What did you say!?!

hahahahah



LoL!

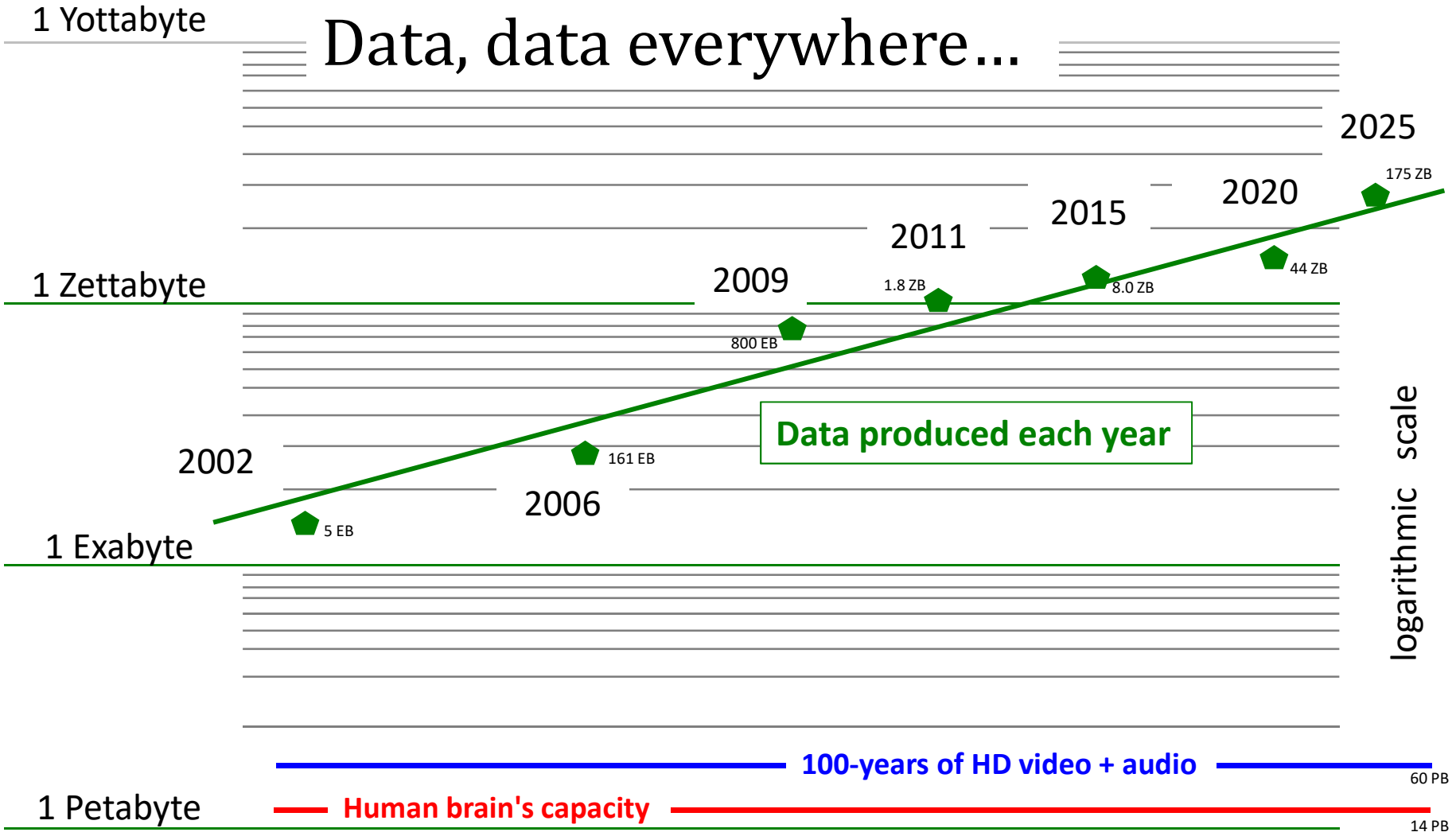
Pass those to the East!

LoL!

Data, data everywhere…

Data

Data, data everywhere…



1 Petabyte, PB == 1000 Terabytes, TB

1 Terabyte, TB == 1000 Gigabytes, GB

References

(2020) 44ZB: http://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm
(2015) 8 ZB: http://www.emc.com/collateral/analyst-reports/idc-extracting-value-from-chaos-ar.pdf
(2011) 1.8 ZB: http://www.emc.com/leadership/programs/digital-universe.htm
(2009) 800 EB: http://www.emc.com/collateral/analyst-reports/idc-digital-universe-are-you-ready.pdf
(2006) 161 EB: http://www.emc.com/collateral/analyst-reports/expanding-digital-idc-white-paper.pdf
(2002) 5 EB: http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/execsum.htm
(2019) www.networkworld.com/article/3325397/idc-expect-175-zettabytes-of-data-worldwide-by-2025.html
(life in video) 60 PB: in 4320p resolution, extrapolated from 16MB for 1:21 of 640x480 video (w/sound) – almost certainly a gross overestimate, as sleep can be compressed significantly!
(brain) 14 PB: http://www.quora.com/Neuroscience-1/How-much-data-can-the-human-brain-store

Big *Data?*

Big data: The next frontier for innovation, competition, and productivity

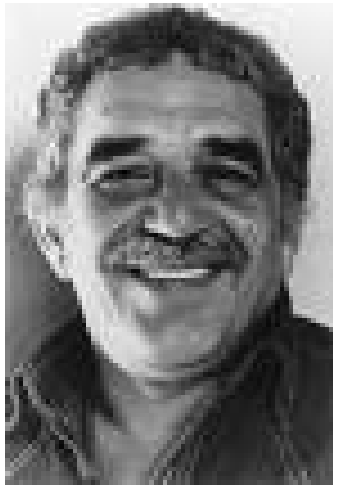
The New York Times

SundayReview | The Opinion Pages

WORLD U.S. N.Y. / REGION BUSINESS TECHNOLOGY SCIENC

NEWS ANALYSIS

Is Big Data an Economic Big Dud?



G. Garcia Marquez

wisdom

G.G.M, et al.

knowledge

Google's users

information

Google

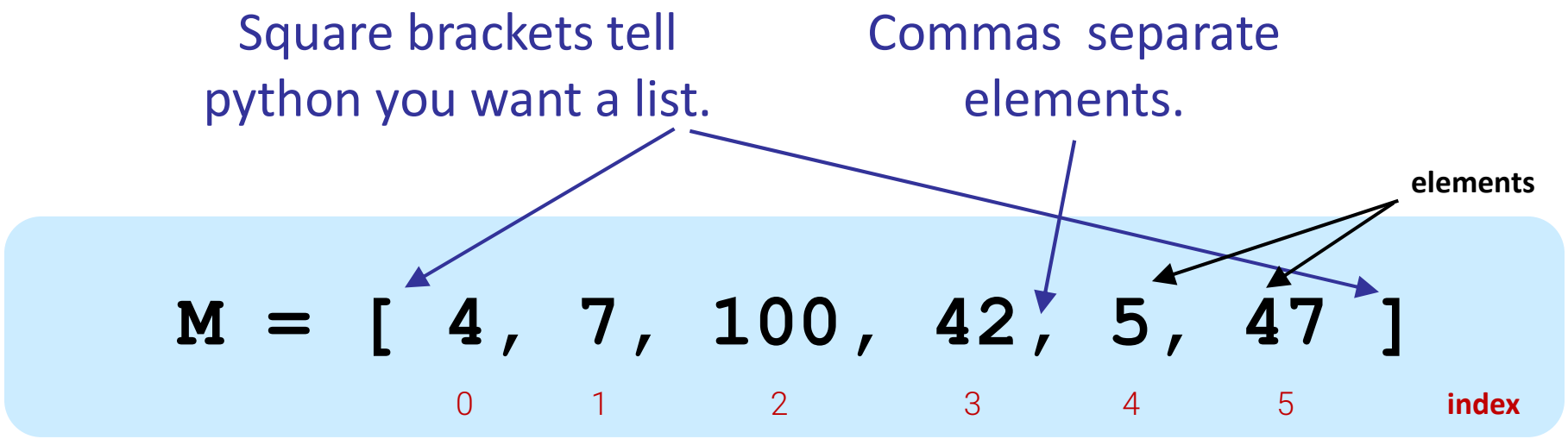
data

Data's *elevation*?

Lists ~ collections of *any* data

```
M = [ 4, 7, 100, 42, 5, 47 ]
```

Lists ~ collections of *any* data



`len(M)`	`M[0]`	`M[0:3]`
top-level length	indexing	slicing

Lists ~ collections of *any* data

string

```
L = [ 3.14, [2,40], 'third', 42 ]
```

`len(L)`	`L[0]`	`L[0:1]`
top-level length	**indexing**	**slicing**
*only counts **top-level** elements*	***could** return a different type*	***always** returns the same type, and **always** returns a **sub**structure!*

Indexing uses `[ ]` *Strings*

```
s = 'harvey mudd college'
```

```
    0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

Indexing uses [ ] *Strings*

```
s = 'harvey mudd college'
    0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

Some **German words** are so long that they have a perspective. For example,

- Freundschaftsbezeigungen.
- Dilettantenaufdringlichkeiten.
- Stadtverordnetenversammlungen.

These things are not words, they are **alphabetical processions**.

- Mark Twain

Indexing uses [ ]

for strings, too

```
s = 'harvey mudd college'
    0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

index

`s[0]` is `'h'` Read as "s-of-zero" or "s-zero"

`s[17]` is

`s[6]` is

`s[ ]` is `'e'`

Negative indices…

In a negative mood ?
Python's there for you !

```
          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17 18
s =     'harvey mudd college'
       -19 -17 -15 -13 -11 -9  -7  -5  -3  -1
          -18 -16 -14 -12 -10 -8  -6  -4  -2
```

Negative indices count ***backwards*** from the end!

`s[-1]` is `'e'`

`s[-18]` is

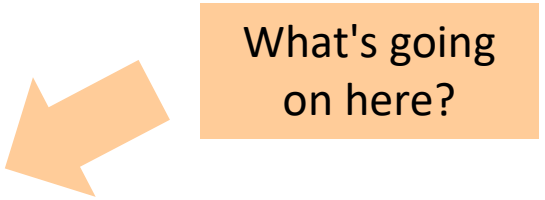
`s[-7]` is

`s[-0]` is

Slicing

```
s = 'harvey mudd college'
    0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

`s[ : ]` ***slices*** the string, returning a substring



`s[0:6]` is `'harvey'`

`s[12:18]` is `'colleg'`

`s[17:]` is `'ge'`

`s[:]` is `'harvey mudd college'`

Slicing

```
s = 'harvey mudd college'
    0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

start end+1

`s[ : ]` ***slices*** the string, returning a substring

first index is the first character

second index is **ONE AFTER** the last character

a missing index means ***that end*** of the string

`s[0:6]` is `'harvey'`

`s[12:18]` is `'colleg'`

`s[17:]` is `'ge'`

`s[:]` is `'harvey mudd college'`

Slicing

```
          0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17 18
s = 'harvey mudd college'
   -19 -18 -17 -16 -15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1
```

What are these slices?

start end+1

`s[15:-1]` is

`s[:2]` is

and these?

is `'mud'`

is `'e'`

Skip-Slicing

`s[ start : end+1 : ]` ← the third index is the **stride length**

default is +1

```
s = 'harvey mudd college'
    0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

`s[2:11:2]` is `'re ud'`

Raji!

`s[17:12]` is

`s[17:12:-1]` is

is `'doe'`

`s[::-1]` is

`s[1::6]` is

I love this one.

- G. Garcia Marquez

```
pi = [3,1,4,1,5,9]

L = [ 'pi', "isn't", [4,2] ]

M = 'You need parentheses for chemistry !'
    0   4   8   12  16  20  24  28  32
```

Try it!

Part 1

What is `len(pi)` 6

What is `len(L)`

What is `len(L[1])`

What is `pi[2:4]`

What slice of `pi` is `[3,1,4]` pi[0:3]

What slice of `pi` is `[3,4,5]`

Part 2

What is `L[0]` 'pi'

What is `L[0][1]`

What is `L[0:1]`

These three are all different

What slice of `M` is `'try'`?

What slice of `M` is `'shoe'`?

What is `M[9:15]`

What is `M[::5]`

Extra! Mind Muddler

What are `pi[0]*(pi[1]+pi[2])` and `pi[0]*(pi[1:2]+pi[2:3])` ?

These two are different!

```
pi = [3,1,4,1,5,9]

L = [ 'pi', "isn't", [4,2] ]

M = 'You need parentheses for chemistry !'
    0   4   8   12  16  20  24  28  32
```




We ♥ ( )

Part 1

What is `len(pi)` 6

What is `len(L)`

What is `len(L[1])`

What is `pi[2:4]`

What slice of `pi` is `[3,1,4]` pi[0:3]

What slice of `pi` is `[3,4,5]`

Part 2

What is `L[0]` 'pi'

What is `L[0][1]`

What is `L[0:1]`

These three are all different

What slice of `M` is `'try'`?

What slice of `M` is `'shoe'`?

What is `M[9:15]`

What is `M[::5]`

Extra! Mind Muddler

What are `pi[0]*(pi[1]+pi[2])` and `pi[0]*(pi[1:2]+pi[2:3])` ?

These two are different! ➡

```
pi = [3,1,4,1,5,9]

L = [ 'pi', "isn't", [4,2] ]

M = 'You need parentheses for chemistry !'
    0   4   8   12  16  20  24  28  32
```

Try it!



We ♥ ( )

Part 1

What is `len(pi)` 6

What is `len(L)` 3

What is `len(L[1])`

What is `pi[2:4]` [4,1]

What slice of `pi` is `[3,1,4]` pi[:3]

What slice of `pi` is `[3,4,5]` pi[::2]

Part 2

What is `L[0]` 'pi'

What is `L[0][1]` 'i'

What is `L[0:1]` ['pi']

These three are all different

What slice of `M` is `'try'`? M[31:34] *or* M[-5:-2]

What slice of `M` is `'shoe'`?

What is `M[9:15]` 'parent'

What is `M[::5]`

Extra! Mind Muddlers

What are `pi[0]*(pi[1]+pi[2])` and `pi[0]*(pi[1:2]+pi[2:3])` ?

These two are different! ➡

15

[1,4,1,4,1,4]

Python slices - it dices...

(**data**, at least)

... *but wait,* there's more!

Python slices - it dices...

(**data**, at least)

Python **functions**

... *but wait,* there's more!

Functioning in Python

```
# my own function!
def dbl( x ):
    """ returns double its input, x """
    return 2x
```

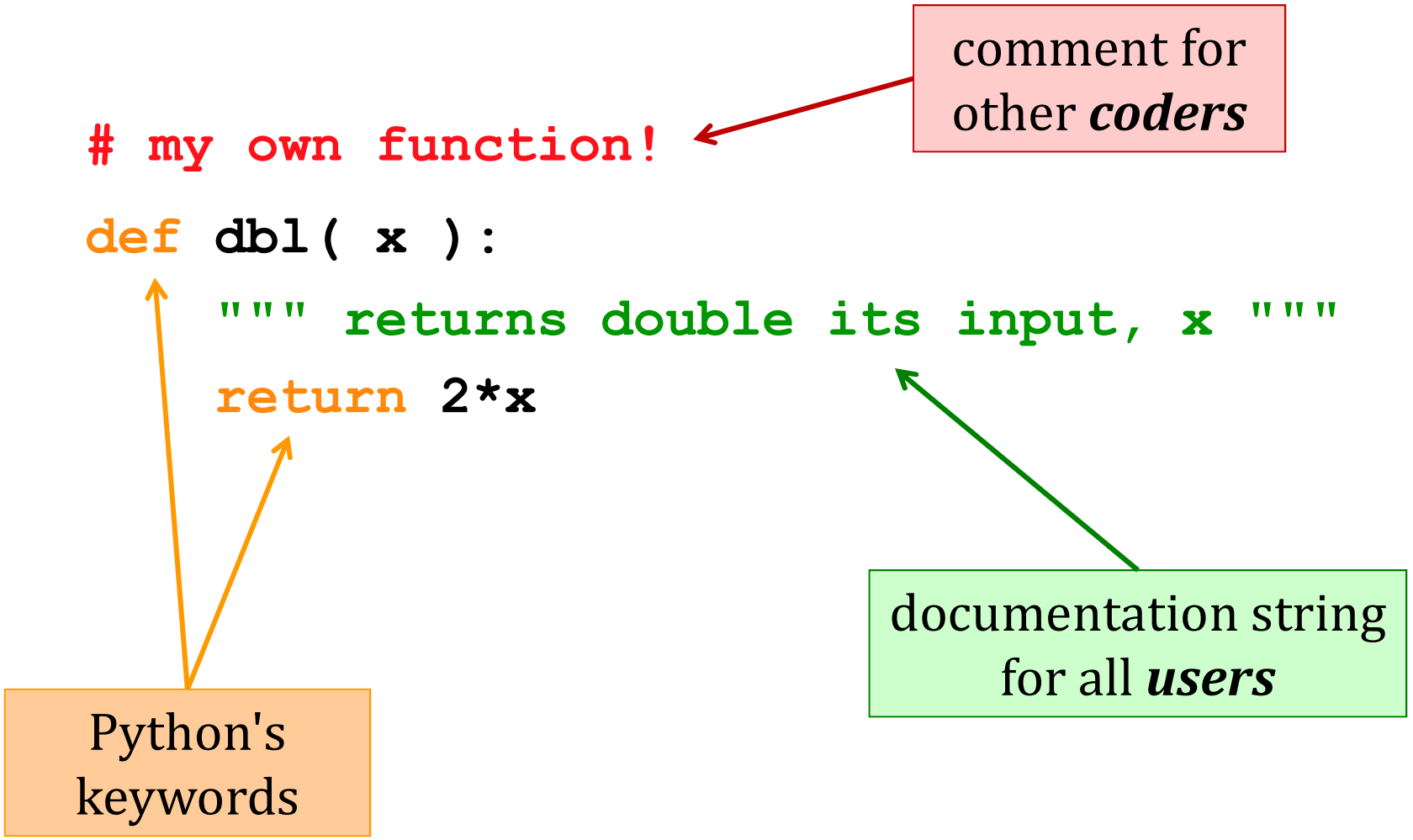
This doesn't ***look*** quite right...

Functioning in Python

```
1  #
2  # Putting the "fun" into Python functions!
3  #
4
5  def dbl( x ):
6      """ returns double its input, x """
7      return 2x
8
```

Still broken… !

Functioning in Python



Some of Python's *baggage*...

Function *Fun* !

```
def undo(s):
    """ this "undoes" its input, s """
    return 'de' + s
```

```
>>> undo('caf')
'decaf'

>>> undo(undo('caf'))
```

strings, lists, numbers ...
*all **data** are fair game*

Have a dedecaf-ternoon!

morning + evening, too

Just ununundo it!

This week's lab ~

*first **two** hw problems*