Smith College
Computer Science

CSC 111 Introduction to Computer Science

Spring 2018 — Week 2

Dominique Thiébaut
dthiebaut@smith.edu

Outline

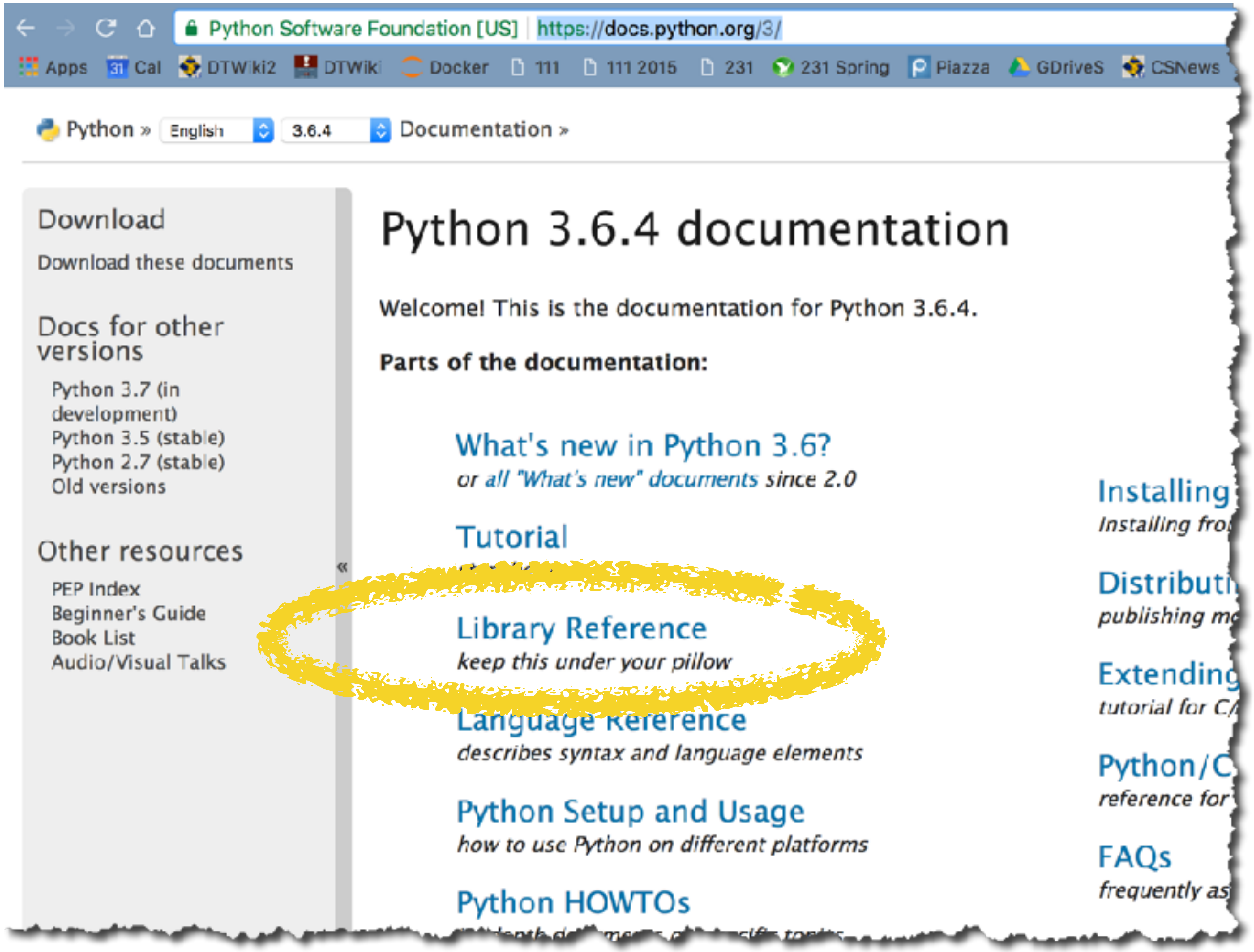
- Moodle Access
- Piazza
- Homework partners
- Loops + **range**()
- Input
- Programming Process

```
for <var> in <sequence>:
     <body>



for name in [ "Alex", "Max", "Rui" ]:
     print( "Conversation with", name )
     call( name )
     chatWith( name )
     sayGoodByeTo( name )
```

```
for <var> in <sequence>:
      <body>
```

```
for x in range( 10 ):
      print( x )
```

http://docs.python.org/3/


Python Software Foundation [US] https://docs.python.org/3/

Apps Cal DTWiki2 DTWiki Docker 111 111 2015 231 231 Spring Piazza GDriveS CSNews

Python » English 3.6.4 Documentation »

Download

Download these documents

Docs for other versions

Python 3.7 (in development)
Python 3.5 (stable)
Python 2.7 (stable)
Old versions

Other resources

PEP Index
Beginner's Guide
Book List
Audio/Visual Talks

Python 3.6.4 documentation

Welcome! This is the documentation for Python 3.6.4.

Parts of the documentation:

What's new in Python 3.6?
or all "What's new" documents since 2.0

Tutorial

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python Setup and Usage
how to use Python on different platforms

Python HOWTOs

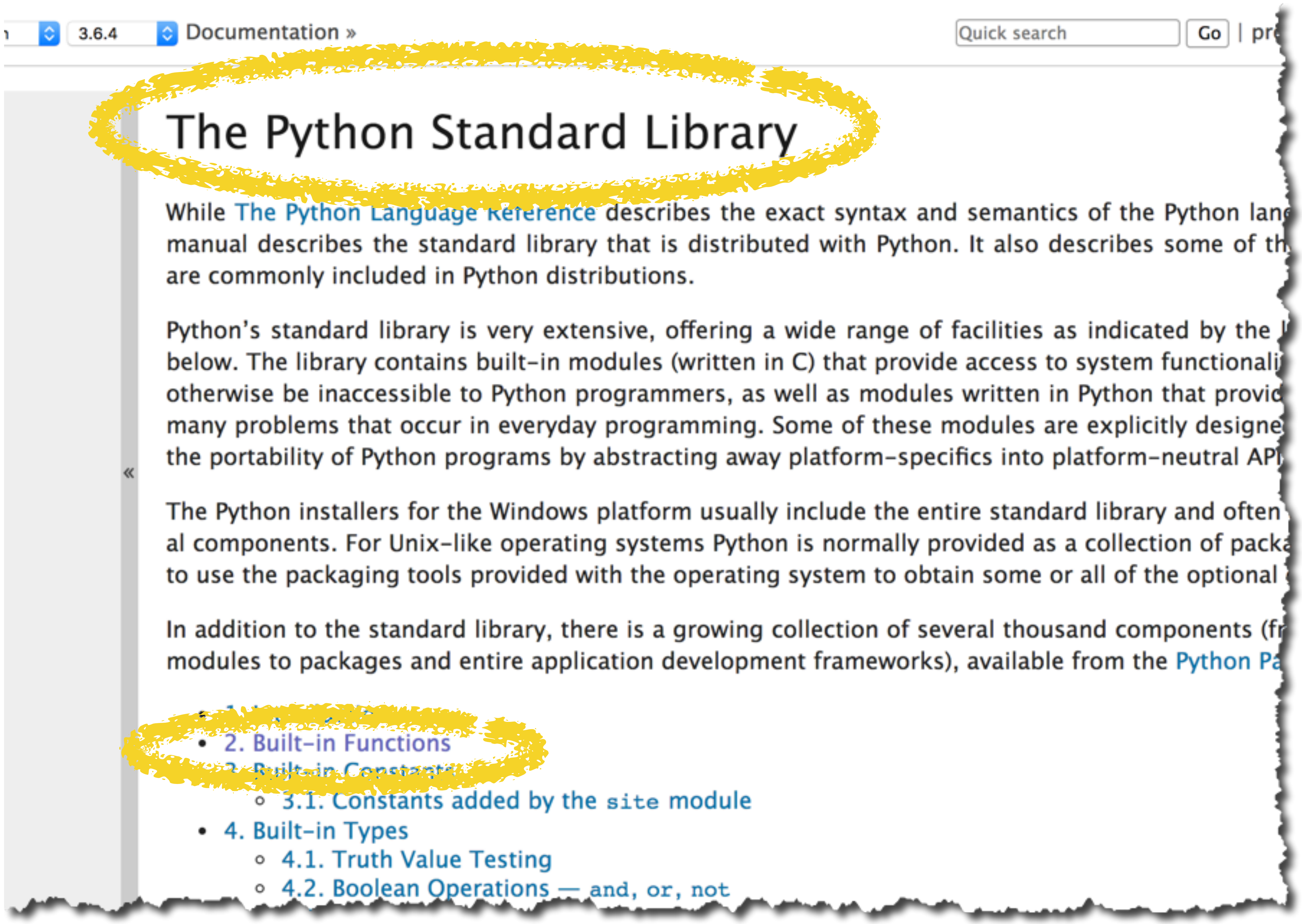
Installing
Installing fro

Distributi
publishing m

Extending
tutorial for C/

Python/C
reference for

FAQs
frequently as



3.6.4 Documentation »

Quick search Go | pr

The Python Standard Library

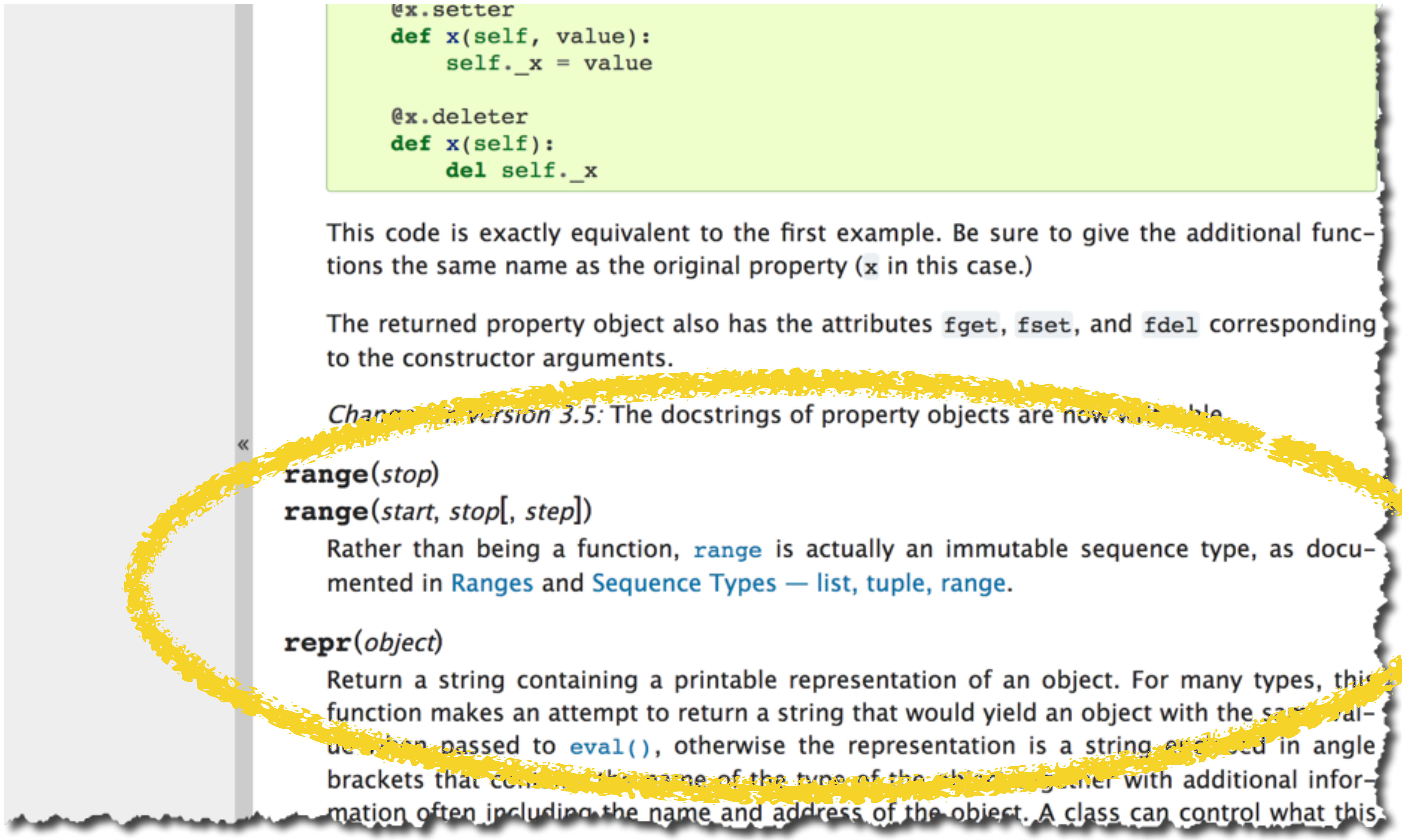
While The Python Language Reference describes the exact syntax and semantics of the Python lan
manual describes the standard library that is distributed with Python. It also describes some of th
are commonly included in Python distributions.

Python's standard library is very extensive, offering a wide range of facilities as indicated by the l
below. The library contains built-in modules (written in C) that provide access to system functionali
otherwise be inaccessible to Python programmers, as well as modules written in Python that provid
many problems that occur in everyday programming. Some of these modules are explicitly designe
the portability of Python programs by abstracting away platform-specifics into platform-neutral API

The Python installers for the Windows platform usually include the entire standard library and often
al components. For Unix-like operating systems Python is normally provided as a collection of pack
to use the packaging tools provided with the operating system to obtain some or all of the optional

In addition to the standard library, there is a growing collection of several thousand components (fr
modules to packages and entire application development frameworks), available from the Python Pa

- 1. Introduction
- 2. Built-in Functions
- 3. Built-in Constants
  - 3.1. Constants added by the `site` module
- 4. Built-in Types
  - 4.1. Truth Value Testing
  - 4.2. Boolean Operations — `and`, `or`, `not`



```
    @x.setter
    def x(self, value):
        self._x = value

    @x.deleter
    def x(self):
        del self._x
```

This code is exactly equivalent to the first example. Be sure to give the additional functions the same name as the original property (`x` in this case.)

The returned property object also has the attributes `fget`, `fset`, and `fdel` corresponding to the constructor arguments.

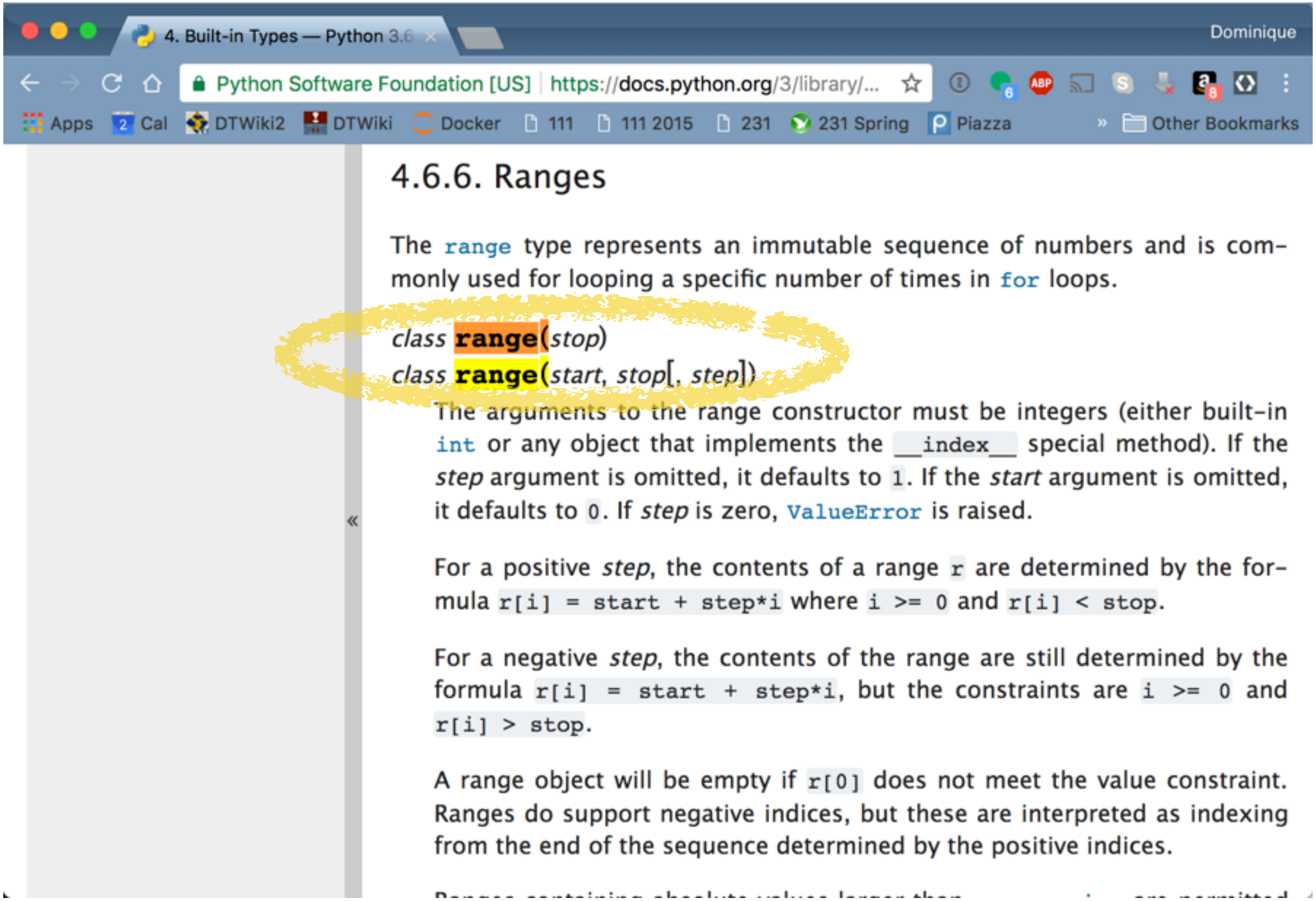
Changed in version 3.5: The docstrings of property objects are now writeable.

range(*stop*)
range(*start*, *stop*[, *step*])

Rather than being a function, `range` is actually an immutable sequence type, as documented in Ranges and Sequence Types — list, tuple, range.

repr(*object*)

Return a string containing a printable representation of an object. For many types, this function makes an attempt to return a string that would yield an object with the same value when passed to `eval()`, otherwise the representation is a string enclosed in angle brackets that contains the name of the type of the object together with additional information often including the name and address of the object. A class can control what this


4. Built-in Types — Python 3.6
Dominique
Python Software Foundation [US] https://docs.python.org/3/library/...
Apps
Cal
DTWiki2
DTWiki
Docker
111
111 2015
231
231 Spring
Piazza
Other Bookmarks

4.6.6. Ranges

The `range` type represents an immutable sequence of numbers and is commonly used for looping a specific number of times in `for` loops.

class **range**(*stop*)
class **range**(*start*, *stop*[, *step*])

The arguments to the range constructor must be integers (either built-in `int` or any object that implements the `__index__` special method). If the *step* argument is omitted, it defaults to `1`. If the *start* argument is omitted, it defaults to `0`. If *step* is zero, `ValueError` is raised.

For a positive *step*, the contents of a range `r` are determined by the formula `r[i] = start + step*i` where `i >= 0` and `r[i] < stop`.

For a negative *step*, the contents of the range are still determined by the formula `r[i] = start + step*i`, but the constraints are `i >= 0` and `r[i] > stop`.

A range object will be empty if `r[0]` does not meet the value constraint. Ranges do support negative indices, but these are interpreted as indexing from the end of the sequence determined by the positive indices.

Examples to Try Out:

```
for x in range( … ):        # replace … with
    print( x )              # range expression
                            # below:

# range( 10 )
# range( 2, 10 )
# range( -5, 5 )
# range( 0, 10, 2 )
# range( 0, 10, 3 )
# range( 9, 0, -1 )
```

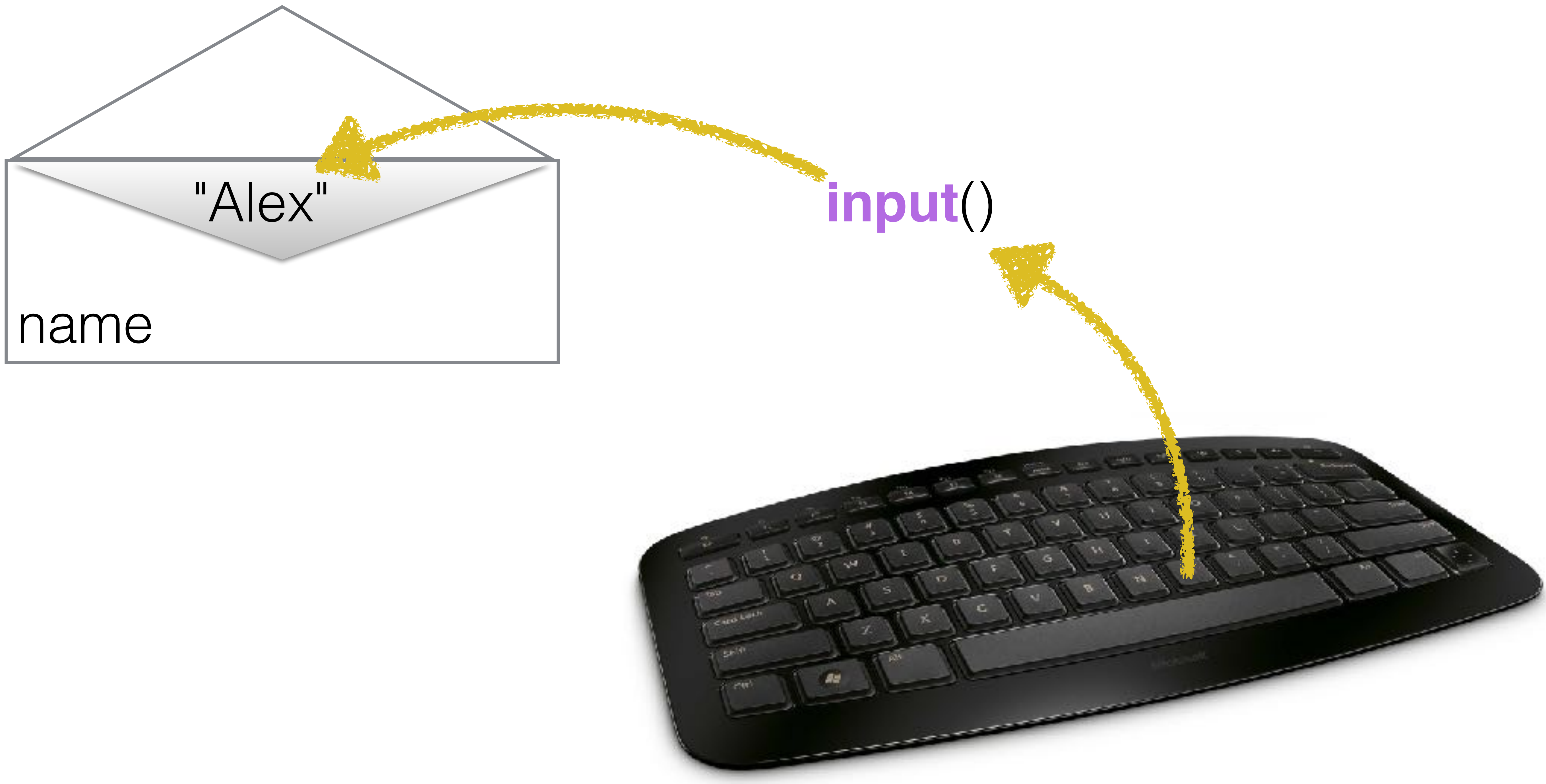
Exercise

**Generate an equivalency table of temperatures in Fahrenheit and Celsius. 100 F should be on the first line, and -30F on the last line.
Show only Fahrenheit temperatures that are multiples of 10.**

Celsius = (Farhenheit - 32) * 5 / 9

Outline

- The Programming Process
- Variables
- Definite Loops
- **Input**

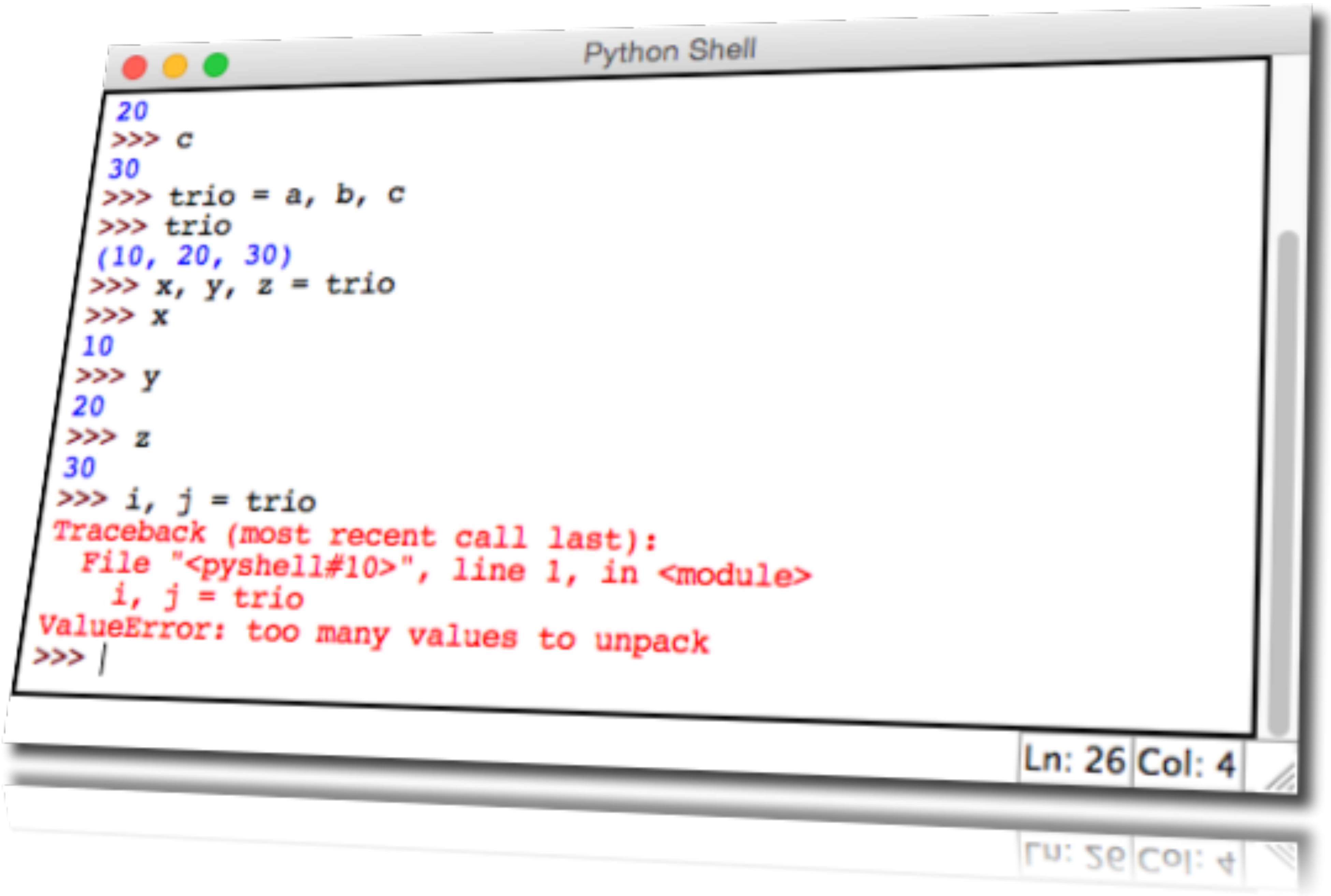


```
name = input( "What is your name? " )
```

The Default Input is Text

```
name    = input( "Enter your name: " )
college = input( "Where do you go to school? " )
nat     = input( "What is your nationality? " )

account = input( "Login: " )
passwd  = input( "Password: " )
```

Demo Time



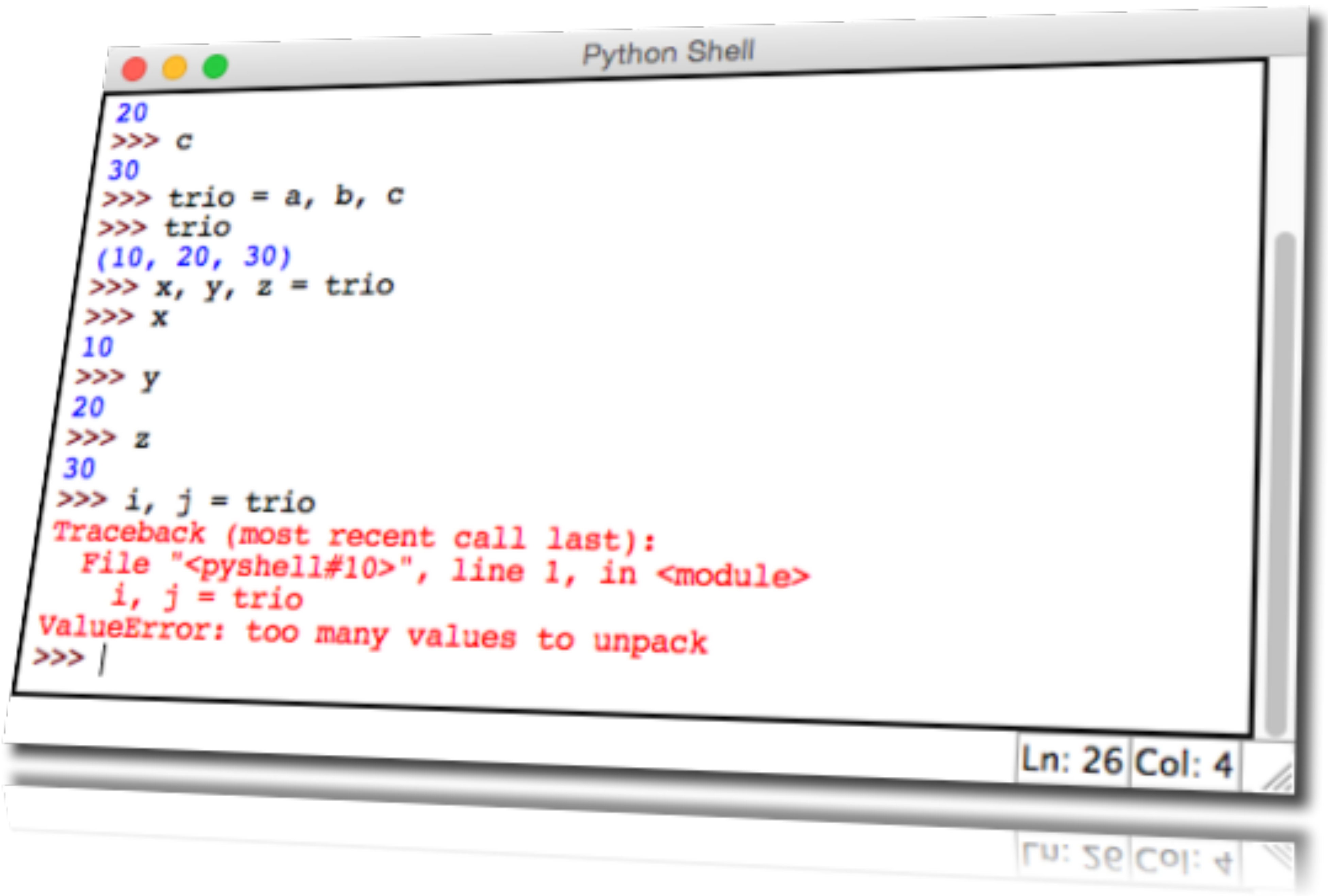
Python Shell

```
20
>>> c
30
>>> trio = a, b, c
>>> trio
(10, 20, 30)
>>> x, y, z = trio
>>> x
10
>>> y
20
>>> z
30
>>> i, j = trio
Traceback (most recent call last):
  File "<pyshell#10>", line 1, in <module>
    i, j = trio
ValueError: too many values to unpack
>>> |
```

Numbers require an Extra Step

```
age       = eval( input( "Enter your age: " ) )
salary    = eval( input( "Income in 2017? " ) )
balance = eval( input( "Account balance? " ) )
```

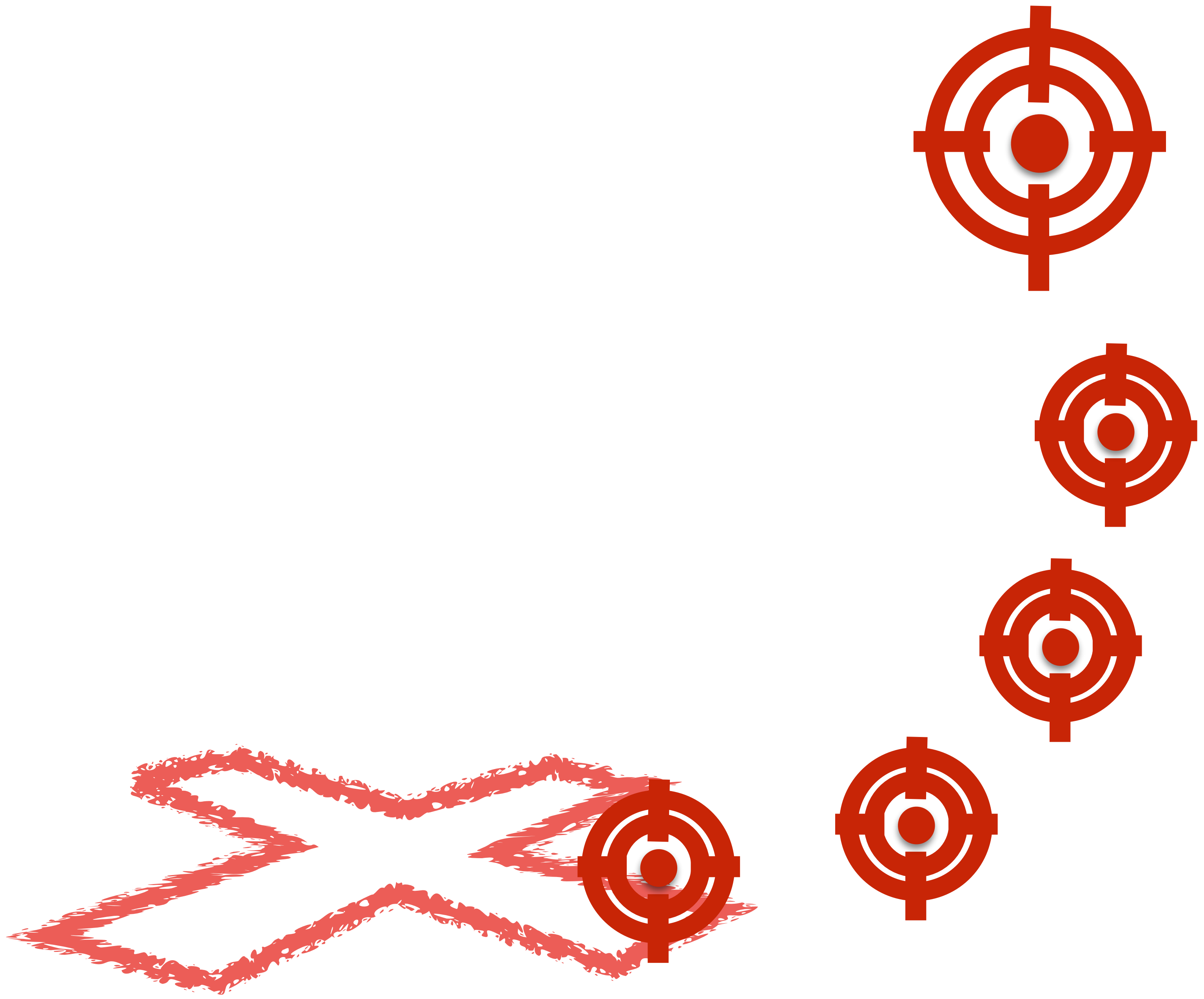
Demo Time



```
20
>>> c
30
>>> trio = a, b, c
>>> trio
(10, 20, 30)
>>> x, y, z = trio
>>> x
10
>>> y
20
>>> z
30
>>> i, j = trio
Traceback (most recent call last):
  File "<pyshell#10>", line 1, in <module>
    i, j = trio
ValueError: too many values to unpack
>>> |
```

The Programming Process
Example 1

The Programming Process
Example 1



Problem 1

Get first name, last name, Id from student,
and final grade, as a number (0-100).

Also known is class average, as a number (0-100).

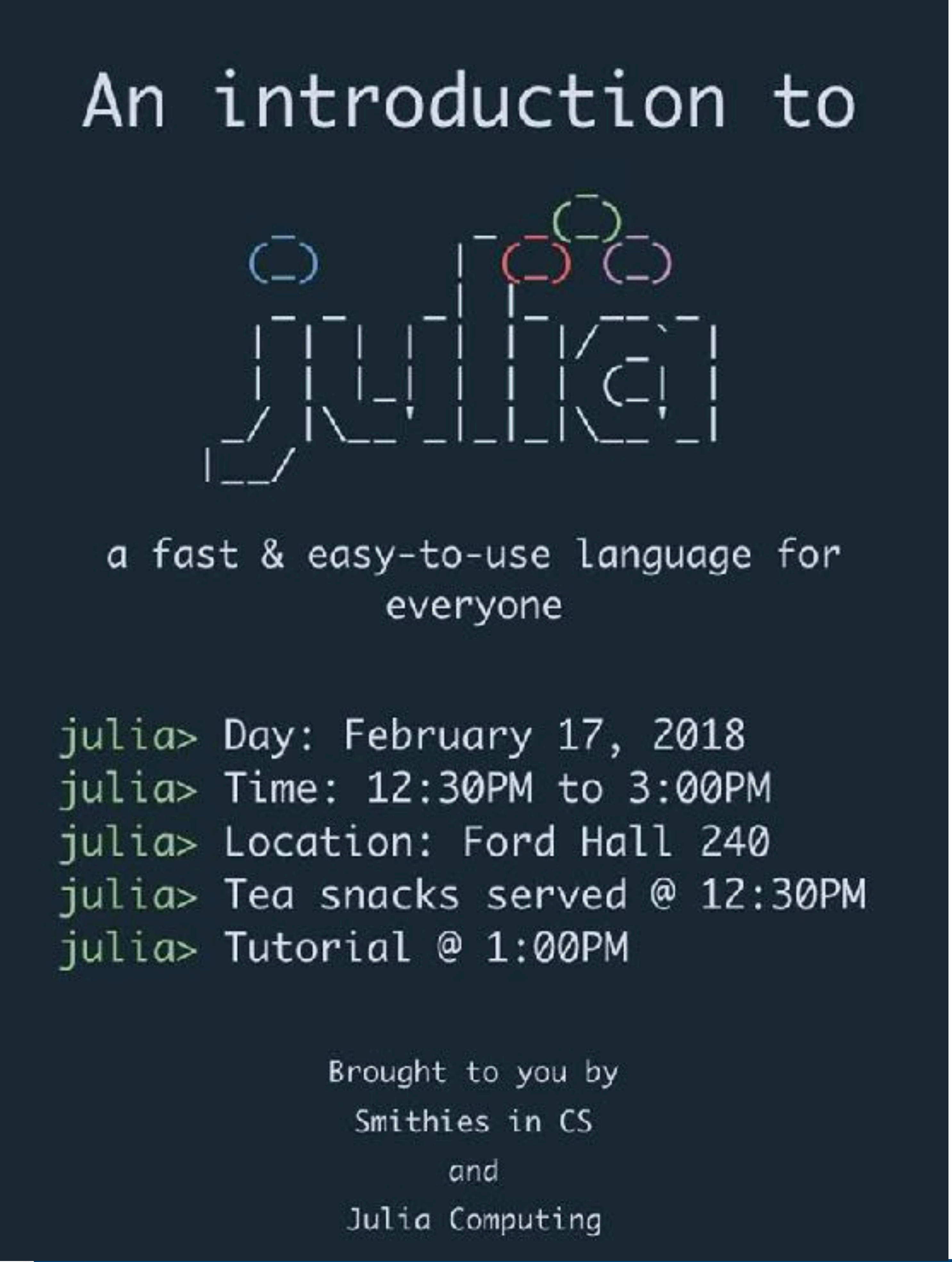
Display student information in a box,
and horizontal bar-graph of 2 grades.

Problem 1

```
First name?  Dominique
Last name?   Thiebaut
Id?          990123456
Final grade? 90

+-------------------------------------------------+
|Dominique Thiebaut                    990123456 |
+-------------------------------------------------+
      00...10...20...30...40...50...60...70...80...90...100
grade: #############################################
class: ########################################
```

We stopped here last time…

A Word from Faith Kim



An introduction to

```
               _
   _       _ _(_)_
  (_)     | (_) (_)
   _ _   _| |_  __ _
  | | | | | | |/ _` |
  | | |_| | | | (_| |
 _/ |\__'_|_|_|\__'_|
|__/
```

a fast & easy-to-use language for everyone

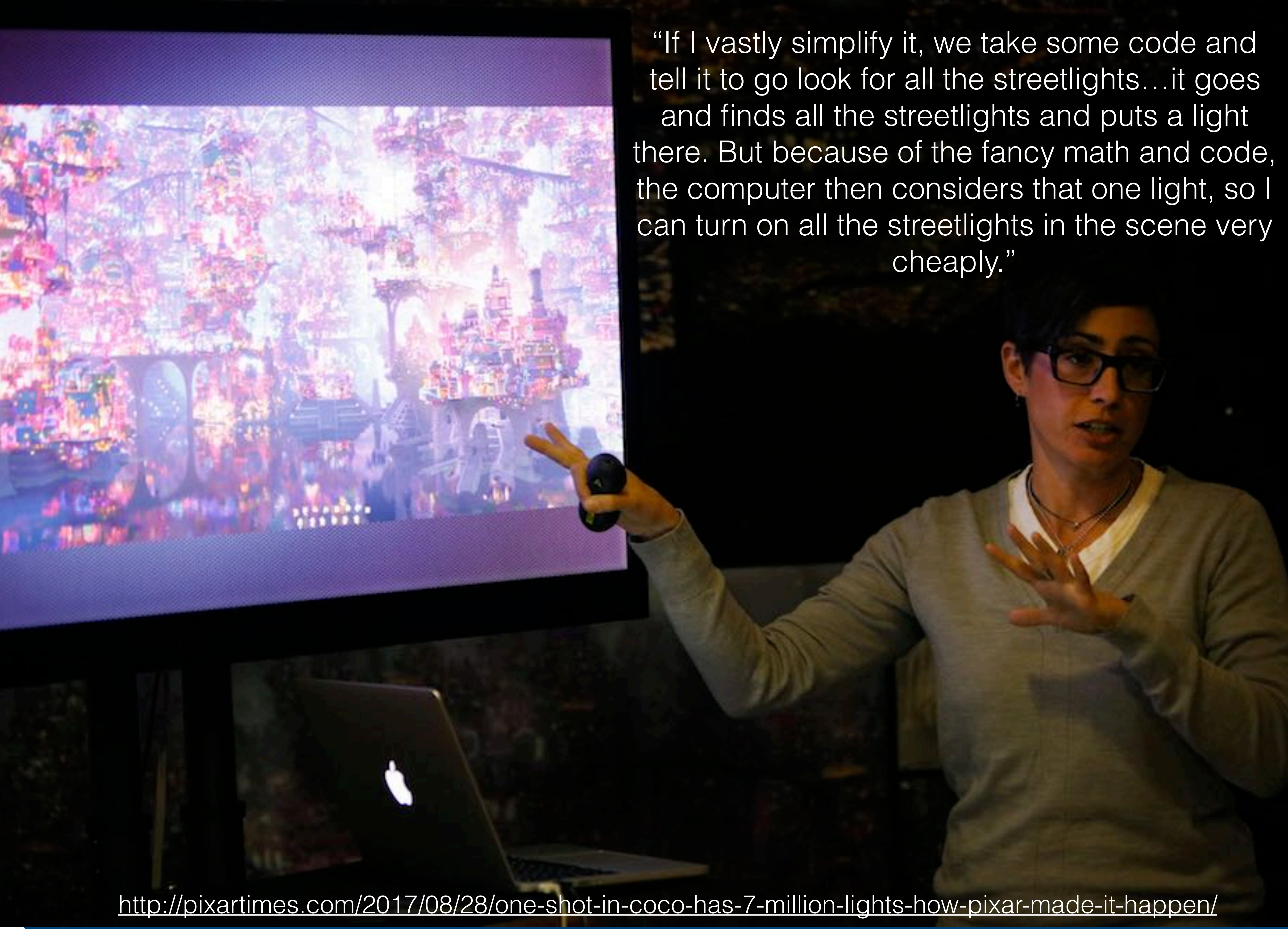
```
julia> Day: February 17, 2018
julia> Time: 12:30PM to 3:00PM
julia> Location: Ford Hall 240
julia> Tea snacks served @ 12:30PM
julia> Tutorial @ 1:00PM
```

Brought to you by
Smithies in CS
and
Julia Computing

Why is my output different in Moodle?

Coco

https://youtu.be/zNCz4mQzfEI

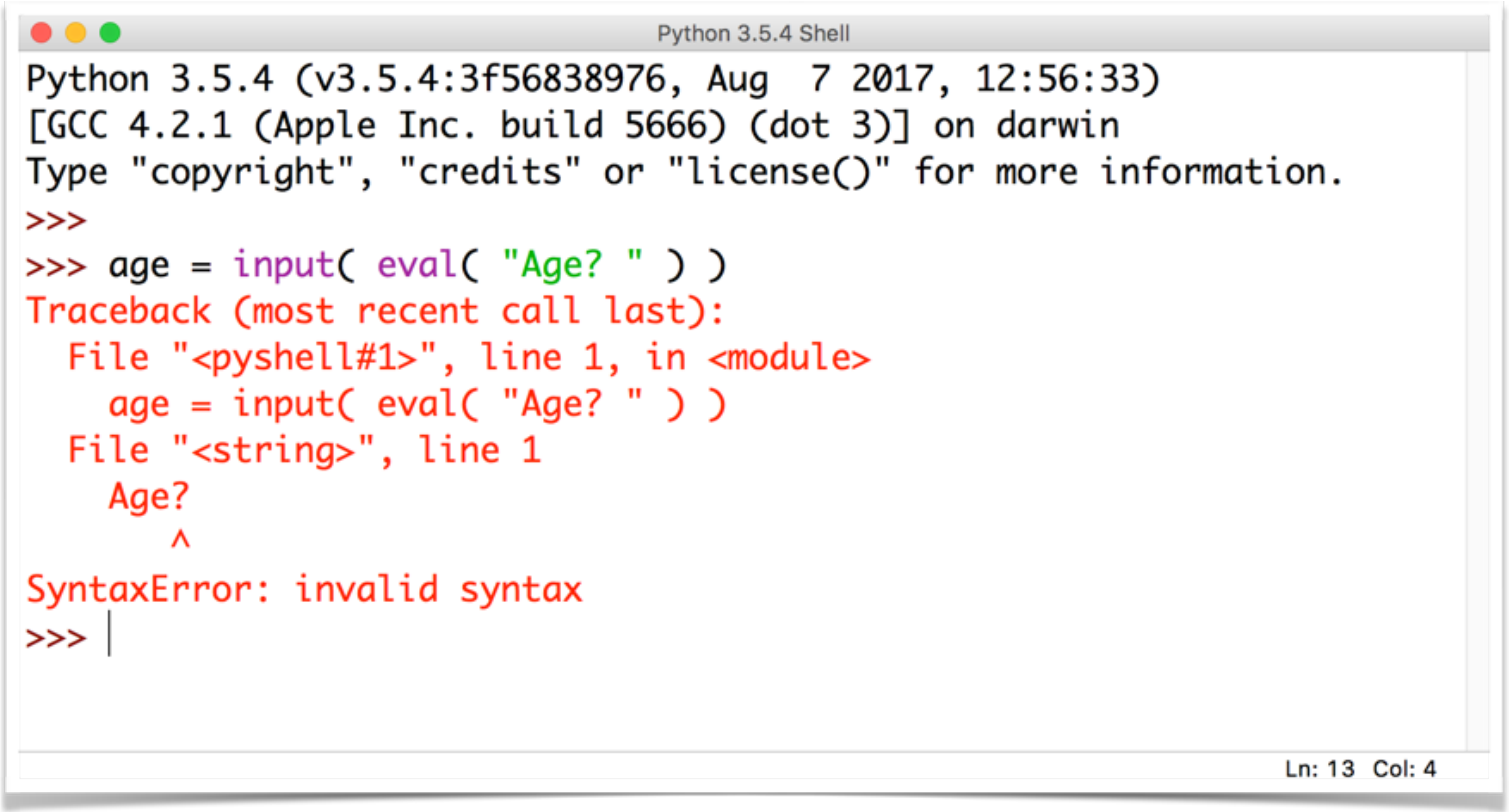


"If I vastly simplify it, we take some code and tell it to go look for all the streetlights…it goes and finds all the streetlights and puts a light there. But because of the fancy math and code, the computer then considers that one light, so I can turn on all the streetlights in the scene very cheaply."

http://pixartimes.com/2017/08/28/one-shot-in-coco-has-7-million-lights-how-pixar-made-it-happen/

Outline

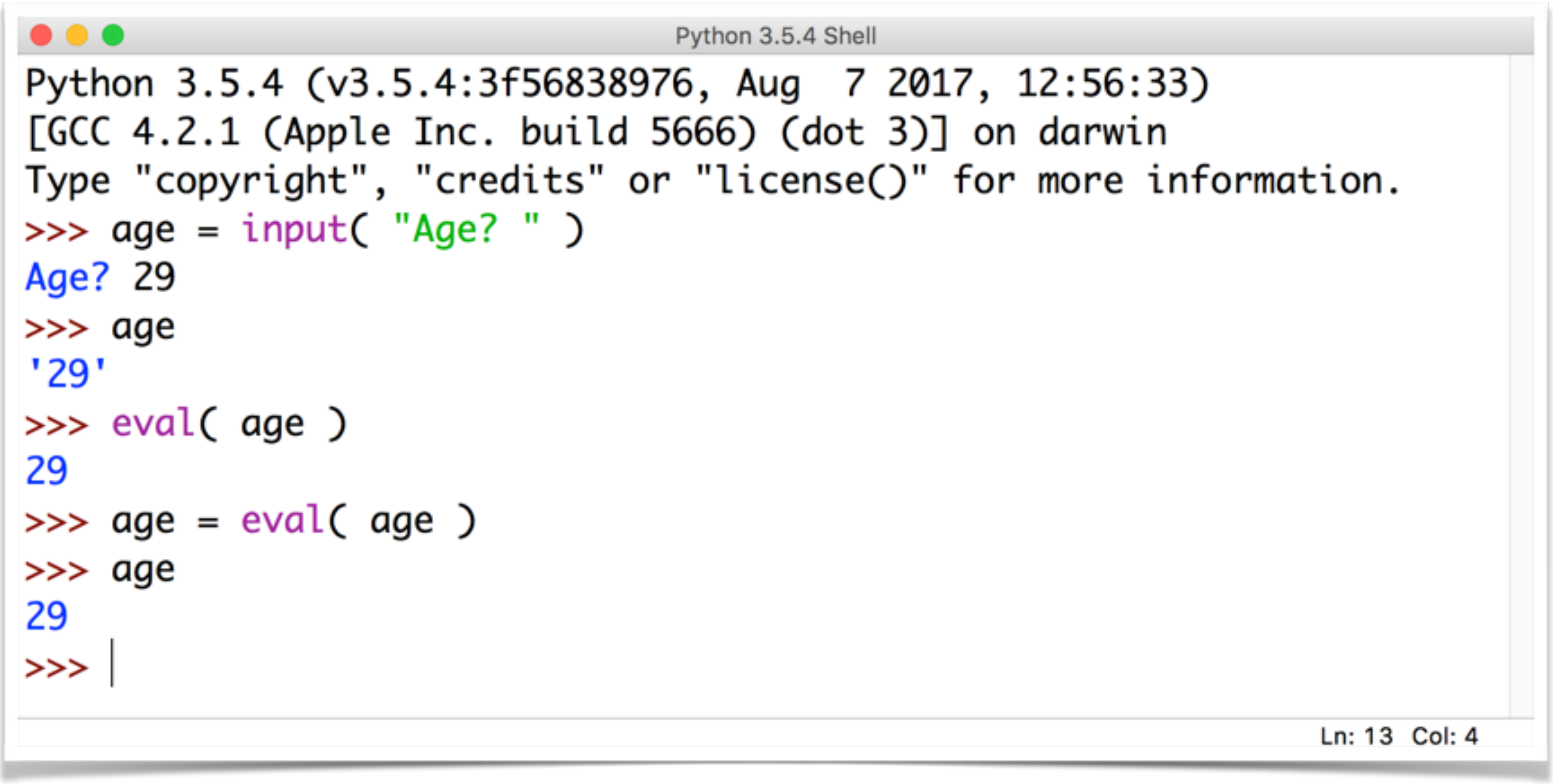
- Learning how to use **eval()**
- Programming Example 1
- Programming Example 2



Python 3.5.4 Shell

```
Python 3.5.4 (v3.5.4:3f56838976, Aug  7 2017, 12:56:33)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>>
>>> age = input( eval( "Age? " ) )
Traceback (most recent call last):
  File "<pyshell#1>", line 1, in <module>
    age = input( eval( "Age? " ) )
  File "<string>", line 1
    Age?
       ^
SyntaxError: invalid syntax
>>> 
```

Ln: 13 Col: 4


Python 3.5.4 Shell

```
Python 3.5.4 (v3.5.4:3f56838976, Aug  7 2017, 12:56:33)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> age = input( "Age? " )
Age? 29
>>> age
'29'
>>> eval( age )
29
>>> age = eval( age )
>>> age
29
>>> 
```

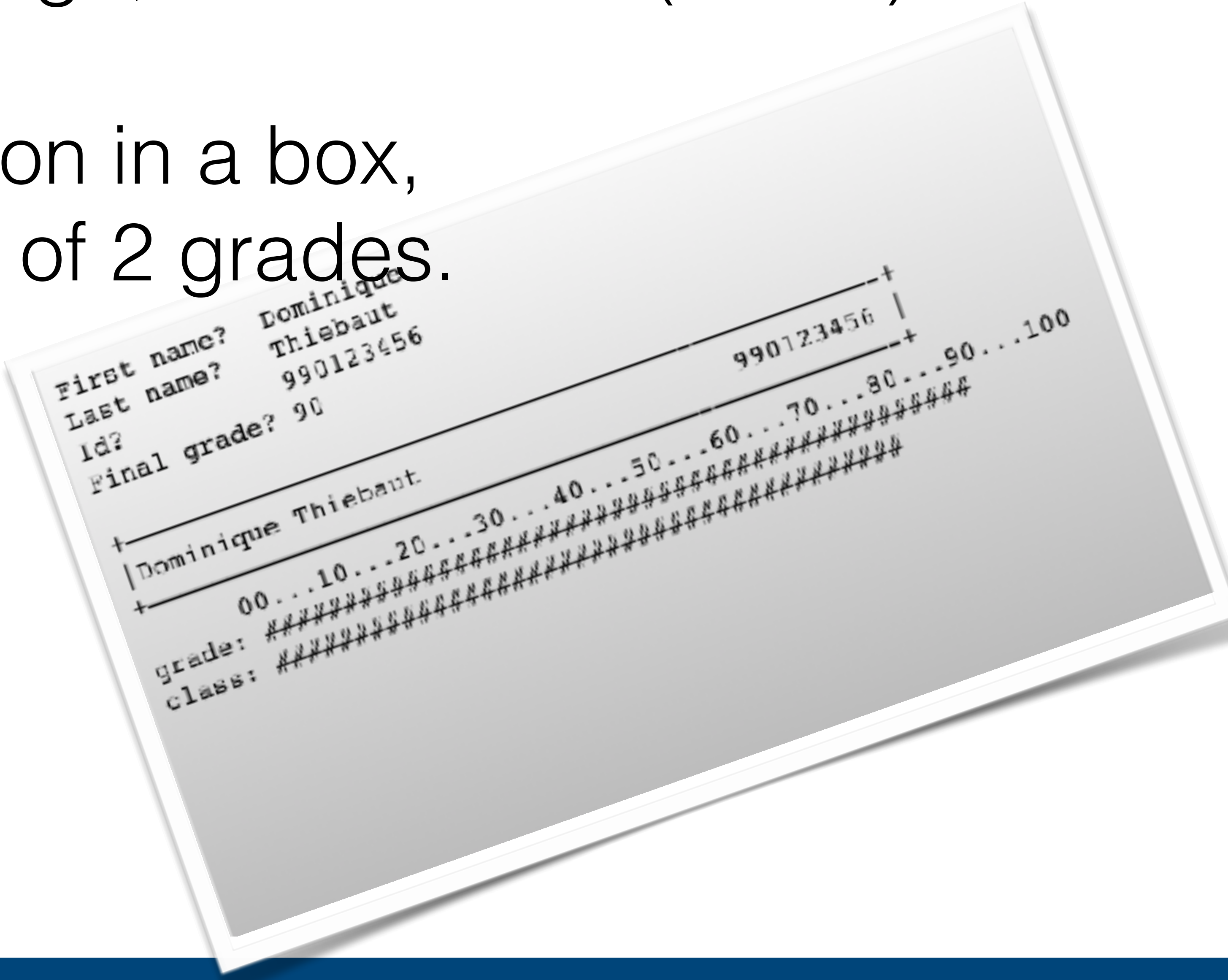
Ln: 13 Col: 4

Problem 1

Get first name, last name, Id from student,
and final grade, as a number (0-100).

Also known is class average, as a number (0-100).

Display student information in a box,
and horizontal bar-graph of 2 grades.

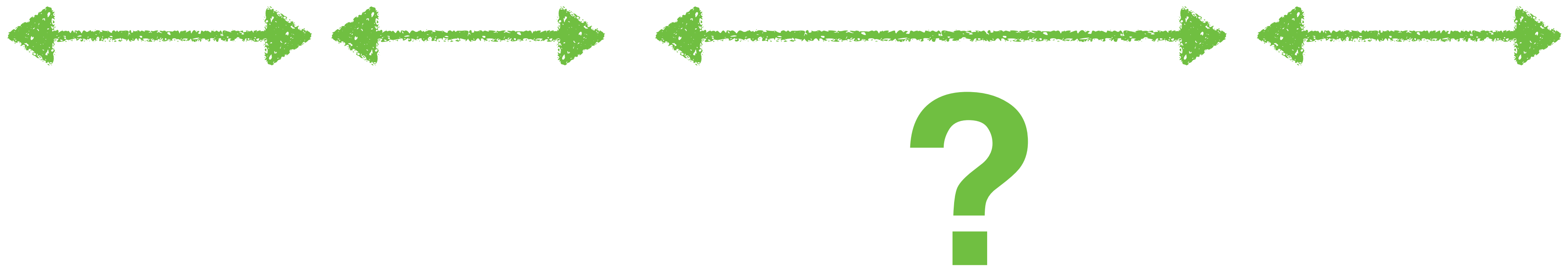


```
First name?  Dominique
Last name?   Thiebaut
Id?          990123456
Final grade? 90

+-------------------------------------------------------+
|Dominique Thiebaut                            990123456 |
+-------------------------------------------------------+
        00...10...20...30...40...50...60...70...80...90...100
grade:  #############################################
class:  ########################################
```

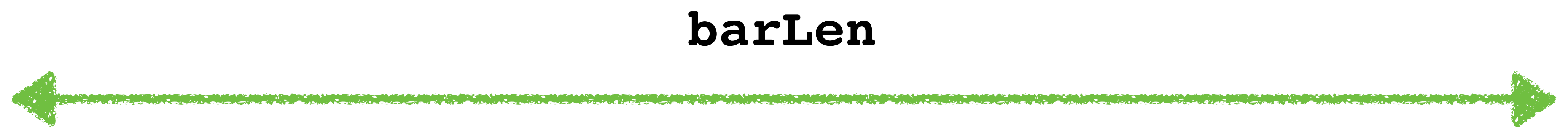
Expected Behavior

```
First name?  Dominique
Last name?   Thiebaut
Id?          990123456
Final grade? 90

+------------------------------------------------+
|Dominique Thiebaut                    990123456 |
+------------------------------------------------+
      00...10...20...30...40...50...60...70...80...90...100
grade: #############################################
class: ########################################
```

```
+------------------------------------------------+
|Dominique Thiebaut                    990123456 |
+------------------------------------------------+
```

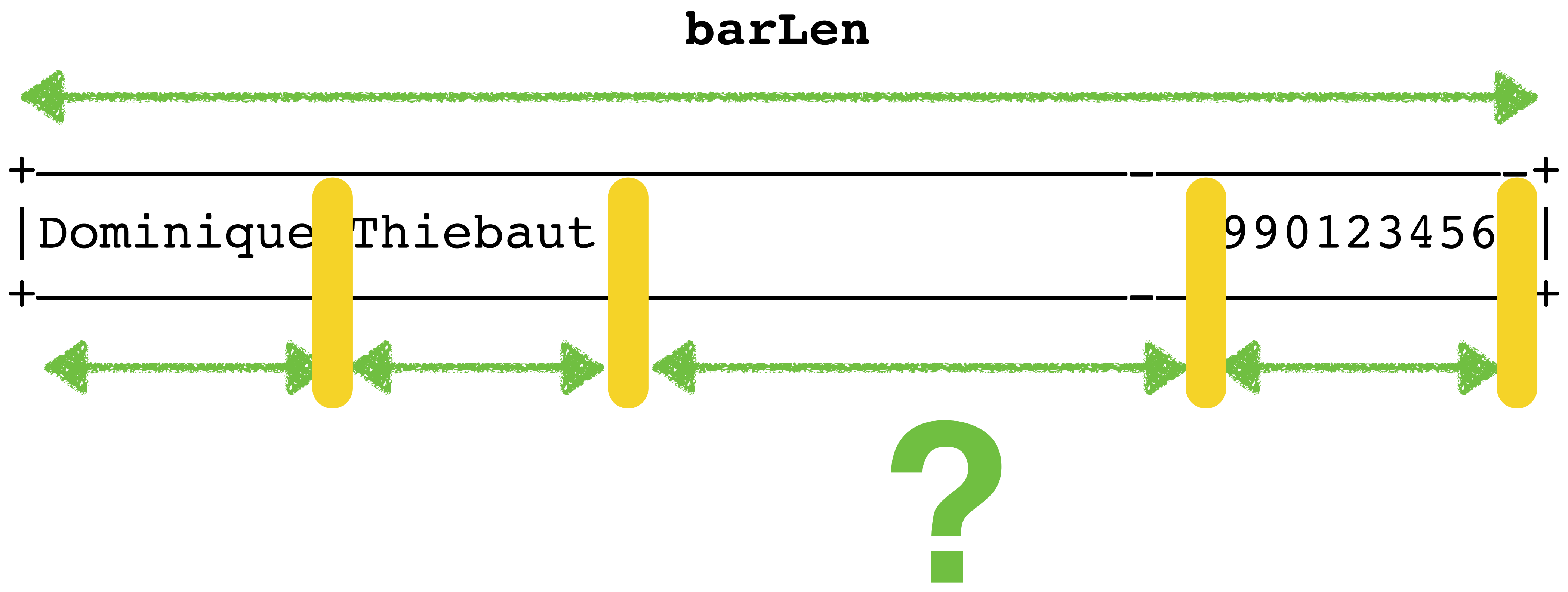


```
+----------------------------------------+---------------+
|Dominique Thiebaut              990123456 |
+----------------------------------------+---------------+
```

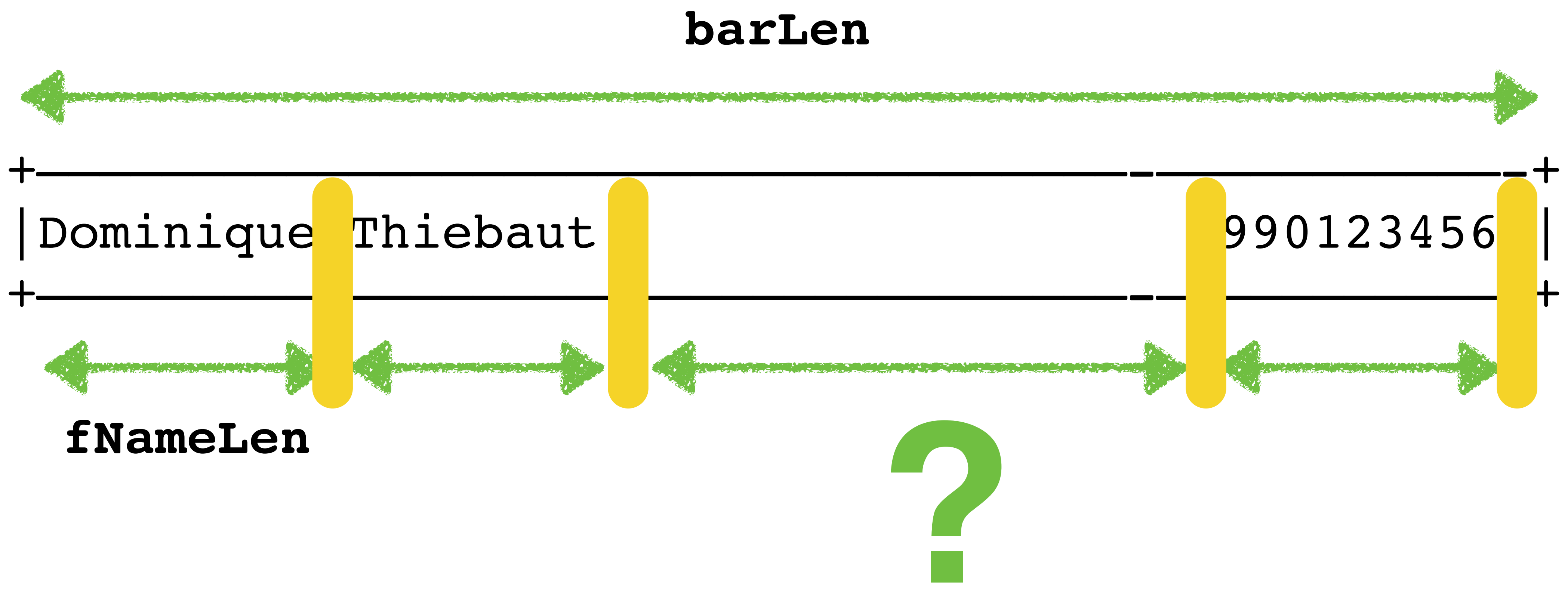


```
+------------------------------------------------+
|Dominique Thiebaut                    990123456 |
+------------------------------------------------+
```

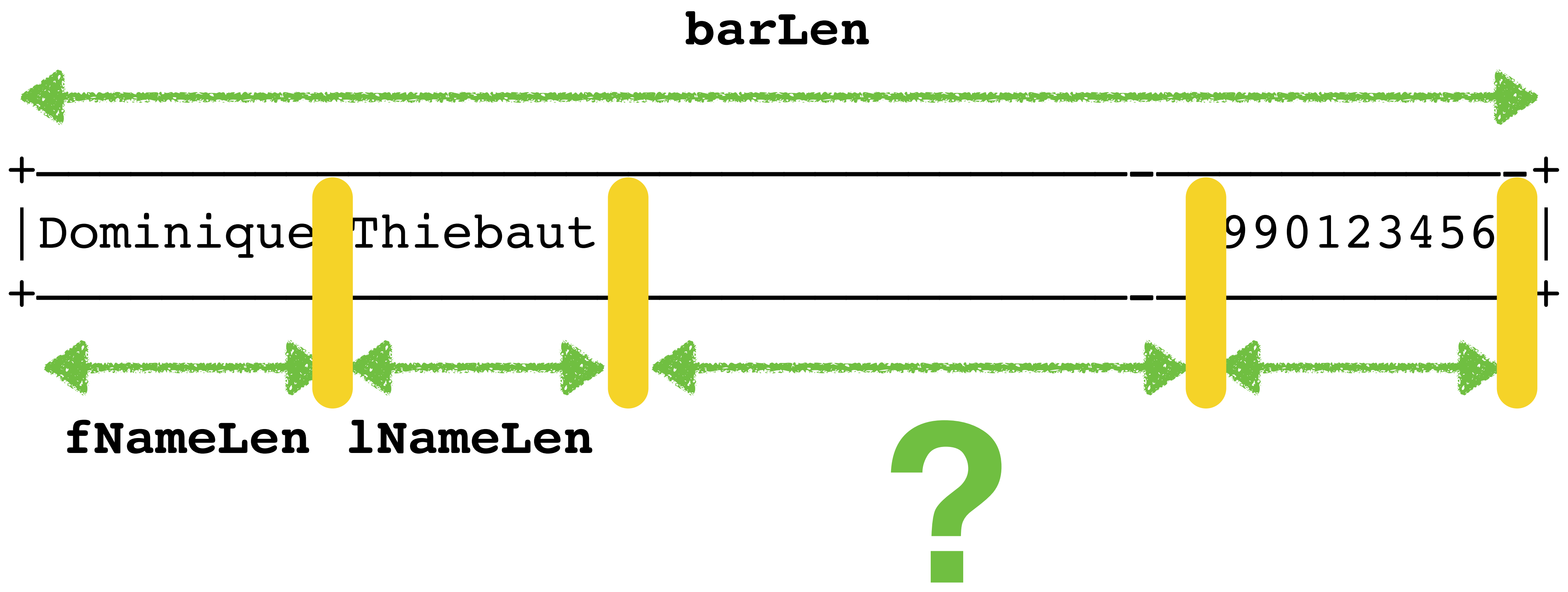
```
bar = “+------------------------------------------------+”
barLen = len( bar )
```



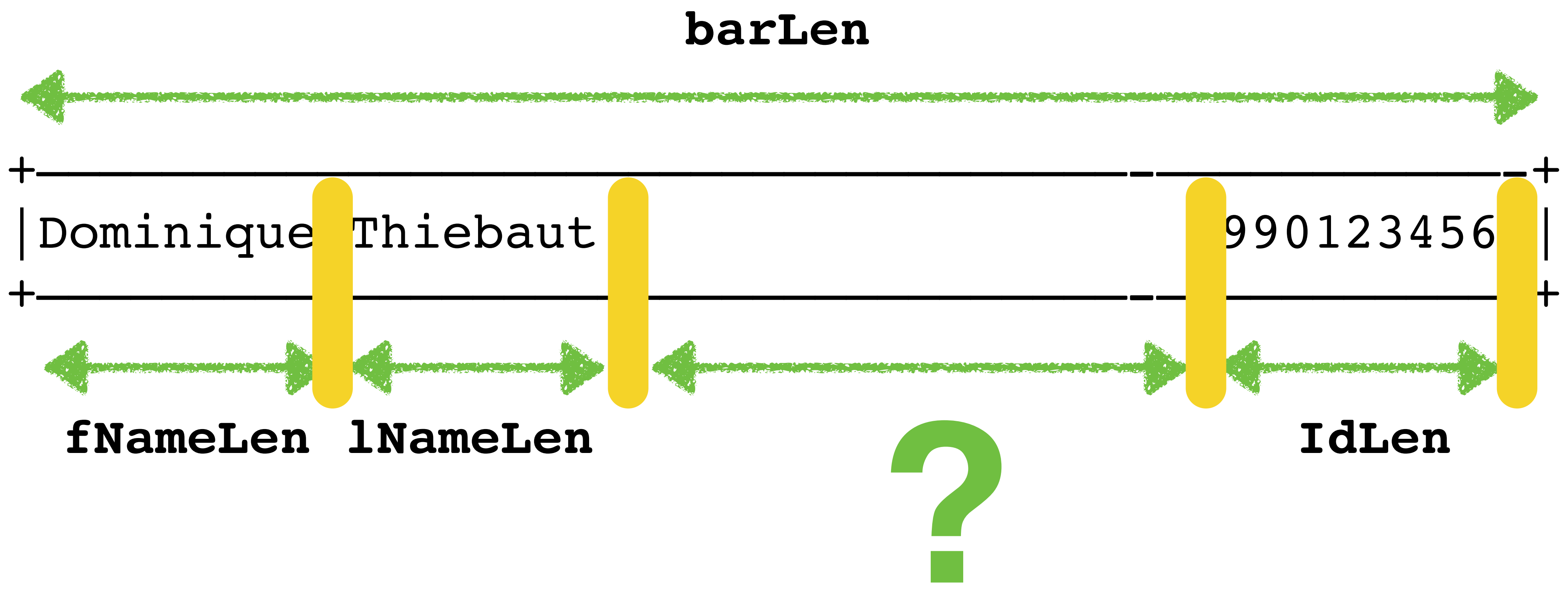
```
+---------------------------------------------------------+
|Dominique Thiebaut                            990123456 |
+---------------------------------------------------------+
```



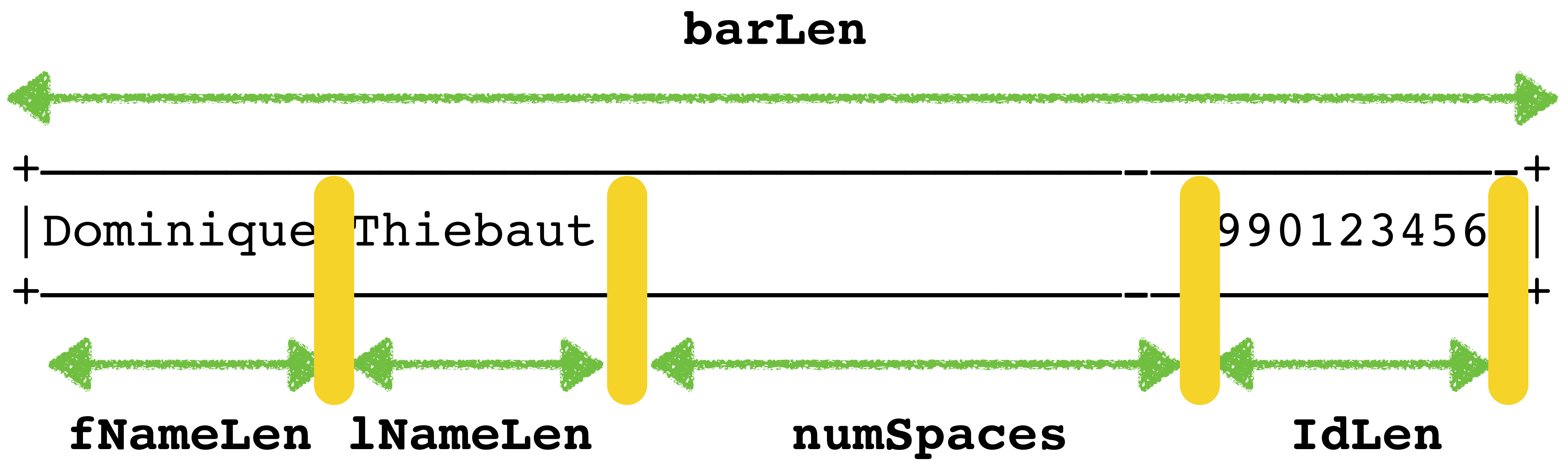
```
fName    = input( "First name? " )
fNameLen = len( fName )
```



```
fName    = input( "First name? " )
fNameLen = len( fName )
lName    = input( "Last name?  " )
lNameLen = len( lName )
```



```
fName    = input( "First name? " )
fNameLen = len( fName )
lName    = input( "Last name?  " )
lNameLen = len( lName )
Id       = input( "Id?         " )
IdLen    = len( Id )
```



```
fName     = input( "First name? " )
fNameLen = len( fName )
lName     = input( "Last name?  " )
lNameLen = len( lName )
Id        = input( "Id?         " )
IdLen     = len( Id )
numSpaces= barLen -2 -(fNameLen+1+lNameLen+1+1+IdLen+1)
```

Several Options For Printing the Box

- **Option 1:** account for the spaces generated by the comma in print()
- **Option 2:** do not use commas in print()
- **Option 3:** learn how print() works

print()

- python.org

- If that doesn't work, Google ***"python print sep end example"***

Bar Graph

```
      00...10...20...30...40...50...60...70...80...90...100
grade: ###################################################
```

Grade Number of #s

100 —> 51

Bar Graph

```
      00...10...20...30...40...50...60...70...80...90...100
grade: ###################################################
```

Grade		Number of #s
100	—>	51
50	—>	25

Bar Graph

```
      00...10...20...30...40...50...60...70...80...90...100
grade: ###################################################
```

Grade		Number of #s
100	—>	51
50	—>	25
25	—>	12

Bar Graph

```
      00...10...20...30...40...50...60...70...80...90...100
grade: ###################################################
```

Grade		Number of #s
100	—>	51
50	—>	25
25	—>	12
grade	—>	numHashTags

Bar Graph

```
      00...10...20...30...40...50...60...70...80...90...100
grade: ###################################################
```

Grade		Number of #s
100	—>	51
50	—>	25
25	—>	12
grade	—>	numDashes

Mathematical equality

numHashTags = grade / 2

Bar Graph

```
      00...10...20...30...40...50...60...70...80...90...100
grade: ###################################################
```

Grade		Number of #s
100	—>	51
50	—>	25
25	—>	12
grade	—>	numDashes

Mathematical equality and valid Python assignmen

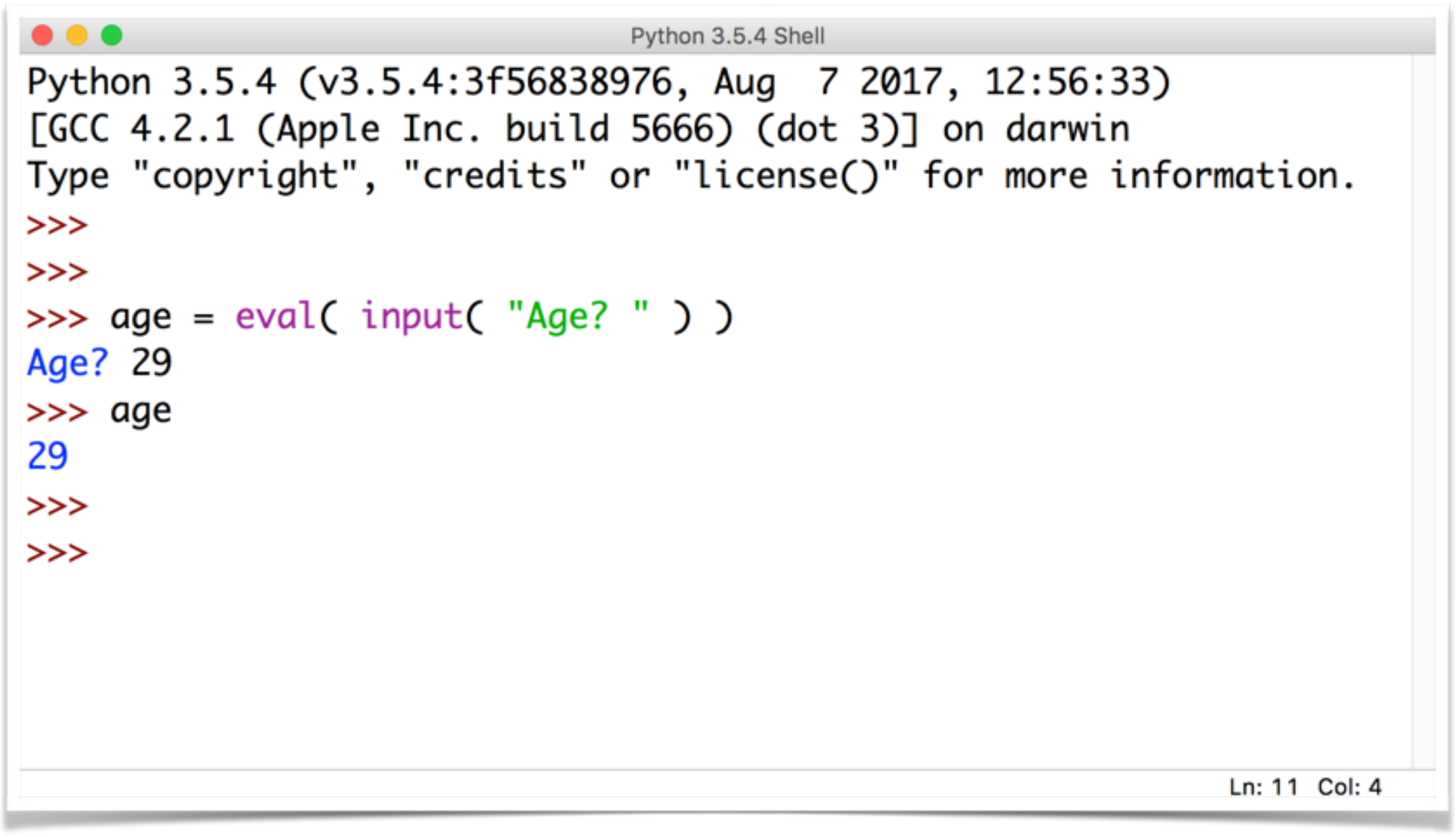
numHashTags = grade / 2

Finish the Program!

eval(*input*(...))

versus

***input*(eval(...))**



```
Python 3.5.4 (v3.5.4:3f56838976, Aug  7 2017, 12:56:33)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> 
>>> 
>>> age = eval( input( "Age? " ) )
Age? 29
>>> age
29
>>> 
>>> 
```

Problem 2

Write a python program that displays an 8x8 chessboard.   Black cells should be 3x3 with #-signs in them, and white cells should be 3x3 with spaces inside.

Problem 2 Output

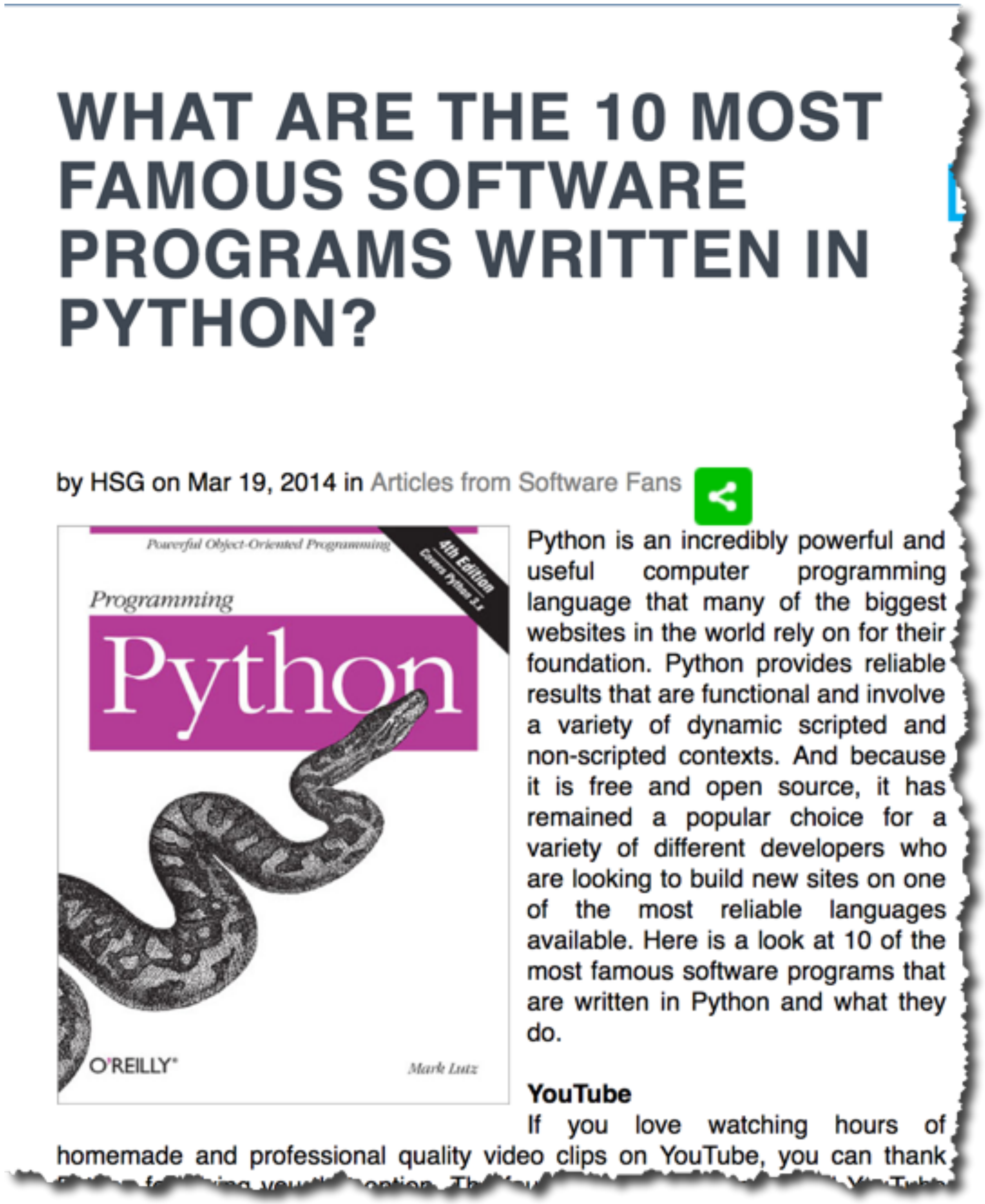
```
###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
```

Problem 2 Output

```
###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
   ###   ###   ###   ###
```

If you are ambitious, when you are done, make the program ask the user for the number of cells wanted, horizontally and vertically.

Some Apps Written in Python



WHAT ARE THE 10 MOST FAMOUS SOFTWARE PROGRAMS WRITTEN IN PYTHON?

by HSG on Mar 19, 2014 in Articles from Software Fans

Python is an incredibly powerful and useful computer programming language that many of the biggest websites in the world rely on for their foundation. Python provides reliable results that are functional and involve a variety of dynamic scripted and non-scripted contexts. And because it is free and open source, it has remained a popular choice for a variety of different developers who are looking to build new sites on one of the most reliable languages available. Here is a look at 10 of the most famous software programs that are written in Python and what they do.

YouTube

If you love watching hours of homemade and professional quality video clips on YouTube, you can thank [illegible]

(2014)

- YouTube
- Dropbox
- Google
- Quora
- Instagram
- BitTorrent
- Spotify
- Reddit
- Yahoo Maps
- Hipmunk

http://www.hartmannsoftware.com/Blog/Articles_from_Software_Fans/Most-Famous-Software-Programs-Written-in-Python

Programming Tips

- **Never** try to solve the whole problem at once
- Figure out how to **solve smaller problems** and merge pieces of code together
- Replace **inputs** by **assignments** until the last steps
- Make the program **print intermediate** values as **debug**ging help. Remove these print statements at the end.