Smith College
Computer Science

CSC 111 Introduction to Computer Science

Spring 2018 — Week 1

Dominique Thiébaut
dthiebaut@smith.edu

Quick Review

Goals for This Week

- Learn the Rules for **Pair Programming**
- Learn how to use **Idle**
- Write simple programs that use **variables, for loops,** and **output** informationf
- **Install** Python and Idle on laptop (optional)
- Learn how to **submit** Python programs to **Moodle** (lab+homework)

Reading

- Read **Chapter 1** in John Zelle's *Python Programming*

What is a Programming language?

Important Concepts…

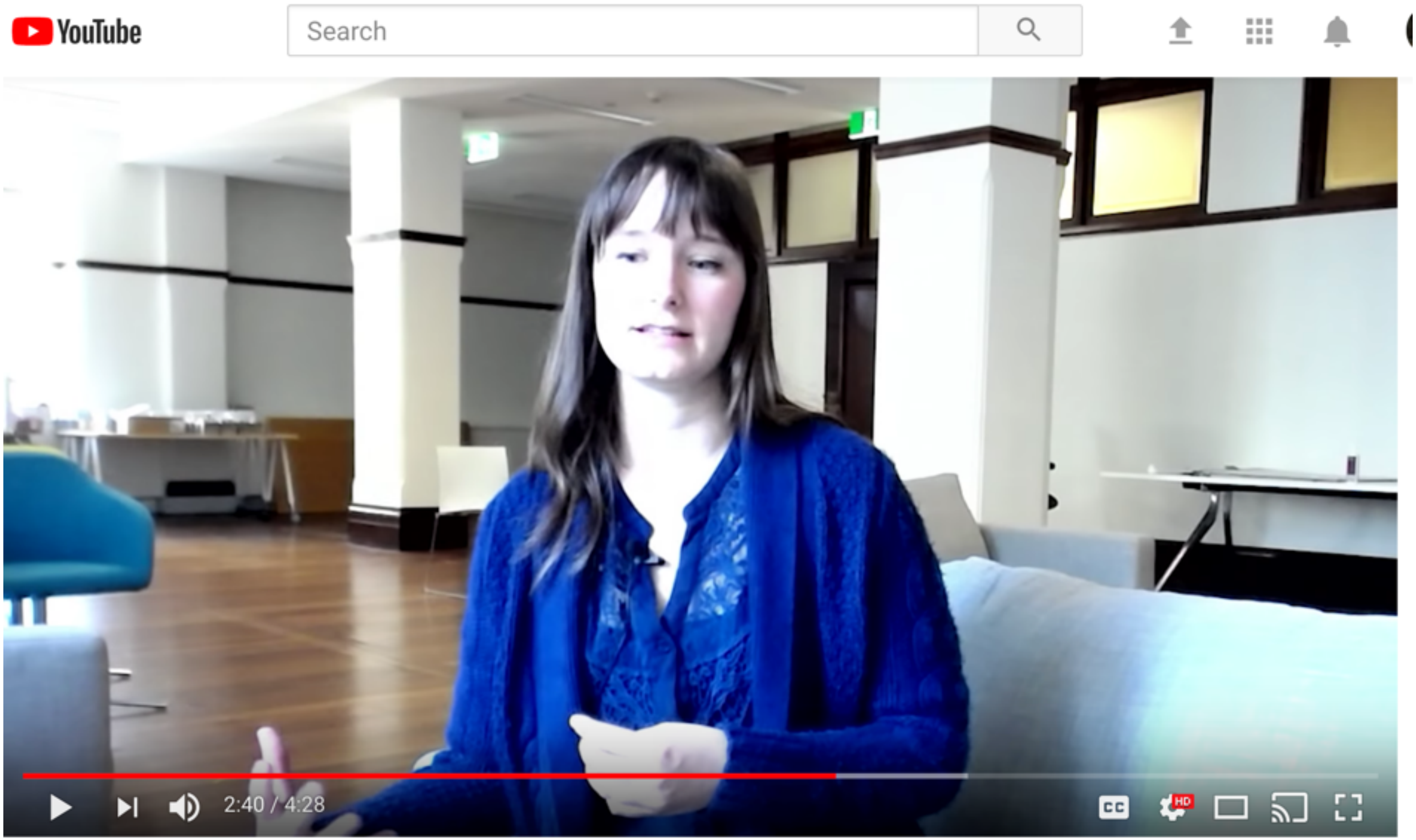
- **Syntax and keywords**

  and del from not while as elif global or with assert else if pass yield break except import **print** class exec in raise continue finally is return **def for** lambda try

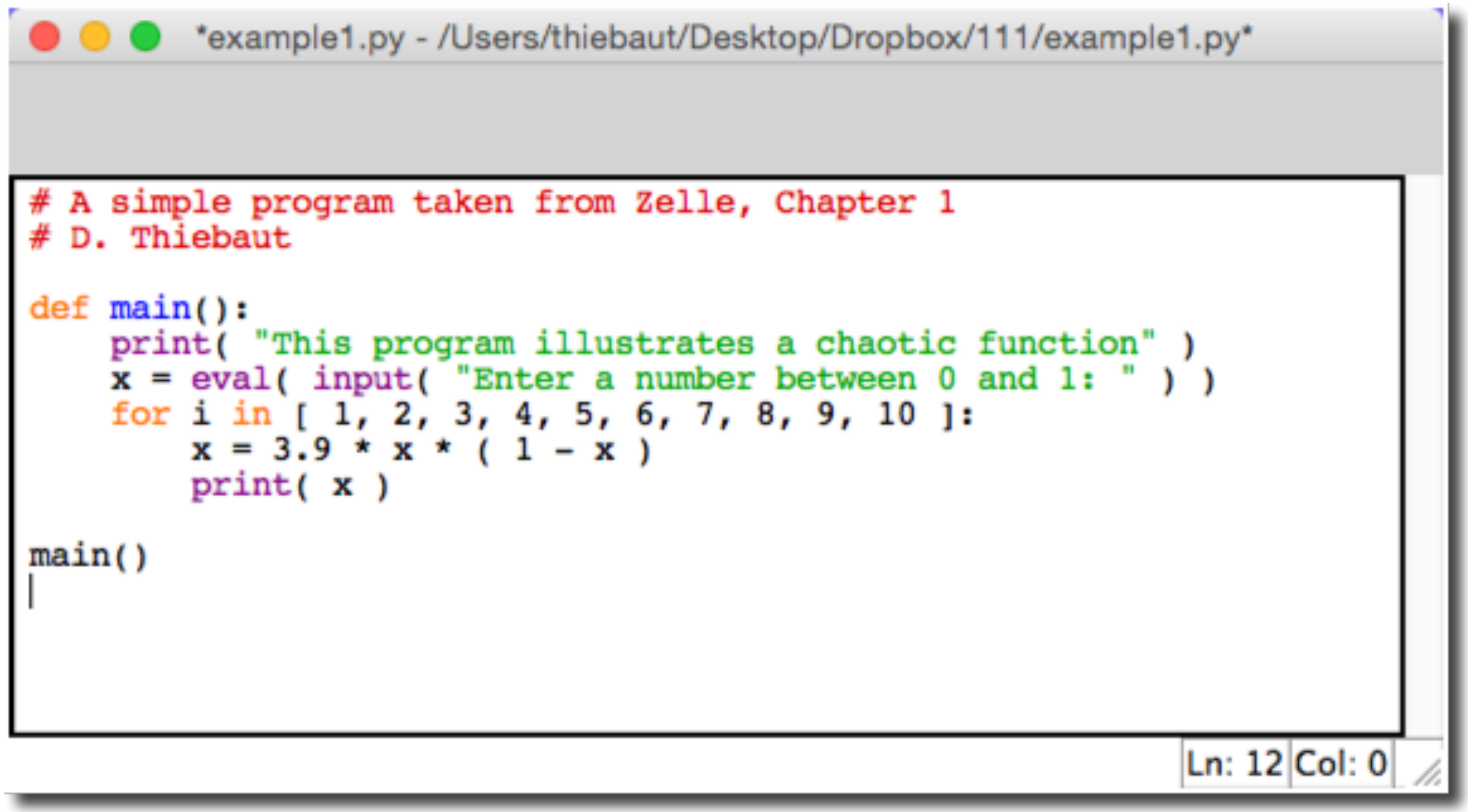
- **Algorithm**

Rules for Pair Programming

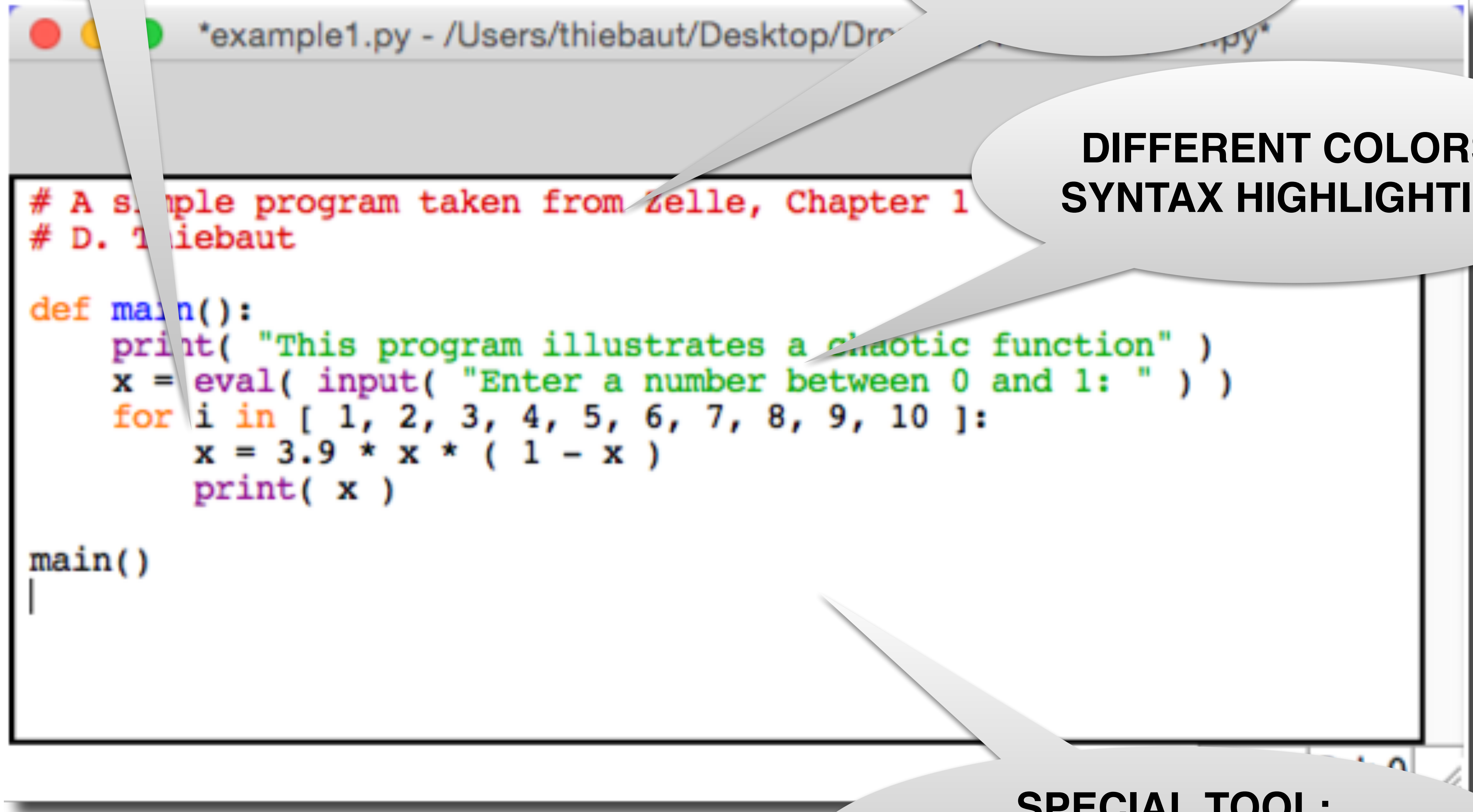
https://youtu.be/fQ-x-T34z9w



An Example Program


example1.py - /Users/thiebaut/Desktop/Dropbox/111/example1.py
Ln: 12 Col: 0

```
# A simple program taken from Zelle, Chapter 1
# D. Thiebaut

def main():
    print( "This program illustrates a chaotic function" )
    x = eval( input( "Enter a number between 0 and 1: " ) )
    for i in [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ]:
        x = 3.9 * x * ( 1 - x )
        print( x )

main()
```



```
# A simple program taken from Zelle, Chapter 1
# D. Thiebaut

def main():
    print( "This program illustrates a chaotic function" )
    x = eval( input( "Enter a number between 0 and 1: " ) )
    for i in [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ]:
        x = 3.9 * x * ( 1 - x )
        print( x )

main()
```

INDENTATION IS IMPORTANT

COMMENT

DIFFERENT COLORS: SYNTAX HIGHLIGHTING

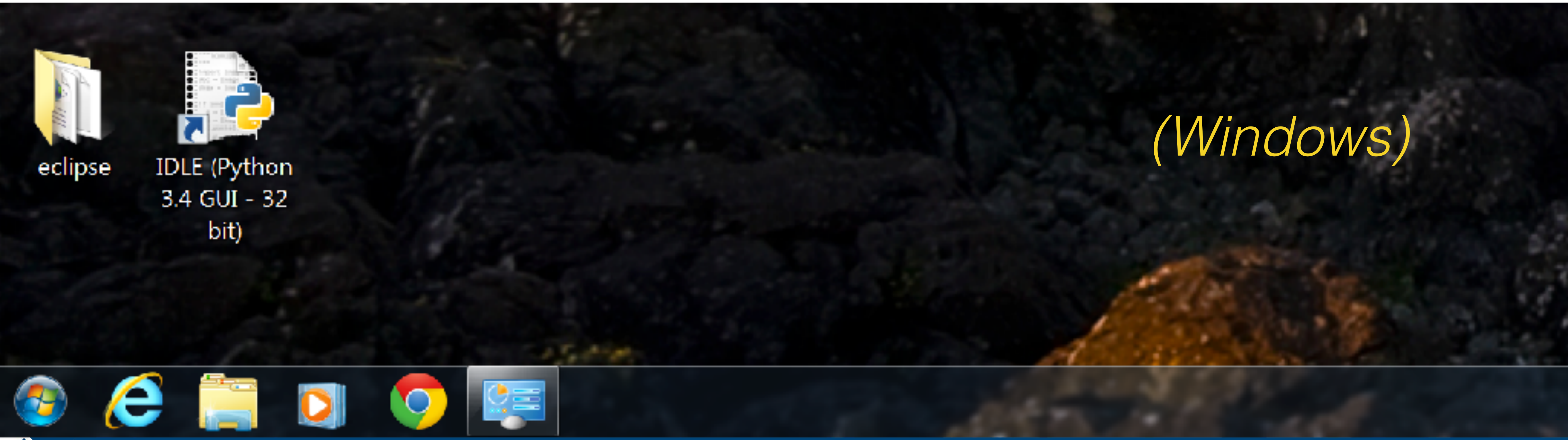
SPECIAL TOOL: EDITOR I D E

Integrated Development Environment = IDLE

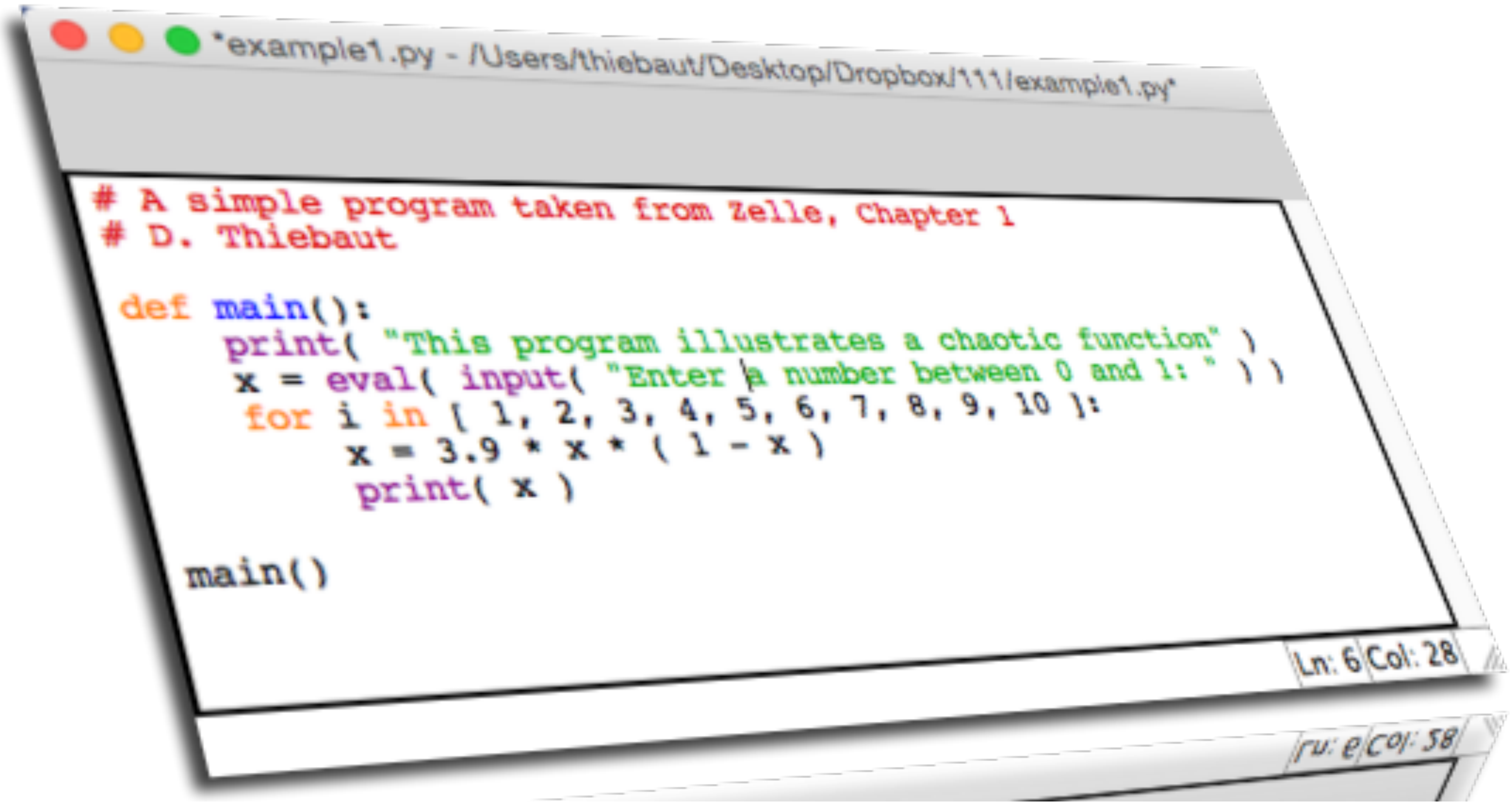
Integrated Development Environment = IDLE

Integrated Development Environment = IDLE

Integrated Development Environment = IDLE



DEMO TIME!


example1.py - /Users/thiebaut/Desktop/Dropbox/111/example1.py
Ln: 6 Col: 28

```
# A simple program taken from Zelle, Chapter 1
# D. Thiebaut

def main():
    print( "This program illustrates a chaotic function" )
    x = eval( input( "Enter a number between 0 and 1: " ) )
    for i in [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ]:
        x = 3.9 * x * ( 1 - x )
        print( x )

main()
```

Beginning of the Semester…

Concepts to Cover in Demo

- **Console** vs. **Edit window**
- ***Variables***
  - numbers: **integers** and **floats**
  - text: **strings** of characters
- **print** function

Demo Programs To Play With…

```
age = 20
year = 2015
yearBorn = year - age

print( "you are", age )
print( "you were born in", yearBorn )
```

```
name = "Alex"
college = "Smith College"
print( name, "goes to", college )
```

```
for name in  [ "Lea Jones", "Julie Fleur", "Anu Vias" ]:
     print( name )
     print( "———————" )
```

Demo Programs To Play With… (cont’d)

```
for name in  [ "Lea Jones", "Julie Fleur", "Anu Vias" ]:
     print( name, len( name ) )
```

```
print( "hello" * 4 )
print( "-" * 10 )
greetings = “hello”
dash = "-"
print( greetings * 4 )
print( dash * 10 )
```

```
greetings = "hello"
longGreetings = greeting * 4
print( greetings )
print( longGreetings )
```

Demo Programs To Play With… (cont'd)

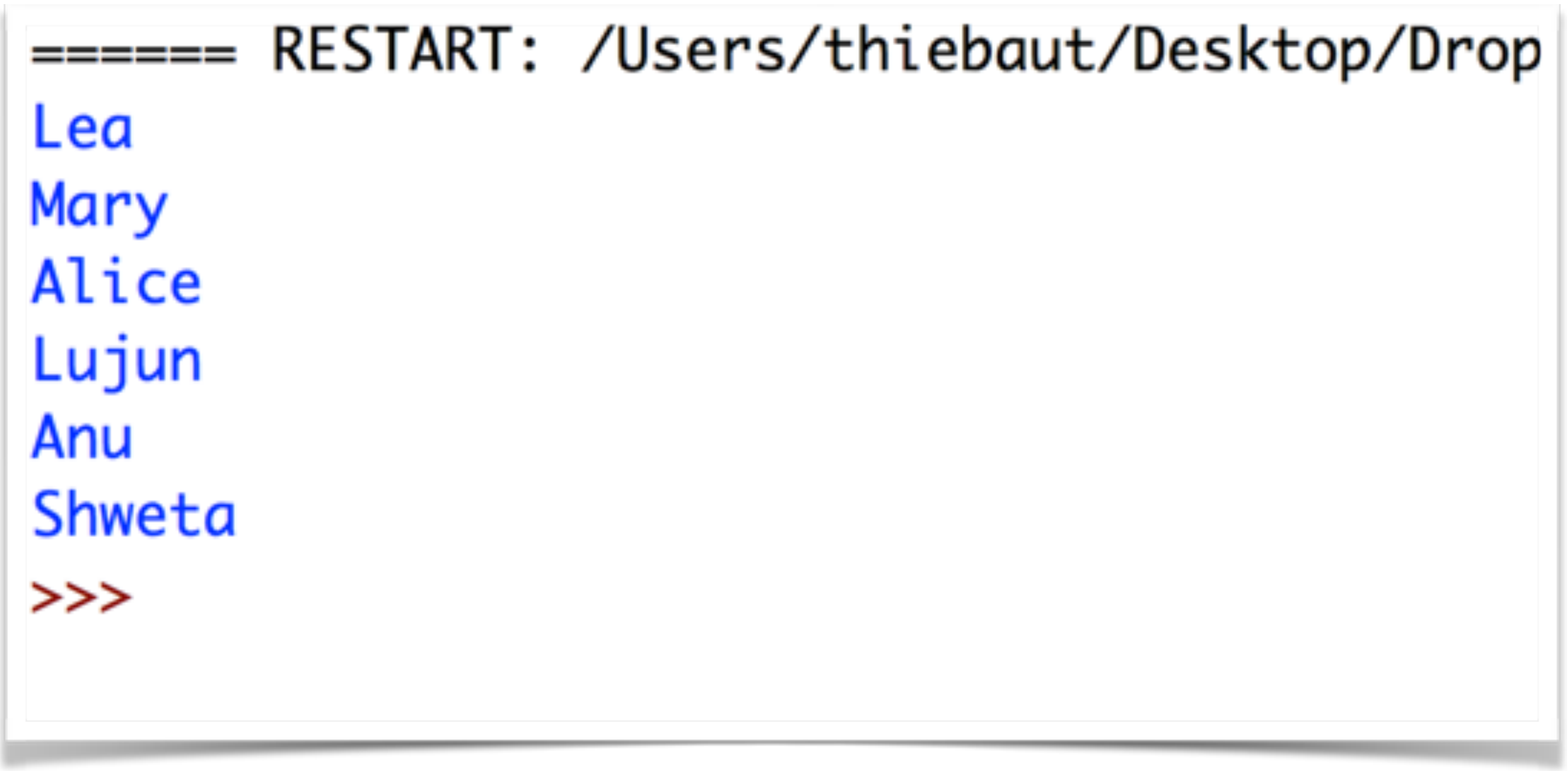
```
for name in  [ "Lea Jones", "Julie Fleur", "Anu Vias" ]:
     bar = len( name ) * "-"
     print( name )
     print( bar )
```

```
print( "hello" * 4 )
print( "-" * 10 )

greetings = “hello”
dash = "-"
print( greetings * 4 )
print( dash * 10 )
```

```
greetings = "hello"
longGreetings = greeting * 4
print( greetings )
print( longGreetings )
```

Exercise 1

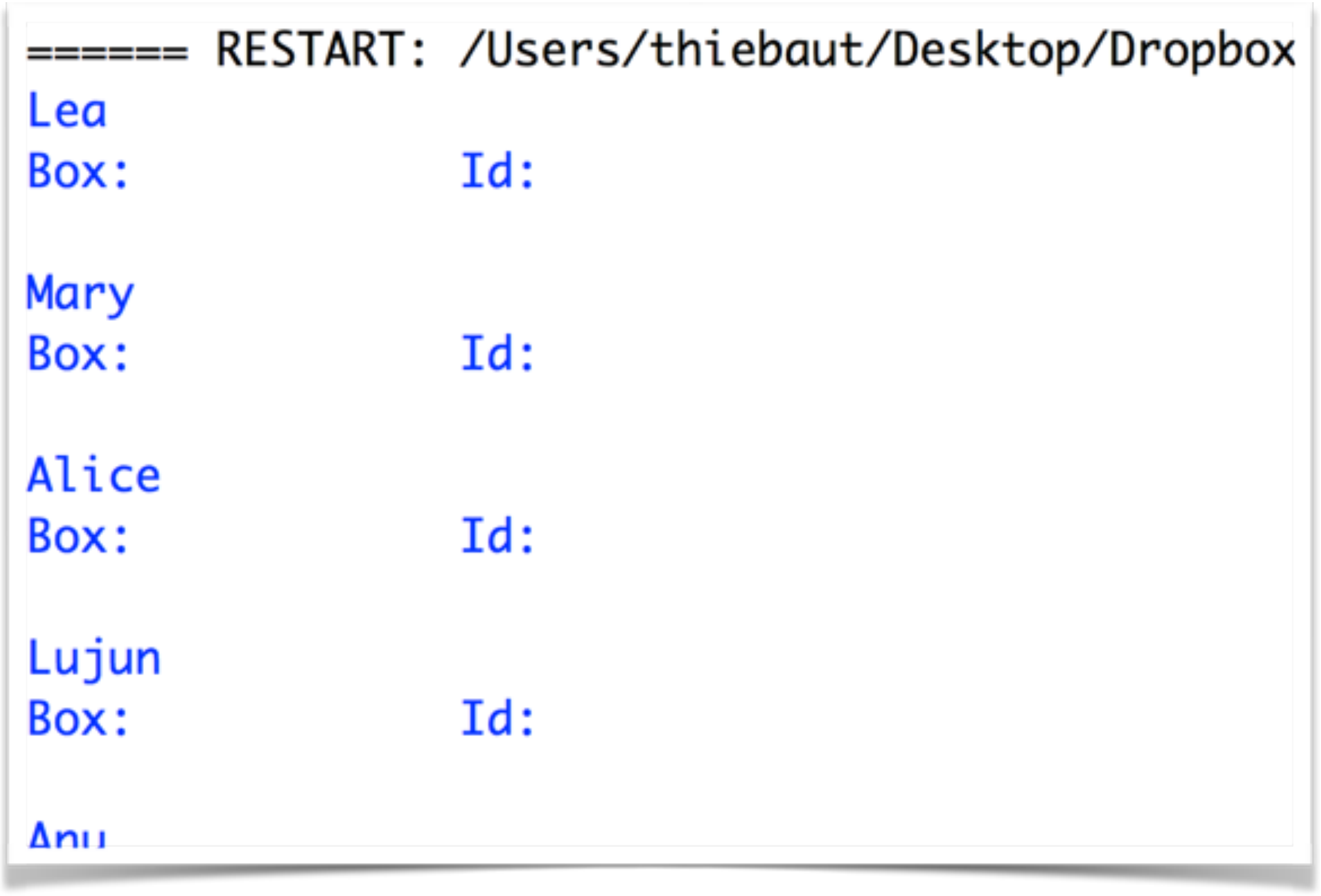
Lea
Mary
Alice
Lujun
Anu
Shweta



```
====== RESTART: /Users/thiebaut/Desktop/Drop
Lea
Mary
Alice
Lujun
Anu
Shweta
>>>
```

Exercise 2

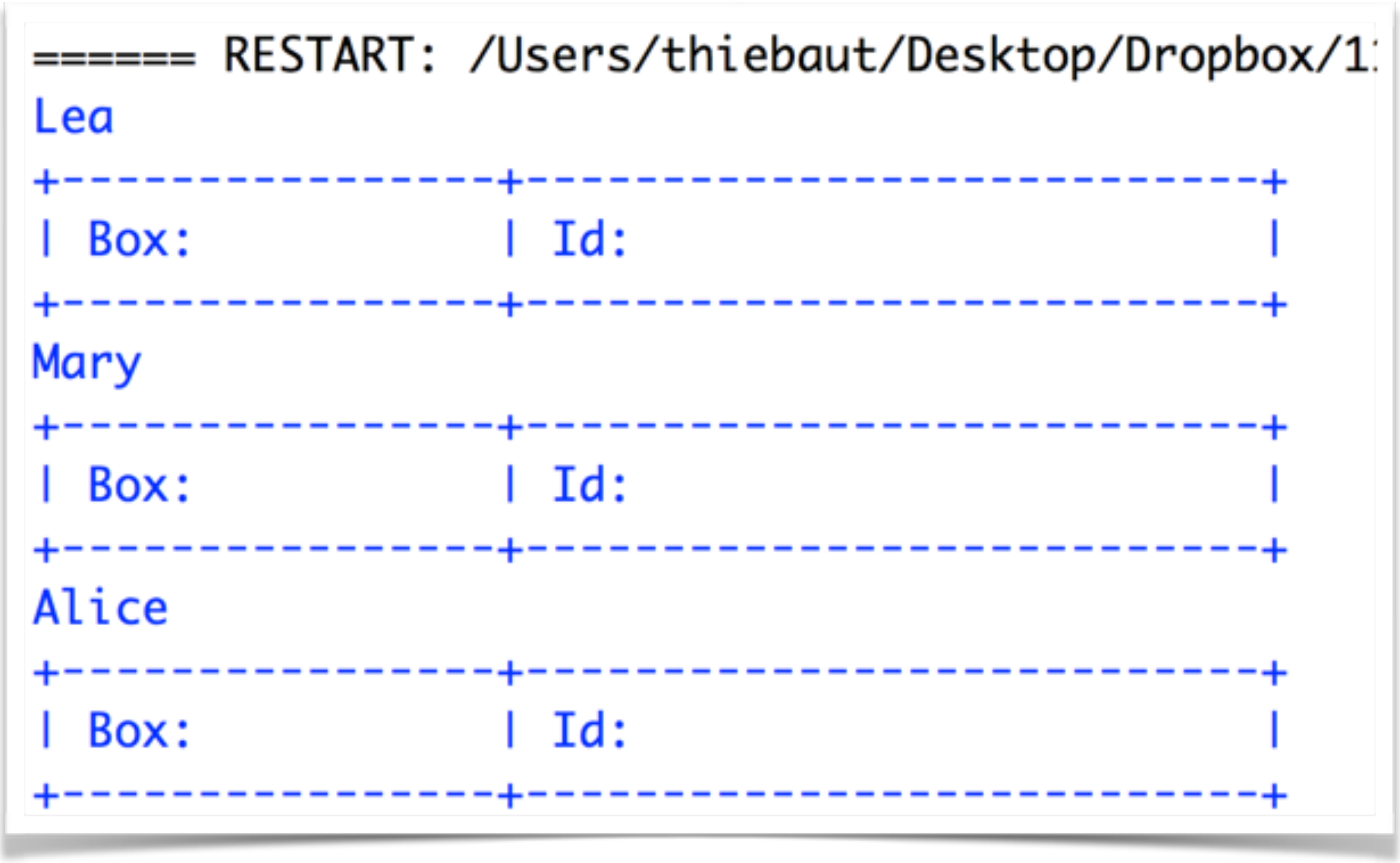
Lea
Mary
Alice
Lujun
Anu
Shweta



```
====== RESTART: /Users/thiebaut/Desktop/Dropbox
Lea
Box:                Id:

Mary
Box:                Id:

Alice
Box:                Id:

Lujun
Box:                Id:

Anu
```

Exercise 3

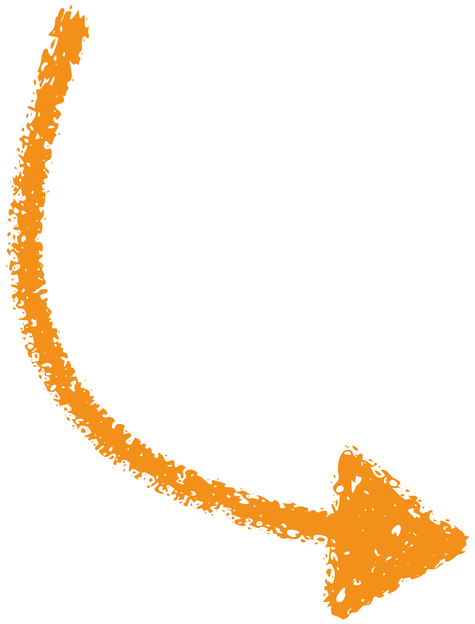
Lea
Mary
Alice
Lujun
Anu
Shweta

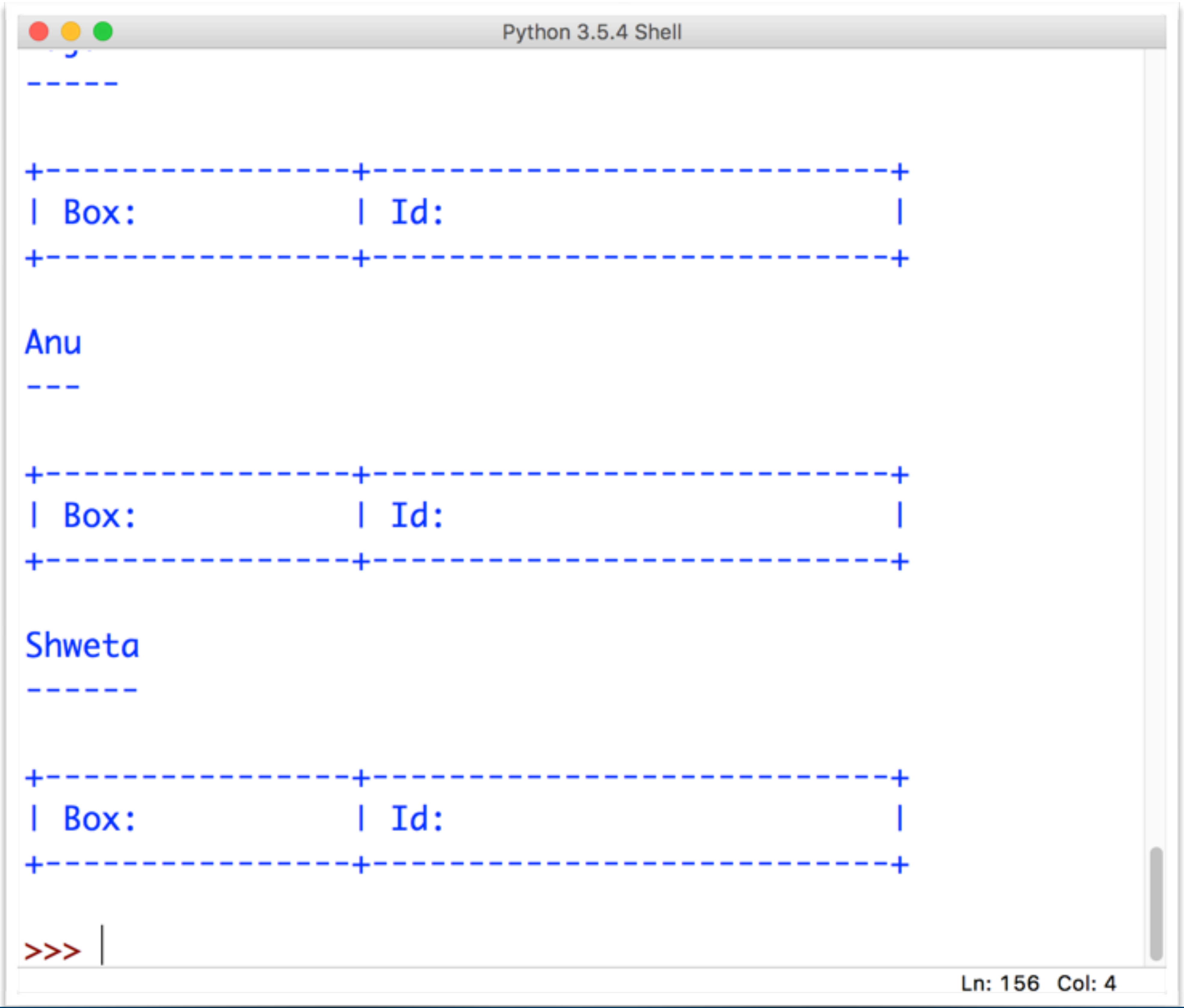


```
====== RESTART: /Users/thiebaut/Desktop/Dropbox/1
Lea
+----------------+--------------------------+
| Box:           | Id:                      |
+----------------+--------------------------+
Mary
+----------------+--------------------------+
| Box:           | Id:                      |
+----------------+--------------------------+
Alice
+----------------+--------------------------+
| Box:           | Id:                      |
+----------------+--------------------------+
```

Exercise 4

Lea
Mary
Alice
Lujun
Anu
Shweta





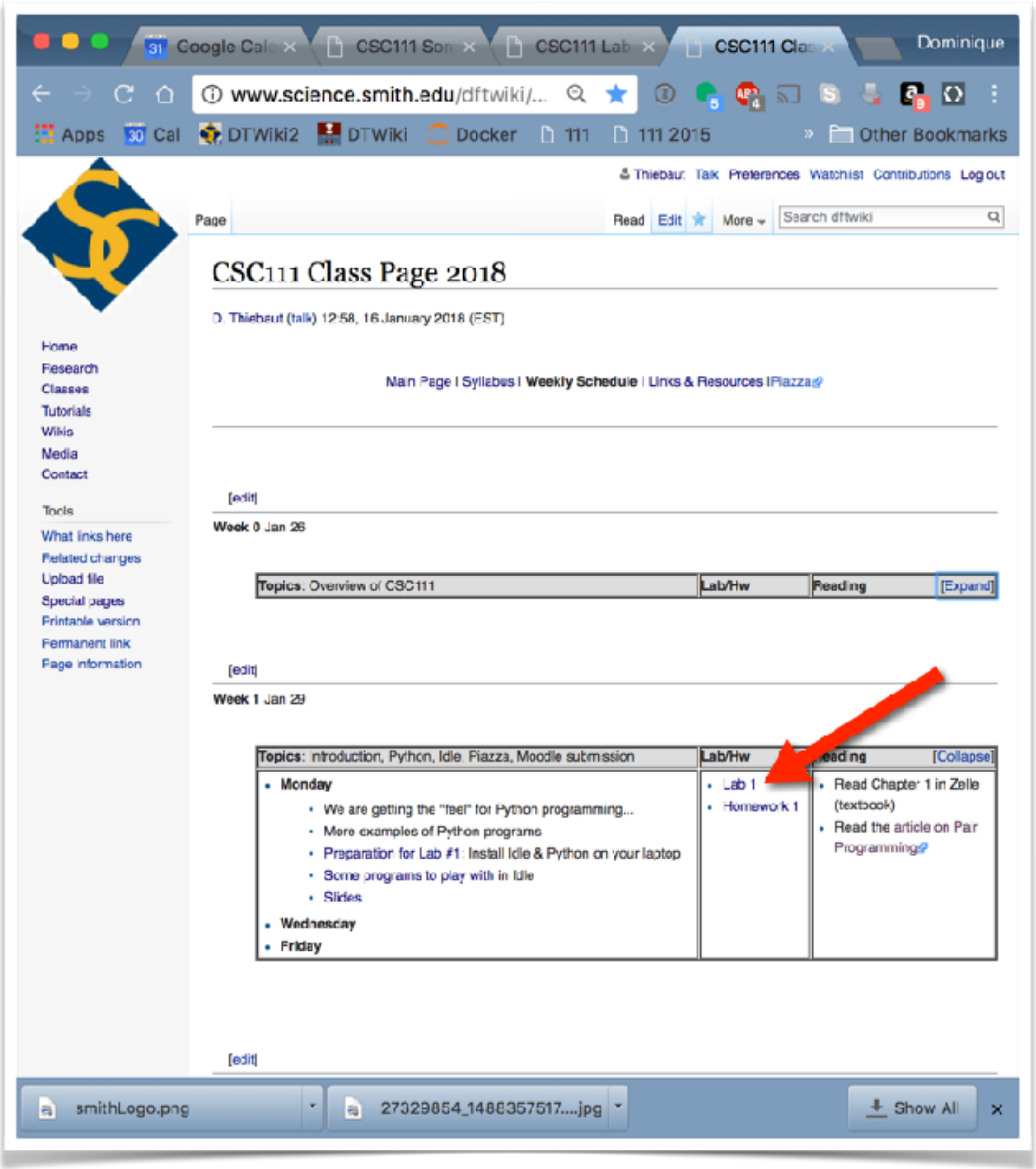
```
-----


+---------------+--------------------------+
| Box:          | Id:                      |
+---------------+--------------------------+

Anu
---


+---------------+--------------------------+
| Box:          | Id:                      |
+---------------+--------------------------+

Shweta
------


+---------------+--------------------------+
| Box:          | Id:                      |
+---------------+--------------------------+

>>> 
```

We stopped here last time…

Outline

- Introduction to Lab 1
- Assignment
- Introduction to Variables
- Exercise

Lab 1



CSC111 Class Page 2018

D. Thiebaut (talk) 12:58, 16 January 2018 (EST)

Main Page | Syllabus | Weekly Schedule | Links & Resources | Piazza

[edit]

Week 0 Jan 26

Topics: Overview of CSC111	Lab/Hw	Reading [Expand]

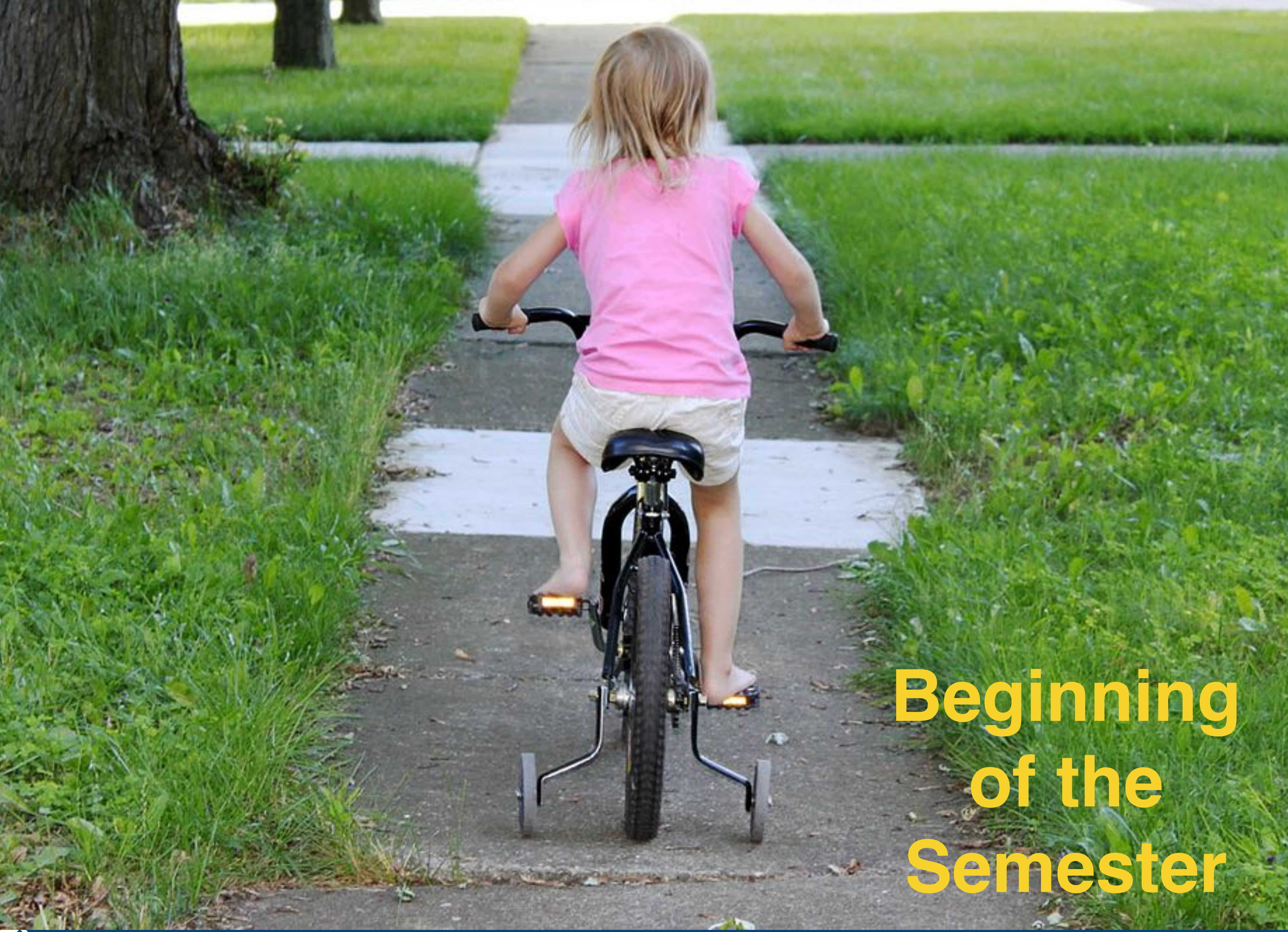
[edit]

Week 1 Jan 29

Topics: Introduction, Python, Idle, Piazza, Moodle submission	Lab/Hw	Reading [Collapse]
• Monday • We are getting the "feel" for Python programming... • More examples of Python programs • Preparation for Lab #1: Install Idle & Python on your laptop • Some programs to play with in Idle • Slides • Wednesday • Friday	• Lab 1 • Homework 1	• Read Chapter 1 in Zelle (textbook) • Read the article on Pair Programming

[edit]

Practice Python!



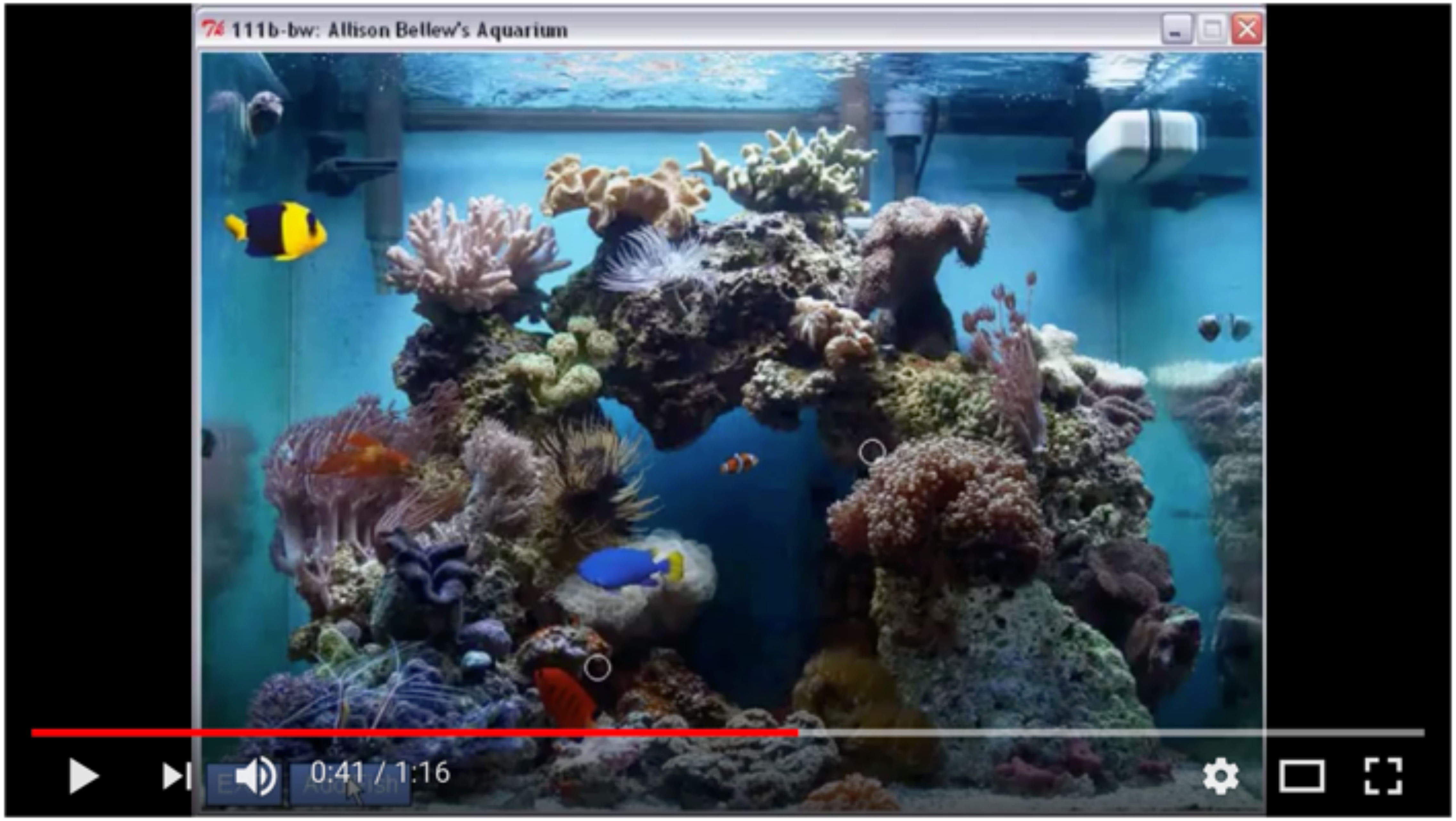
Beginning of the Semester

AFTER ONE SEMESTER

Computer Science Major

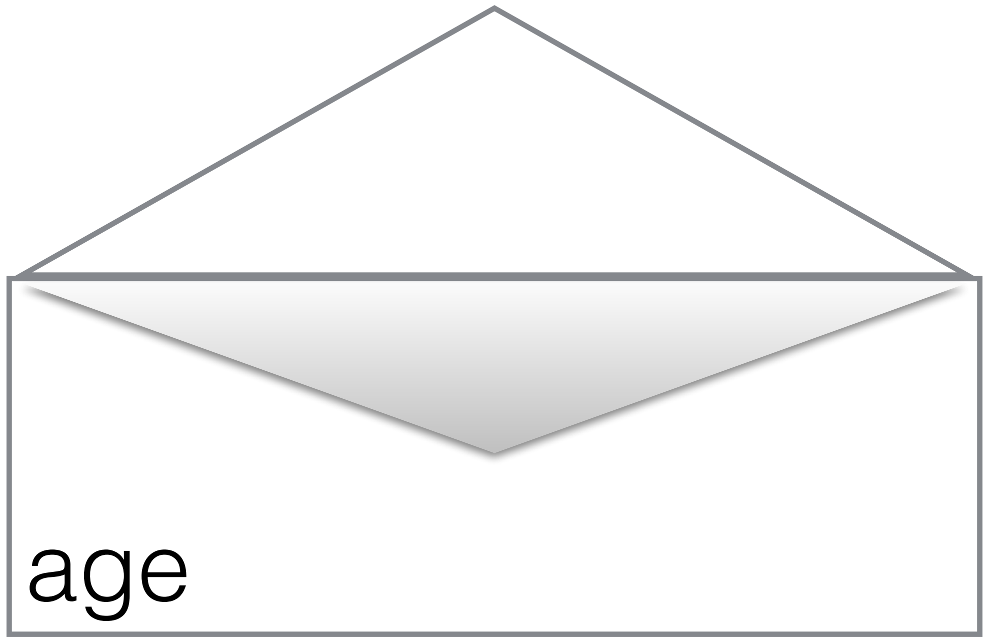
Final Project From the Past

https://www.youtube.com/watch?v=g_82xHimSNE

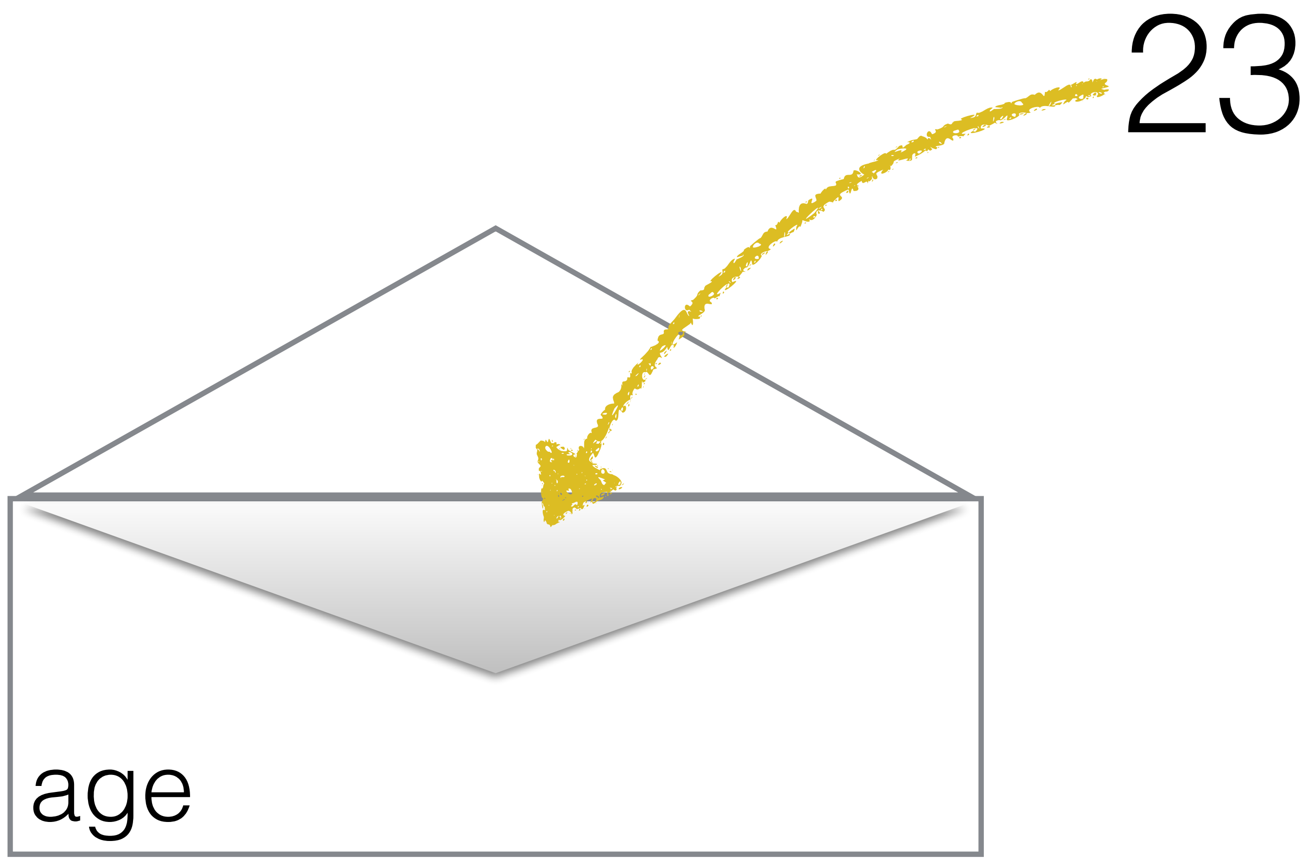


Memory

Variables

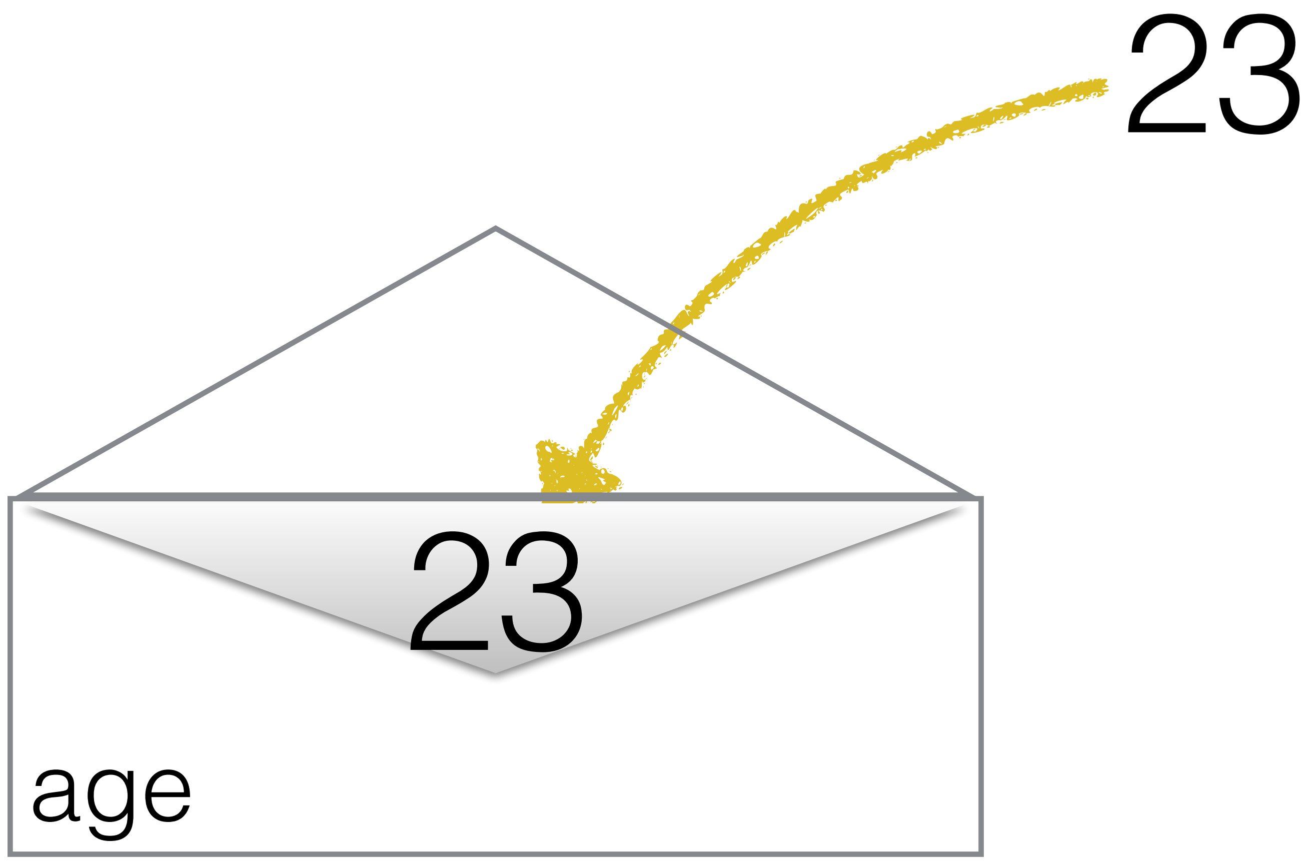


Variables



age = 23

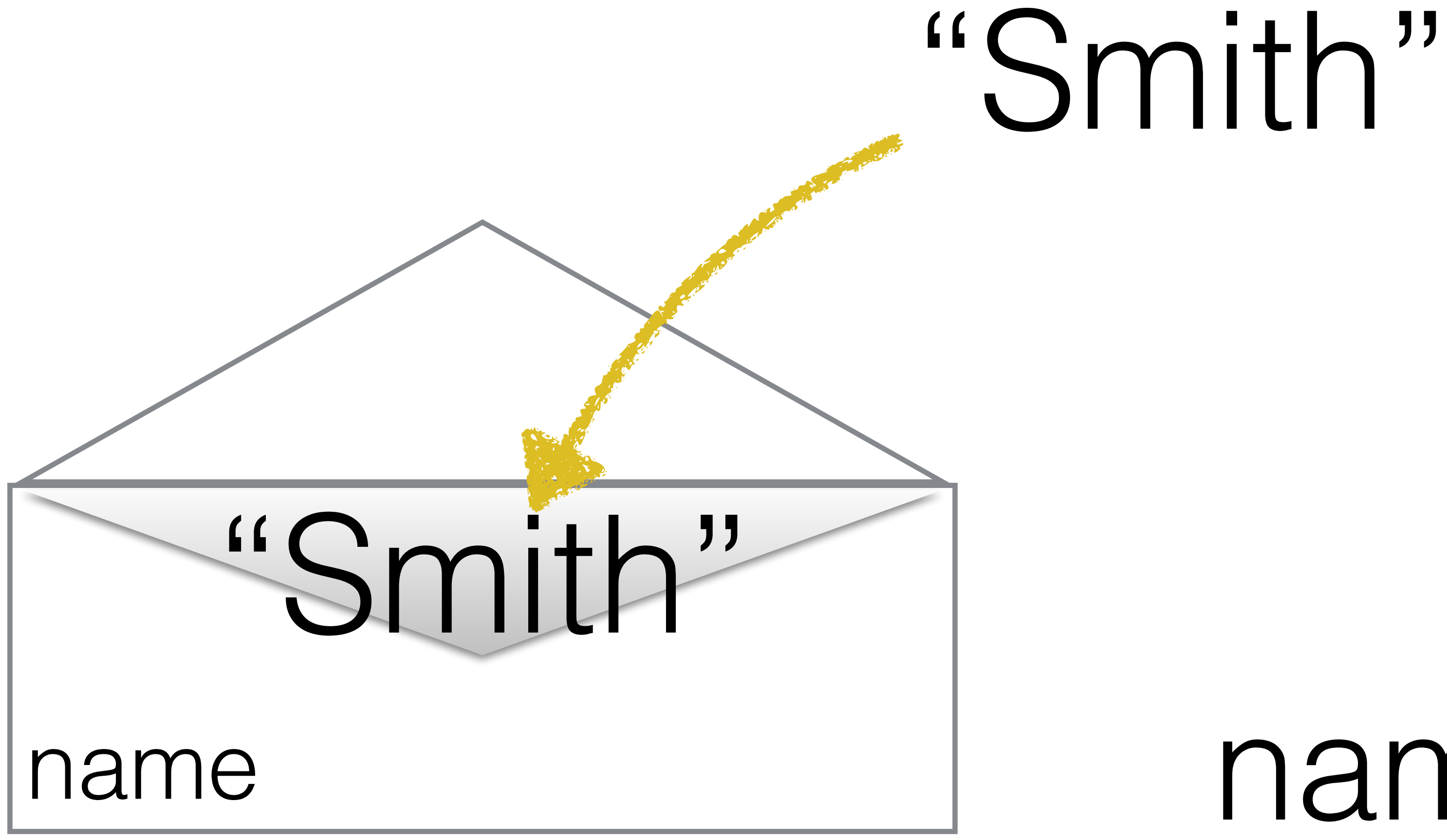
Variables



age = 23

assignment

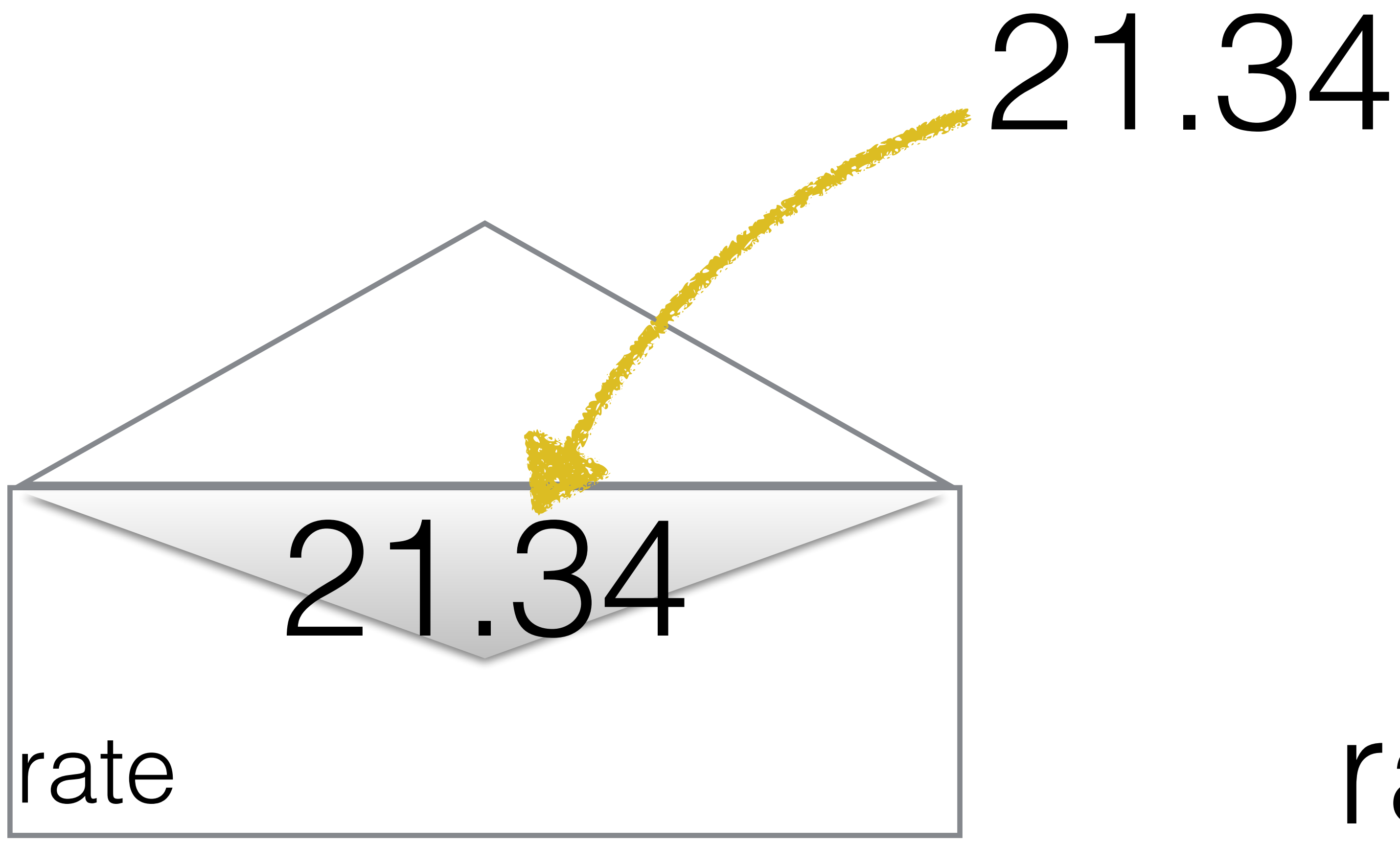
Variables



name = "Smith"

assignment

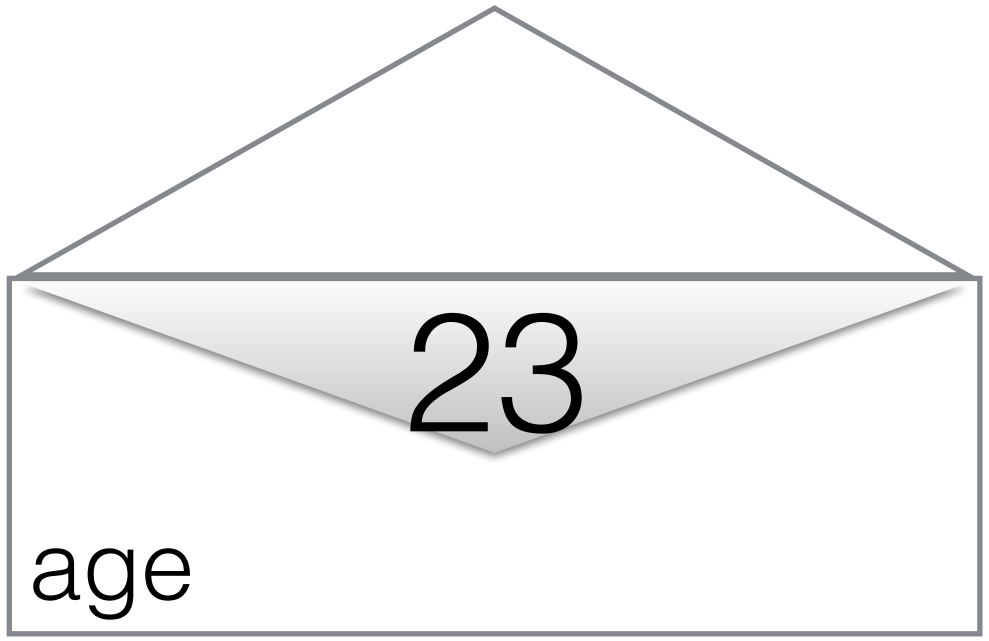
Variables

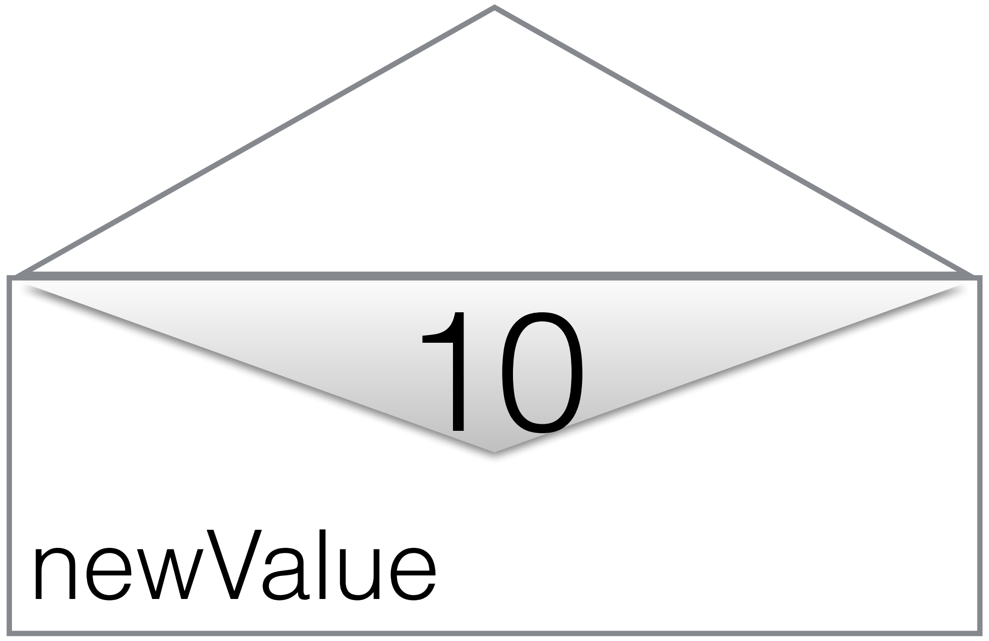


rate = 21.34

assignment

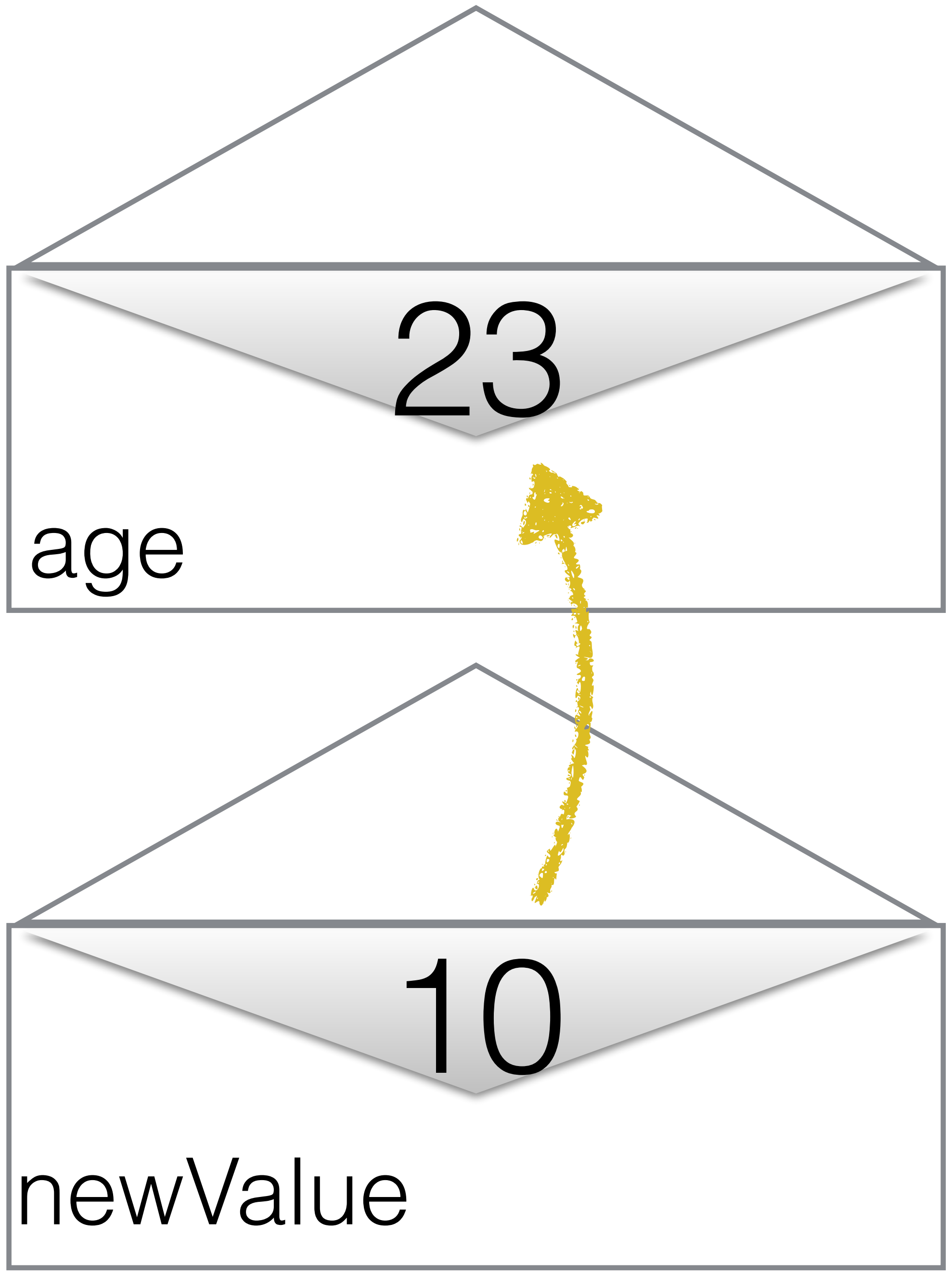
Variables & Expressions





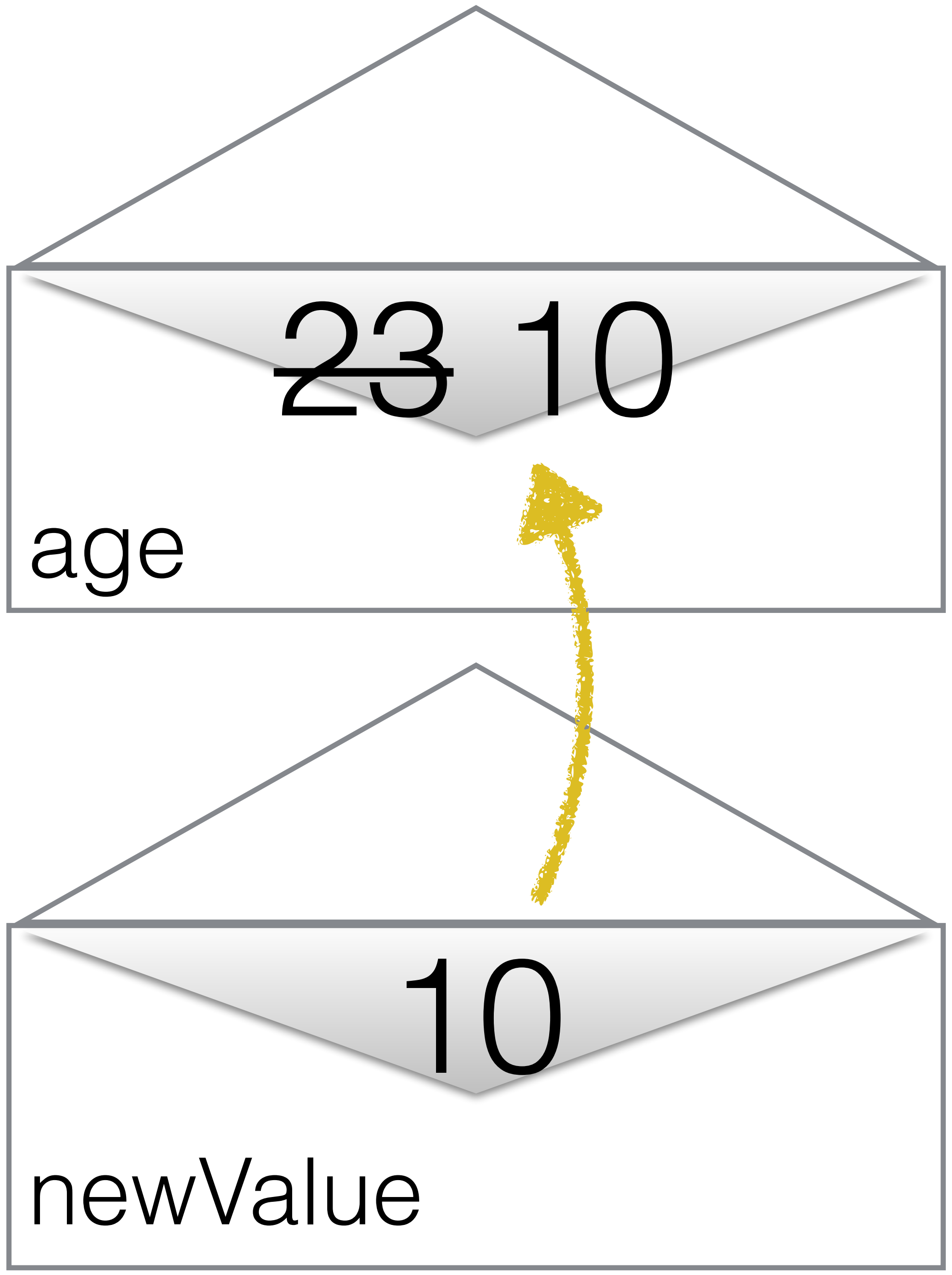
age = 23
newValue = 10

Variables & Expressions



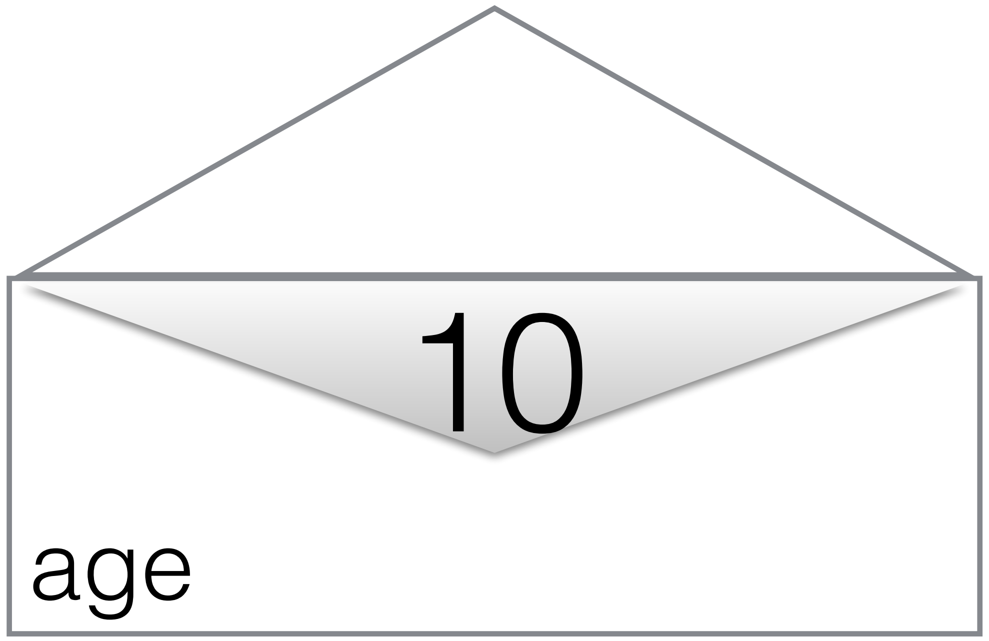
```
age = 23
newValue = 10
age = newValue
```

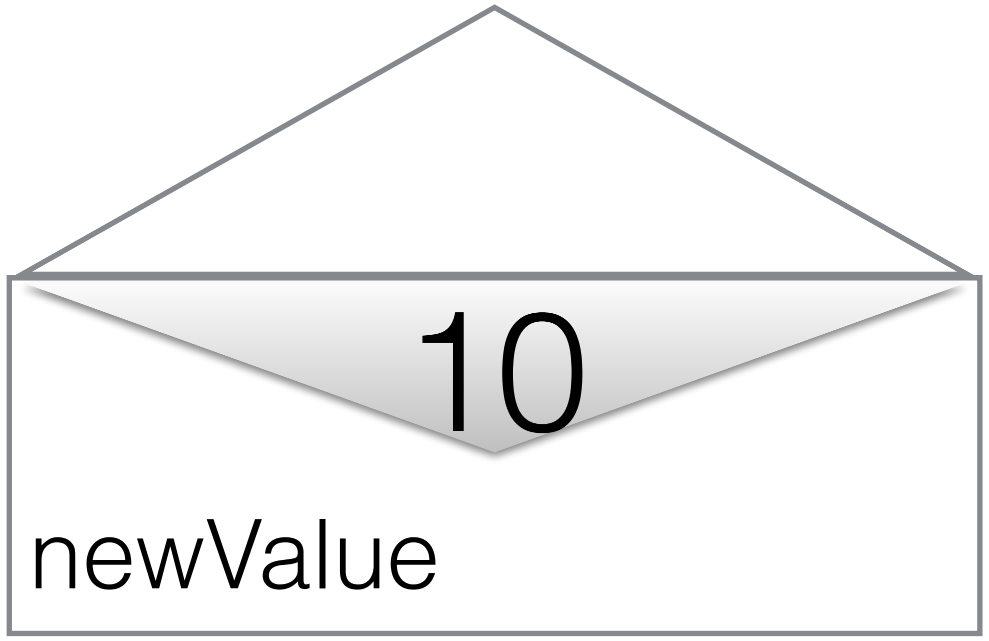
Variables & Expressions



```
age = 23
newValue = 10
age = newValue
```

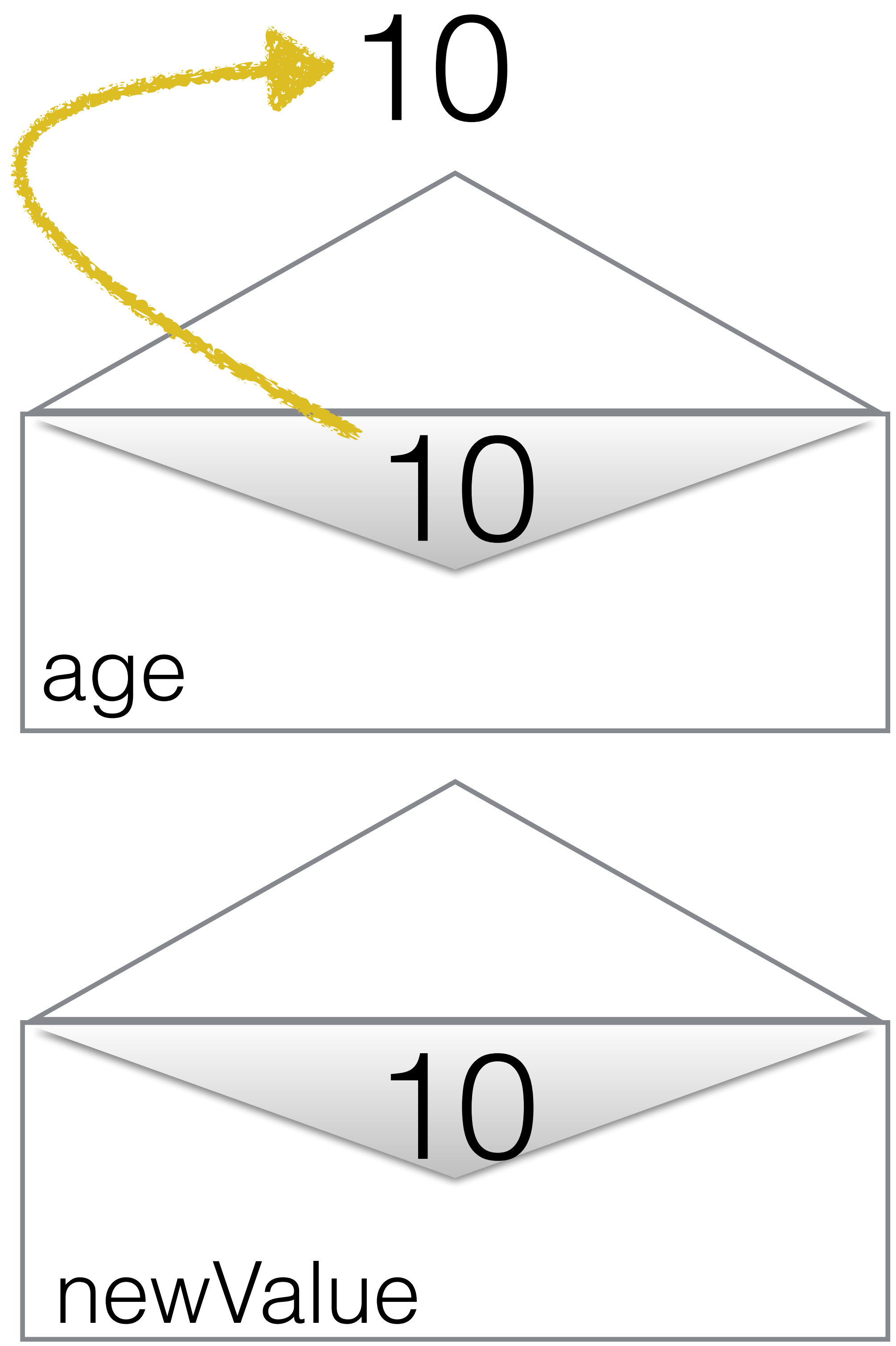
Variables & Expressions





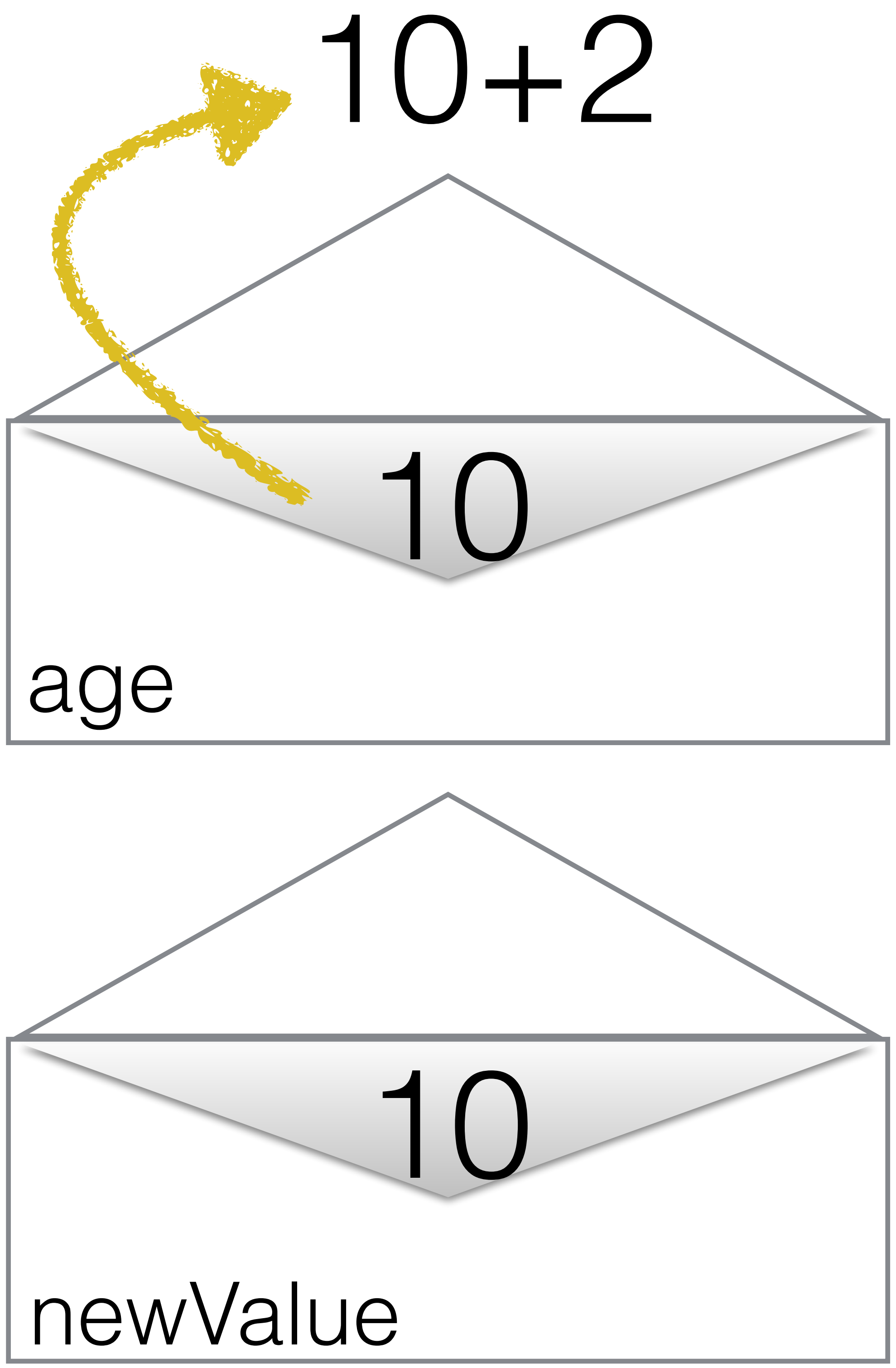
```
age = 23
newValue = 10
age = newValue
age = age + 2
```

Variables & Expressions



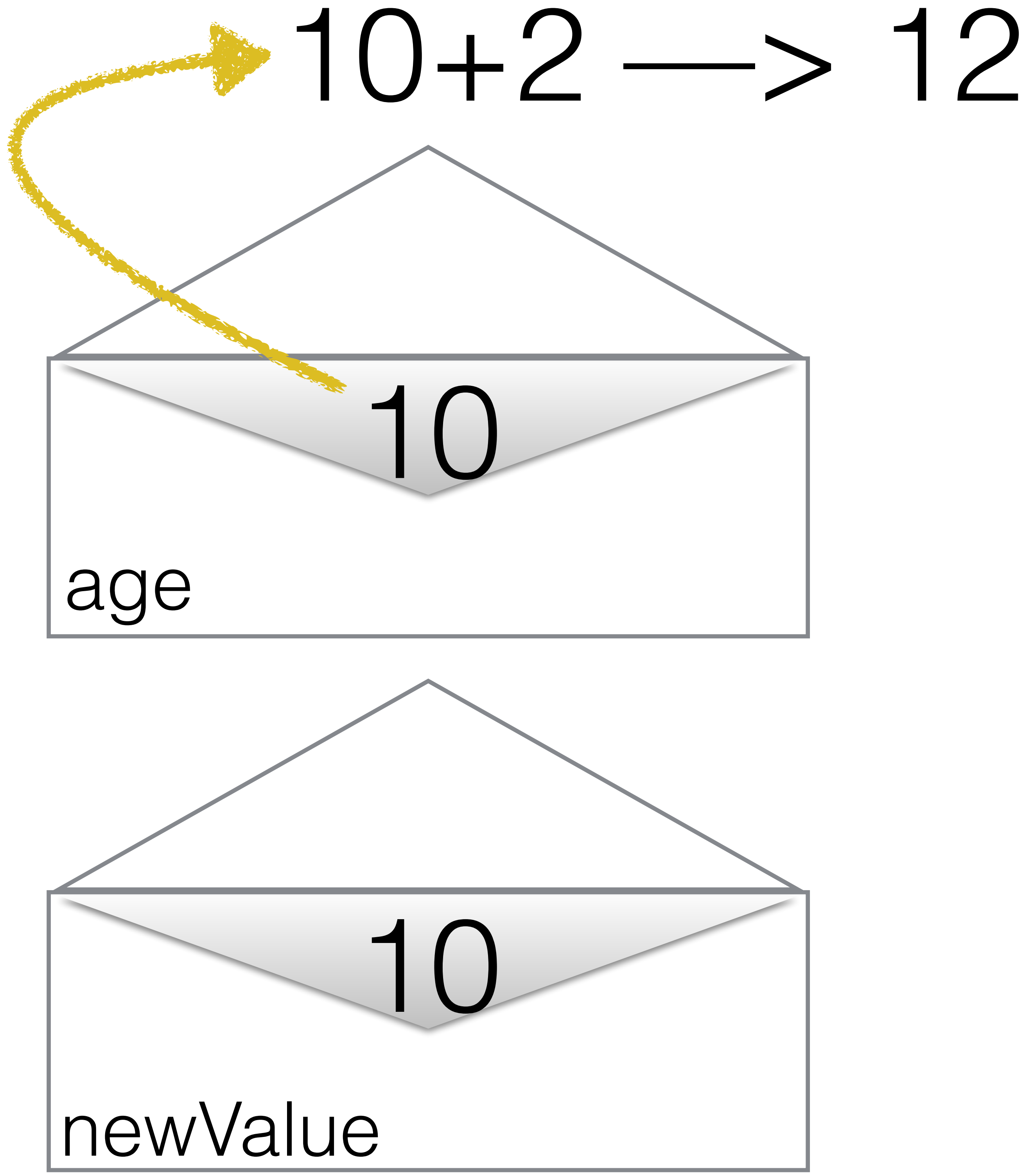
```
age = 23
newValue = 10
age = newValue
age = age + 2
```

Variables & Expressions



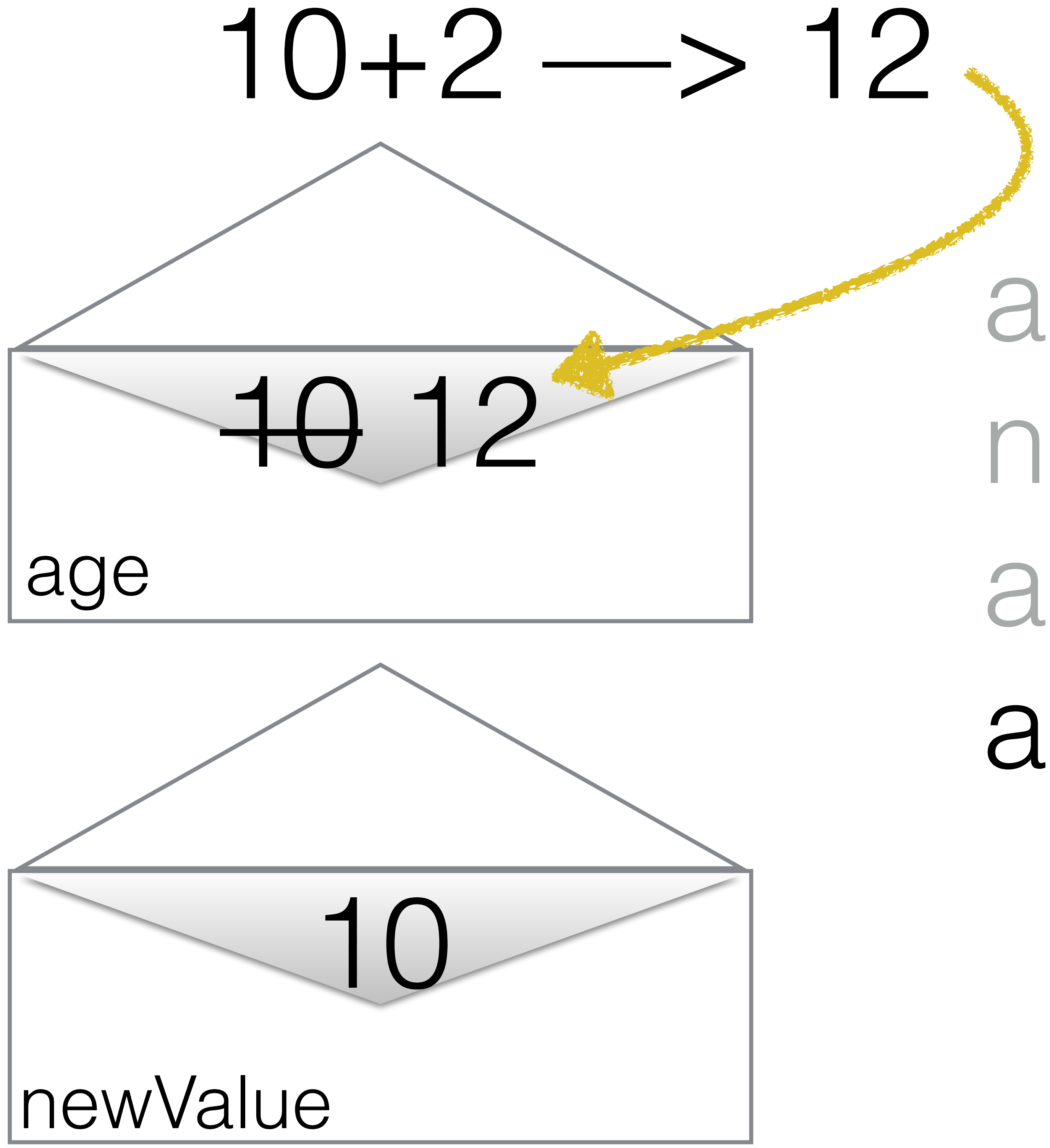
```
age = 23
newValue = 10
age = newValue
age = age + 2
```

Variables & Expressions



```
age = 23
newValue = 10
age = newValue
age = age + 2
```

Variables & Expressions



```
age = 23
newValue = 10
age = newValue
age = age + 2
```

Exercise

```
a = 10
b = 20
c = 30
a = b                    # a = ?
```

Exercise

```
a = 10
b = 20
c = 30
a = b                    # a = 20
b = a                    # a = ?      b = ?
```

Exercise

```
a = 10
b = 20
c = 30
a = b                # a = 20
b = a                # a = 20    b = 20
c = c * 2            # c = ?
```

Exercise

```
a = 10
b = 20
c = 30
a = b                  # a = 20
b = a                  # a = 20    b = 20
c = c * 2              # c = 60
d = d - 10             # d = ?
```

Exercise

```
a = 10
b = 20
c = 30
a = b           # a = 20
b = a           # a = 20    b = 20
c = c * 2       # c = 60
d = d - 10      # NameError:
                # name 'd' is not defined
```

Naming Variables

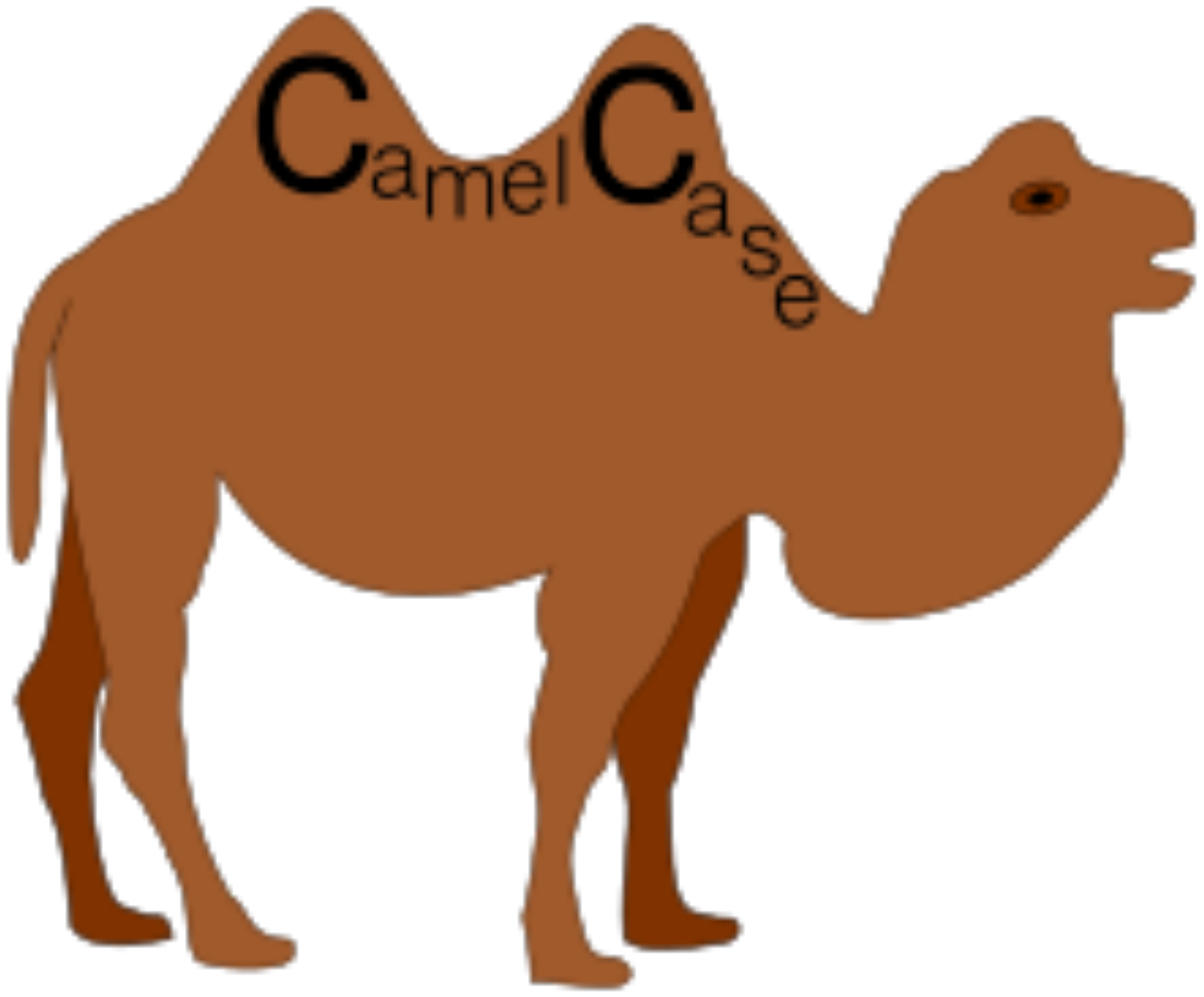
- Variable name cannot be a **keyword**

  and del from not while as elif global or with assert else if pass yield break except import print class exec in raise continue finally is return def for lambda try

- First letter must be **alphabetic** (upper- or lower-case, or underscore)

- Can be followed by 0, 1, or more **letters**, **digits**, or **underscore**

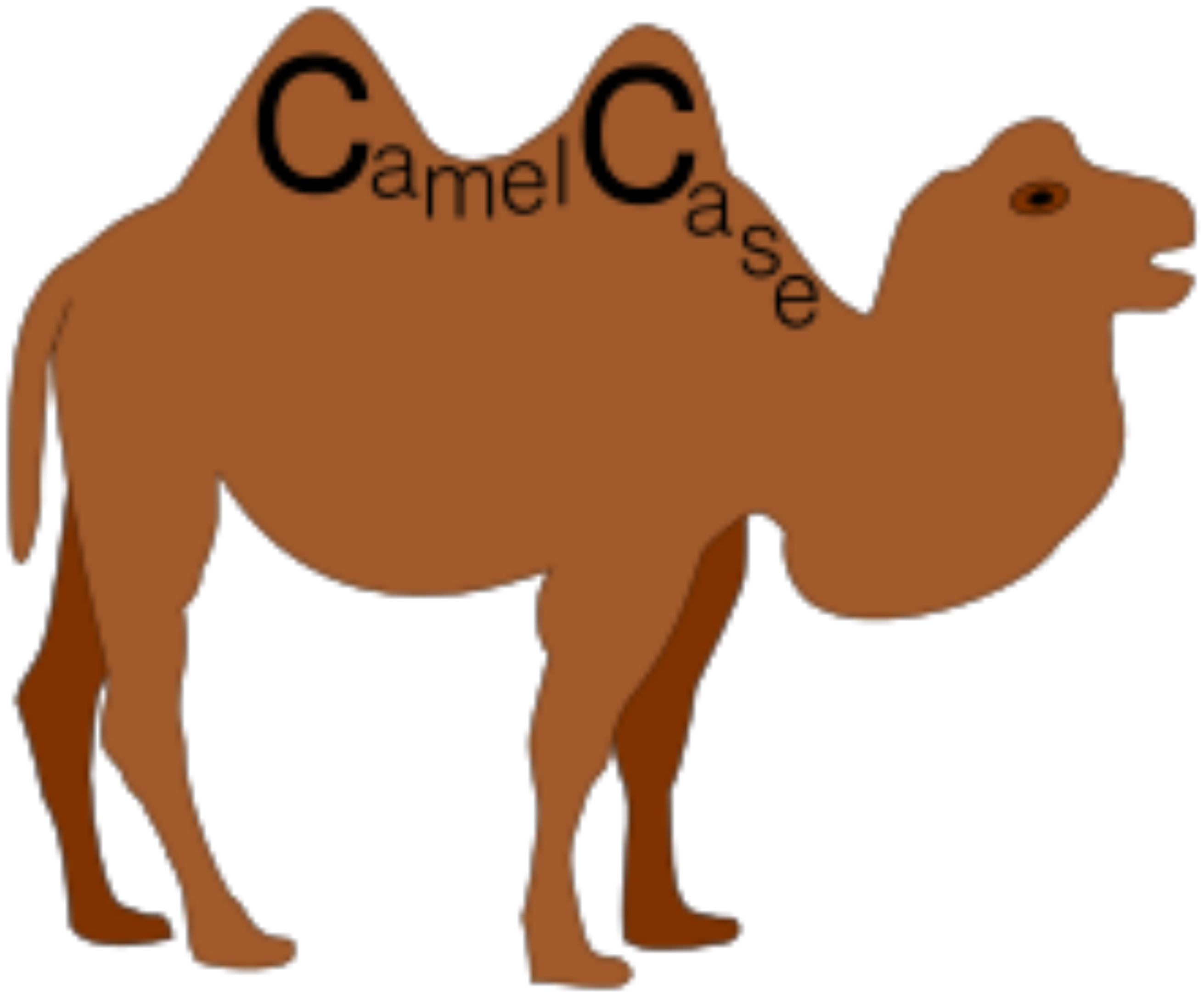
Naming Variables



a
age
delta
name1
name2
R2D2
aVeryLongName

1tooMany

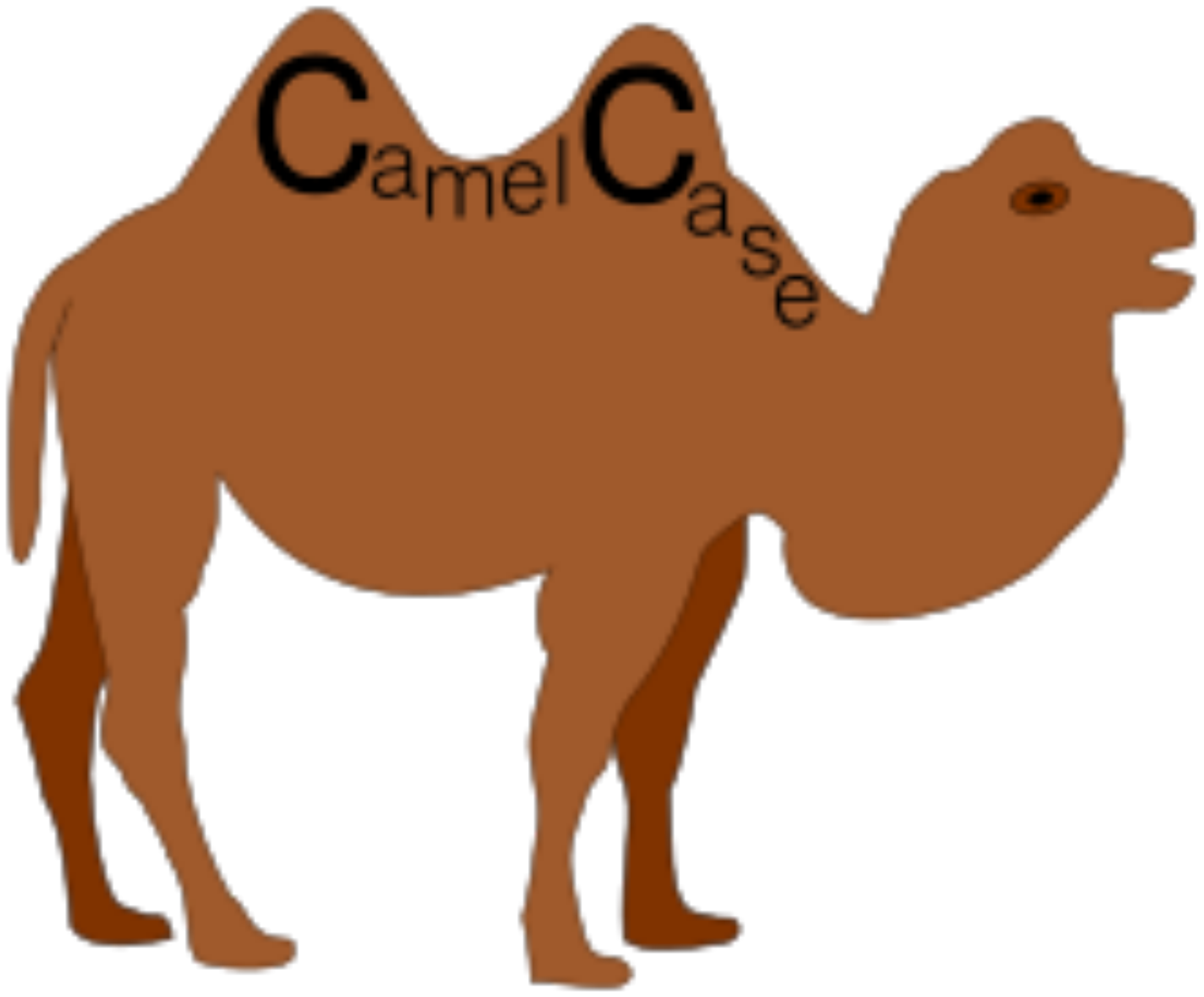
Naming Variables



a
age
delta
name1
name2
R2D2
aVeryLongName

~~1tooMany~~

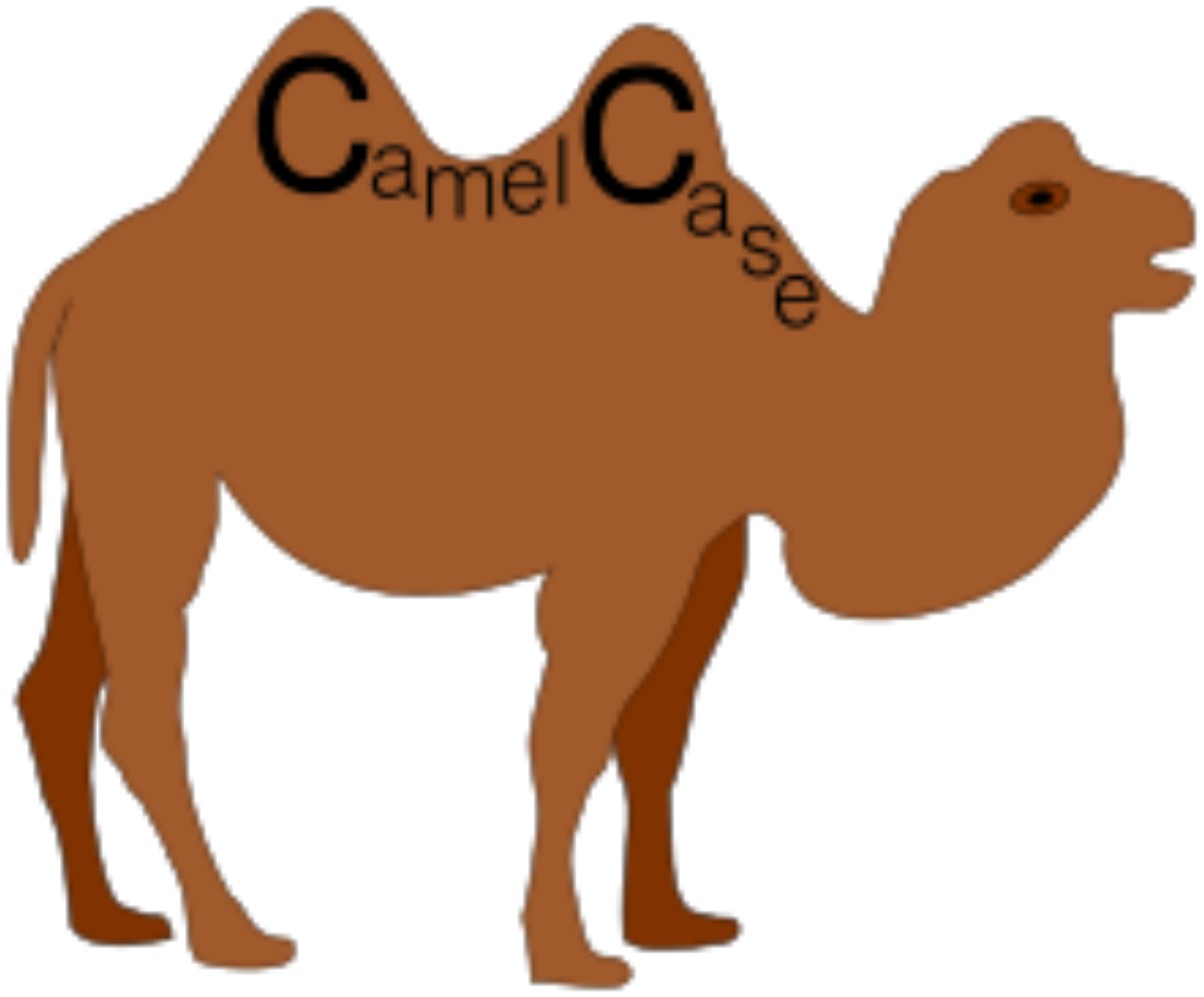
Naming Variables



this_is_good_too
but
wePrefer
thisIsGoodToo

lambda
for
def

Naming Variables



this_is_good_too
but
wePrefer
thisIsGoodToo

~~lambda~~
~~for~~
~~def~~

Exercise 1

```
***
Mae
*****
Alice
*******
Felicia
```

Exercise 2 (Tricky and Unfair)

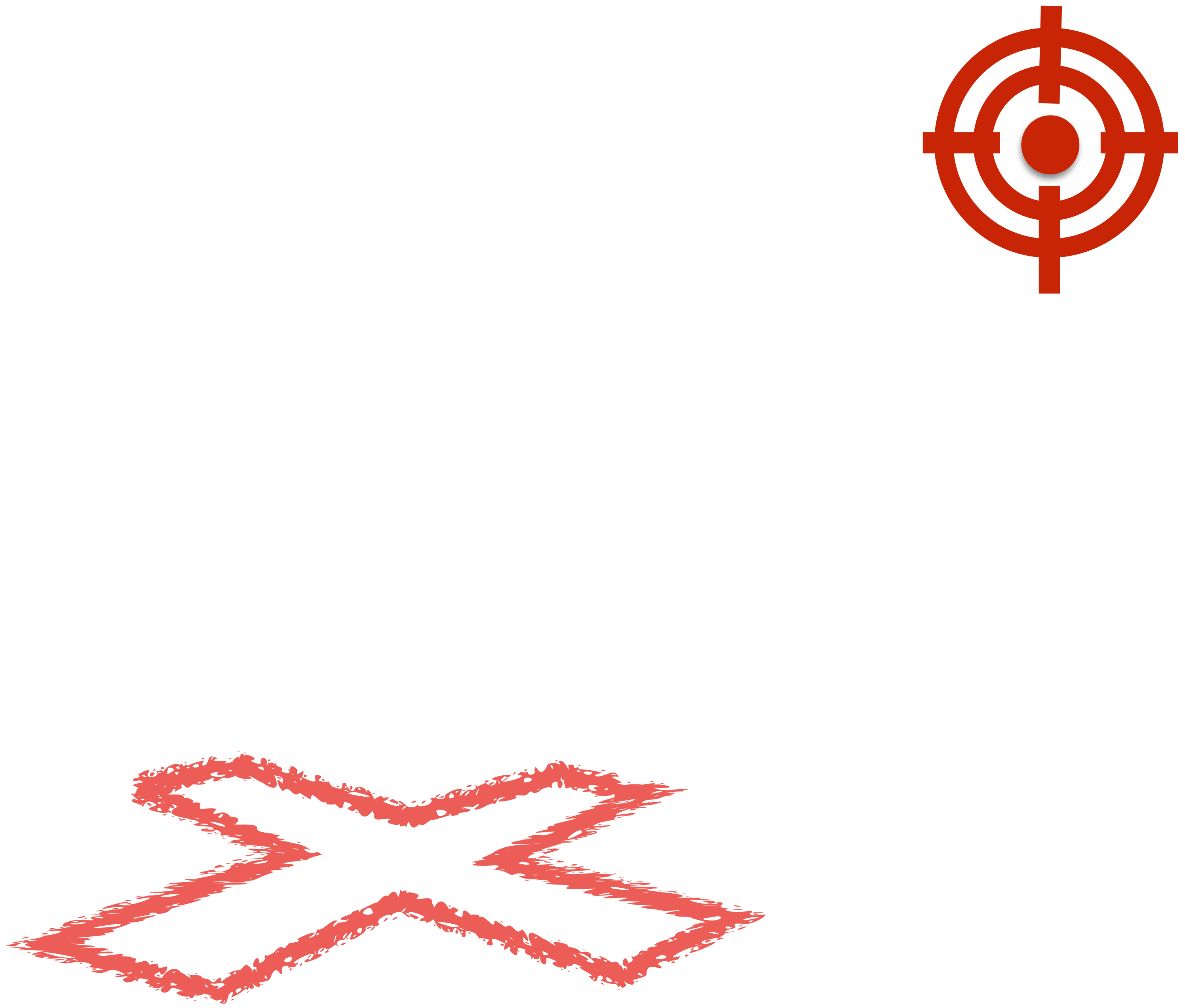
```
*
Mae
********
Alice
****
Felicia
**
```

We stopped here last time…

Outline

- The Programming Process
- Memory: RAM
- Variables revisited
  - Literals: numbers, strings, lists
  - Types: type( )
  - Multiple assignments
  - Operators. Overloaded operators.
- Loops
  - range( ); list( )
- Programming exercises

The Programming Process

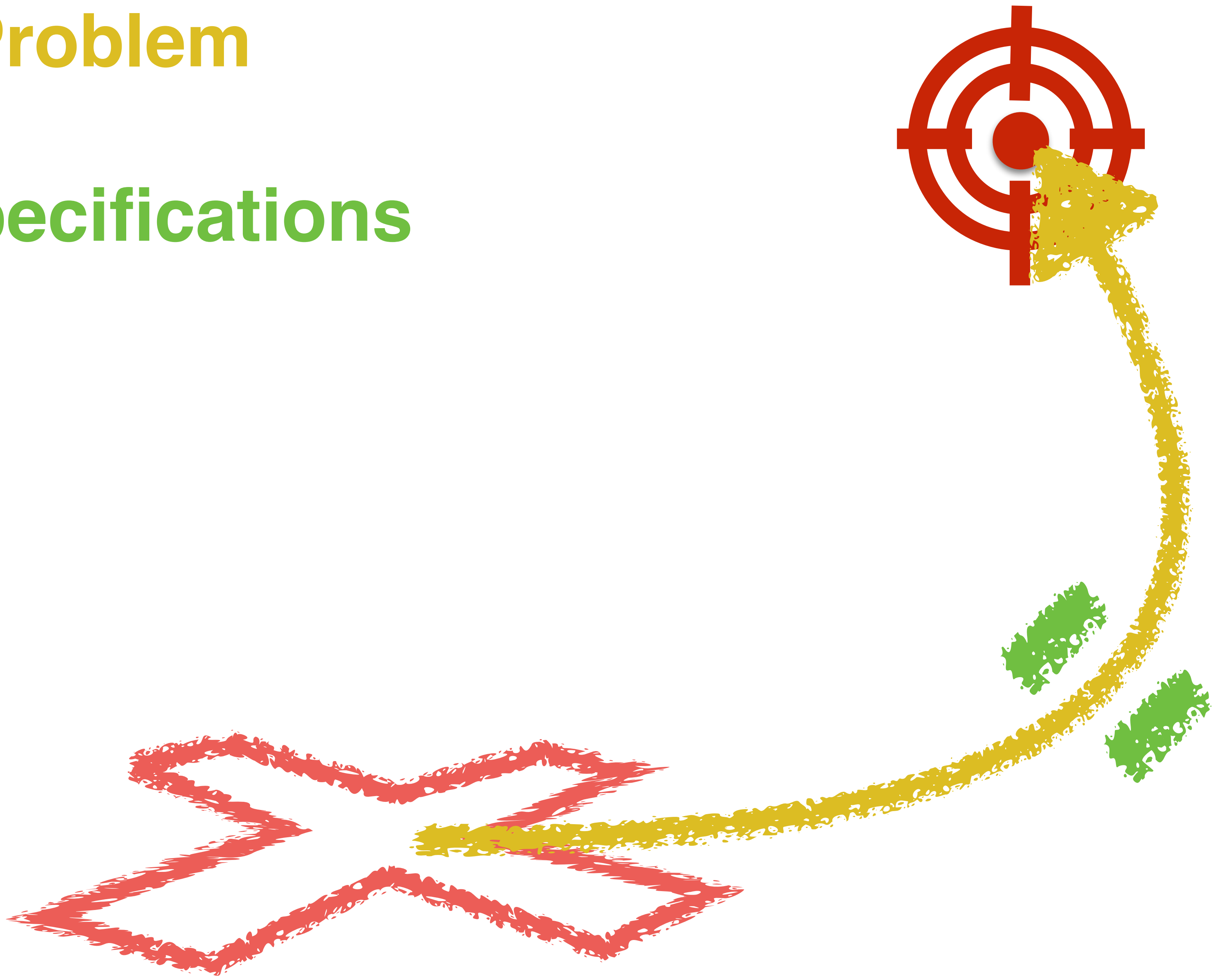


The Programming Process

- Analyze the **Problem**

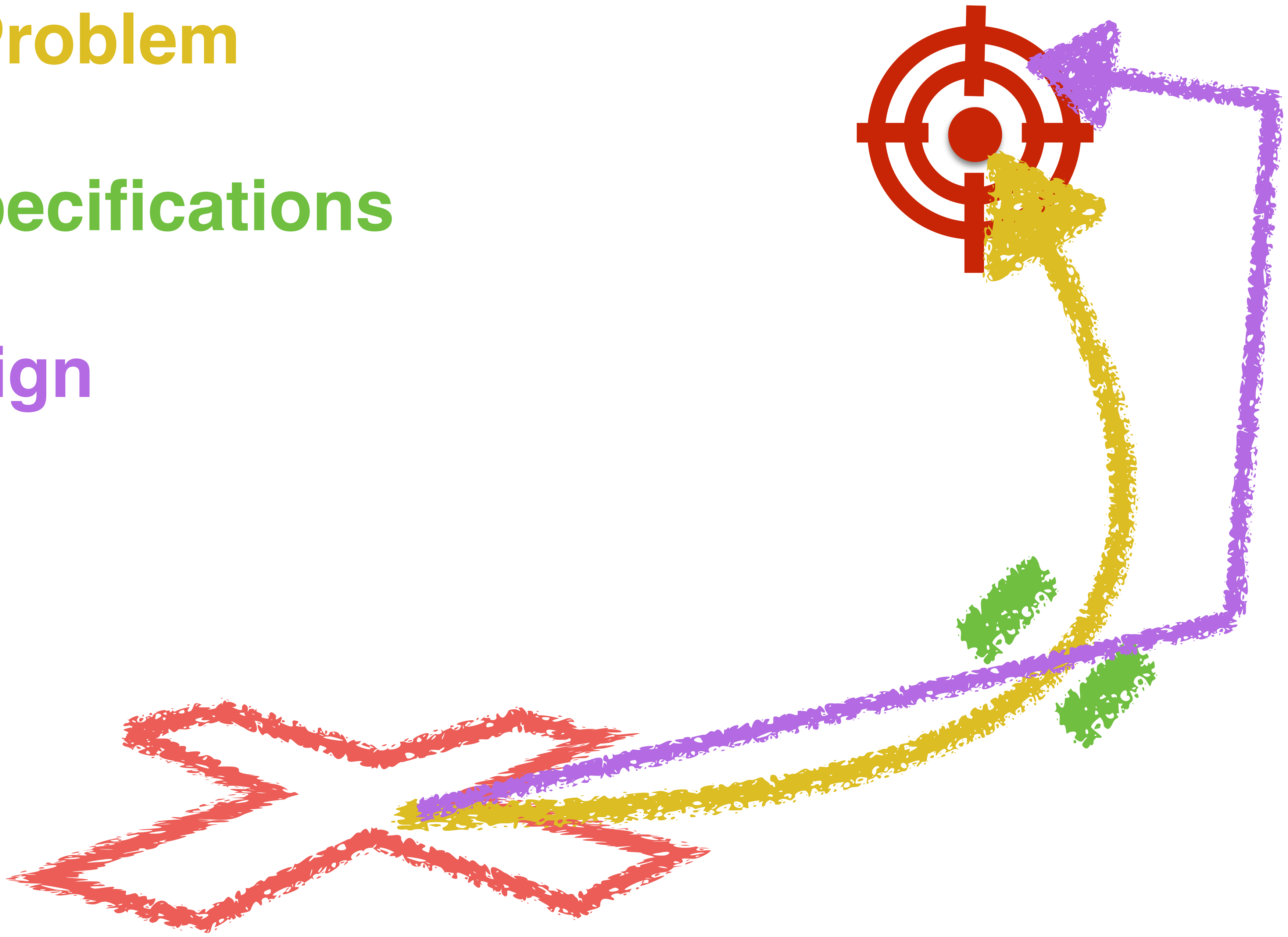
The Programming Process

- Analyze the **Problem**
- Determine **Specifications**



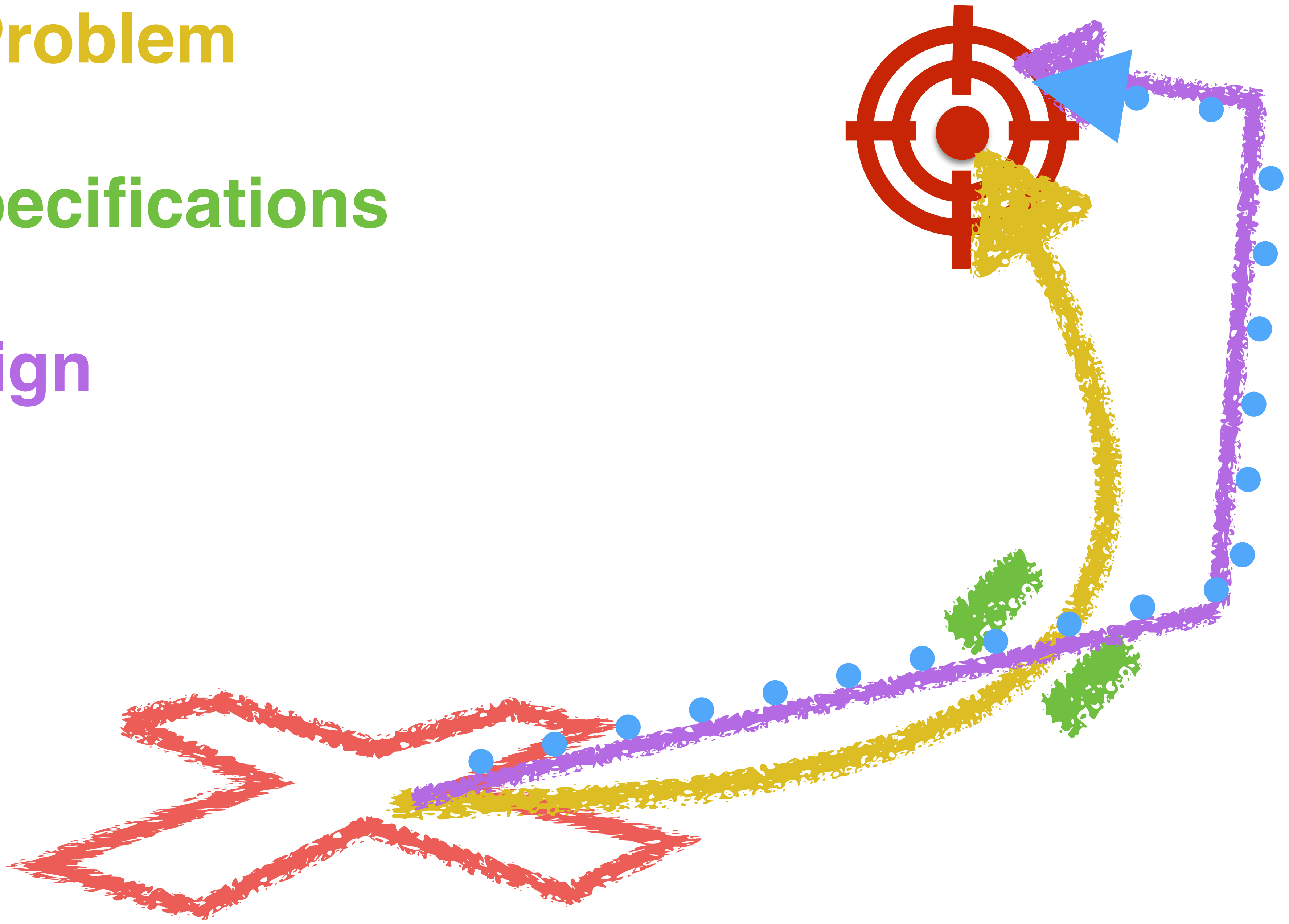
The Programming Process

- Analyze the **Problem**
- Determine **Specifications**
- Create a **Design**



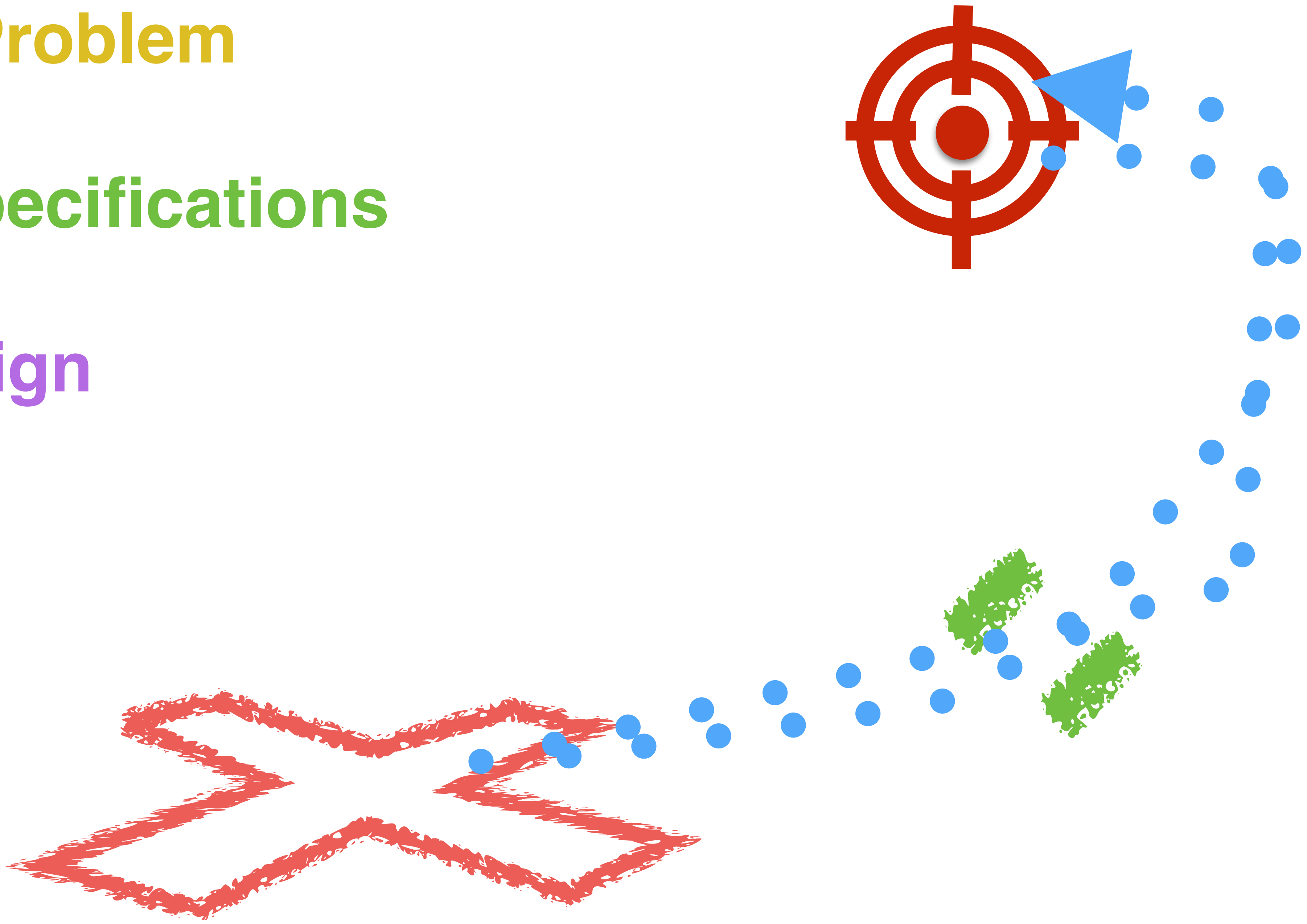
The Programming Process

- Analyze the **Problem**
- Determine **Specifications**
- Create a **Design**
- **Implement**



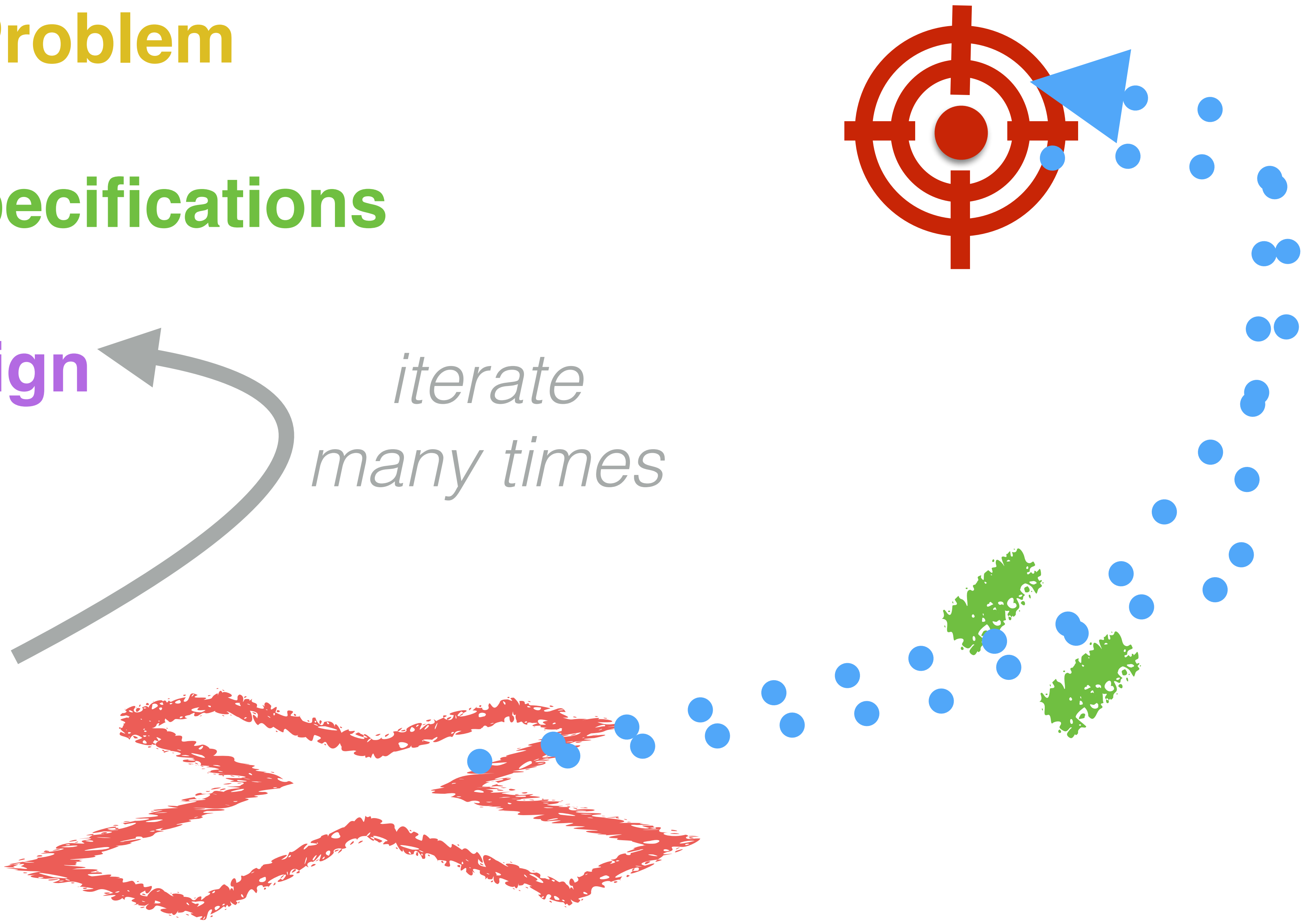
The Programming Process

- Analyze the **Problem**
- Determine **Specifications**
- Create a **Design**
- **Implement**
- Test & Debug



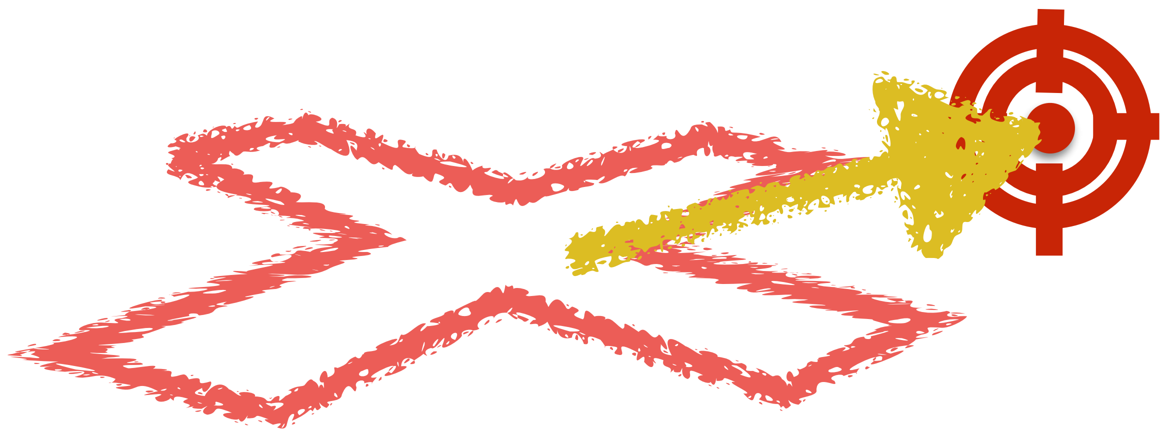
The Programming Process

- Analyze the **Problem**
- Determine **Specifications**
- Refine the ~~Create a~~ **Design**
- **Implement**
- Test & Debug

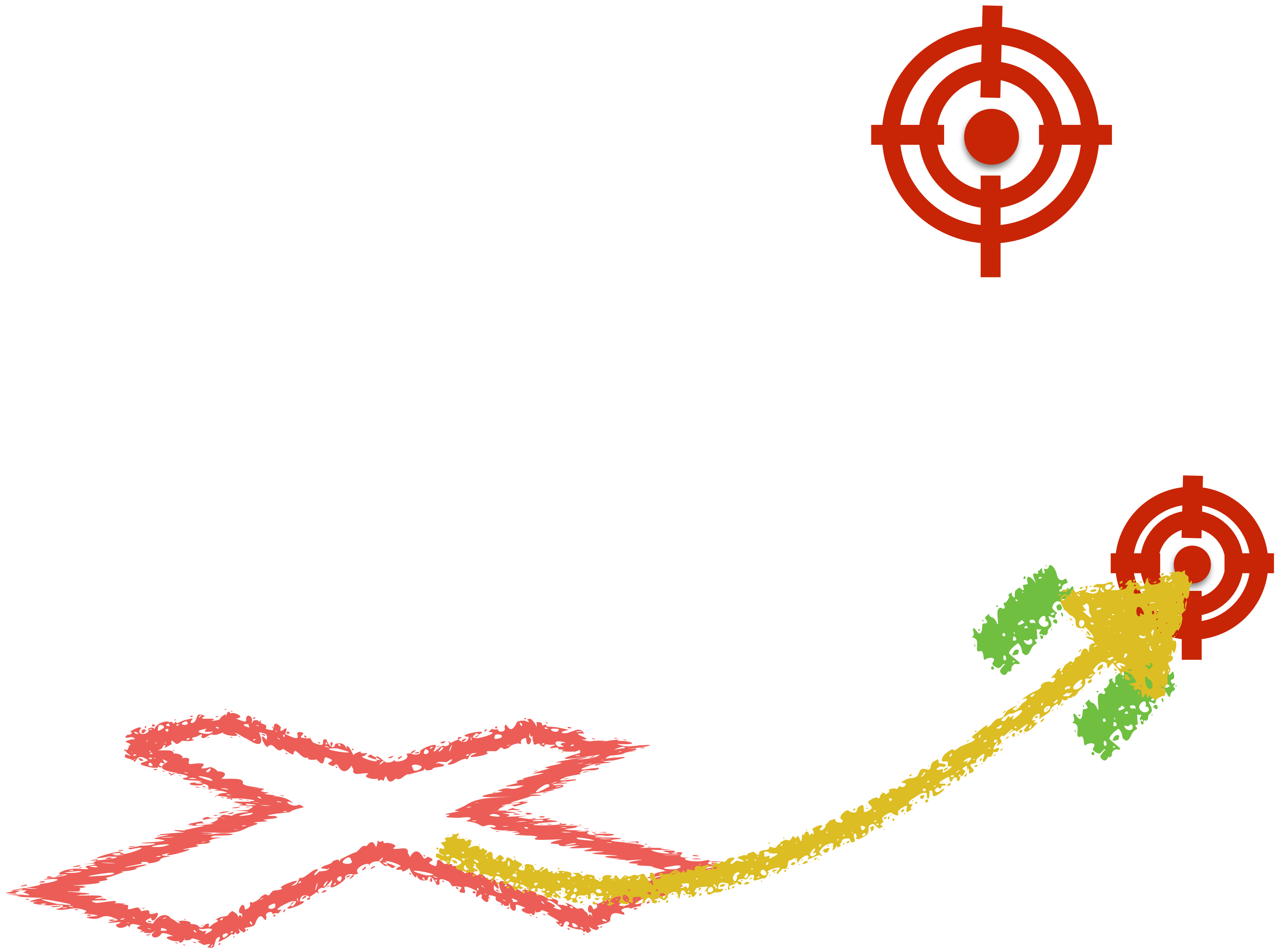


The Programming Process

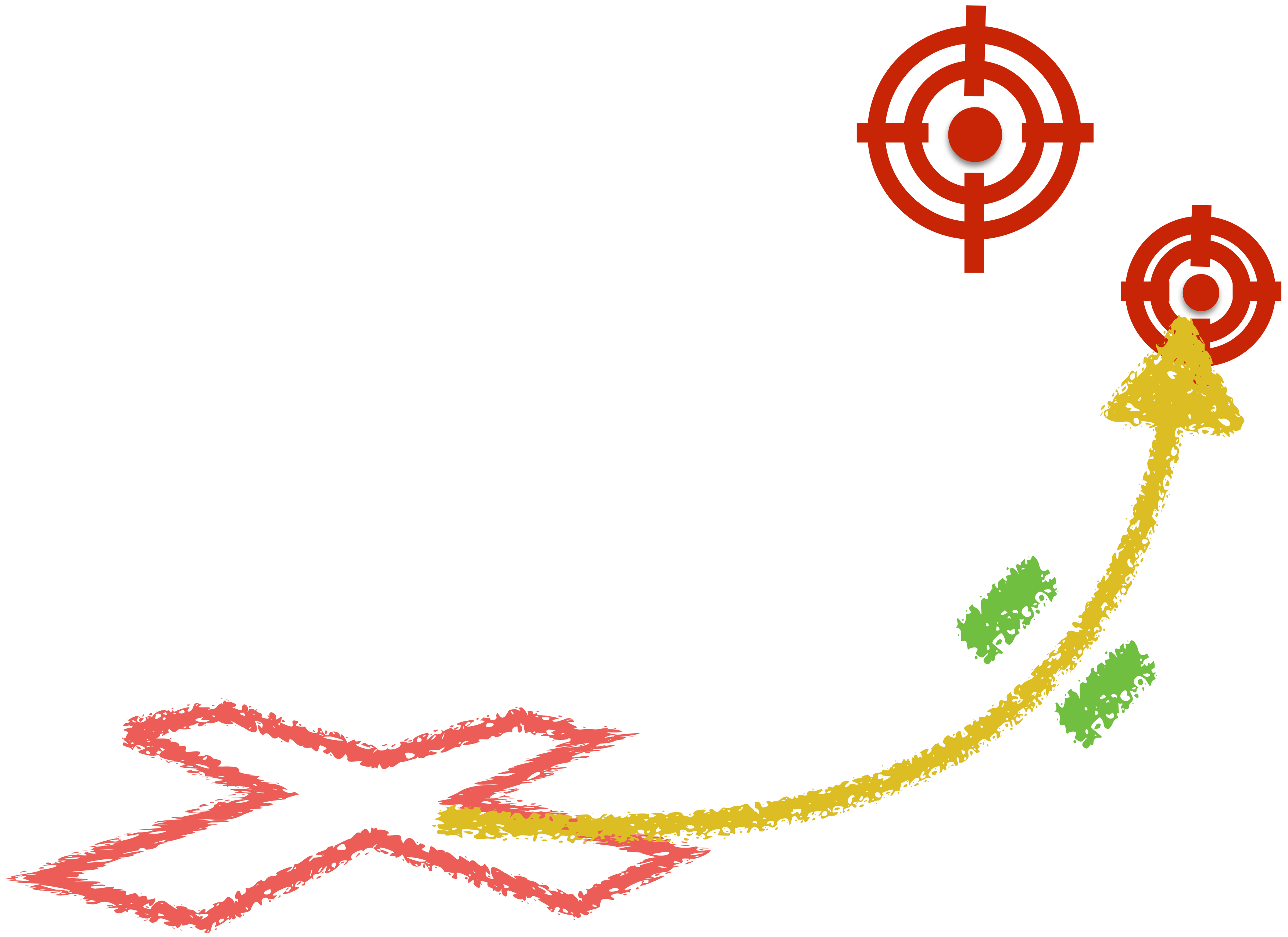
The Programming Process



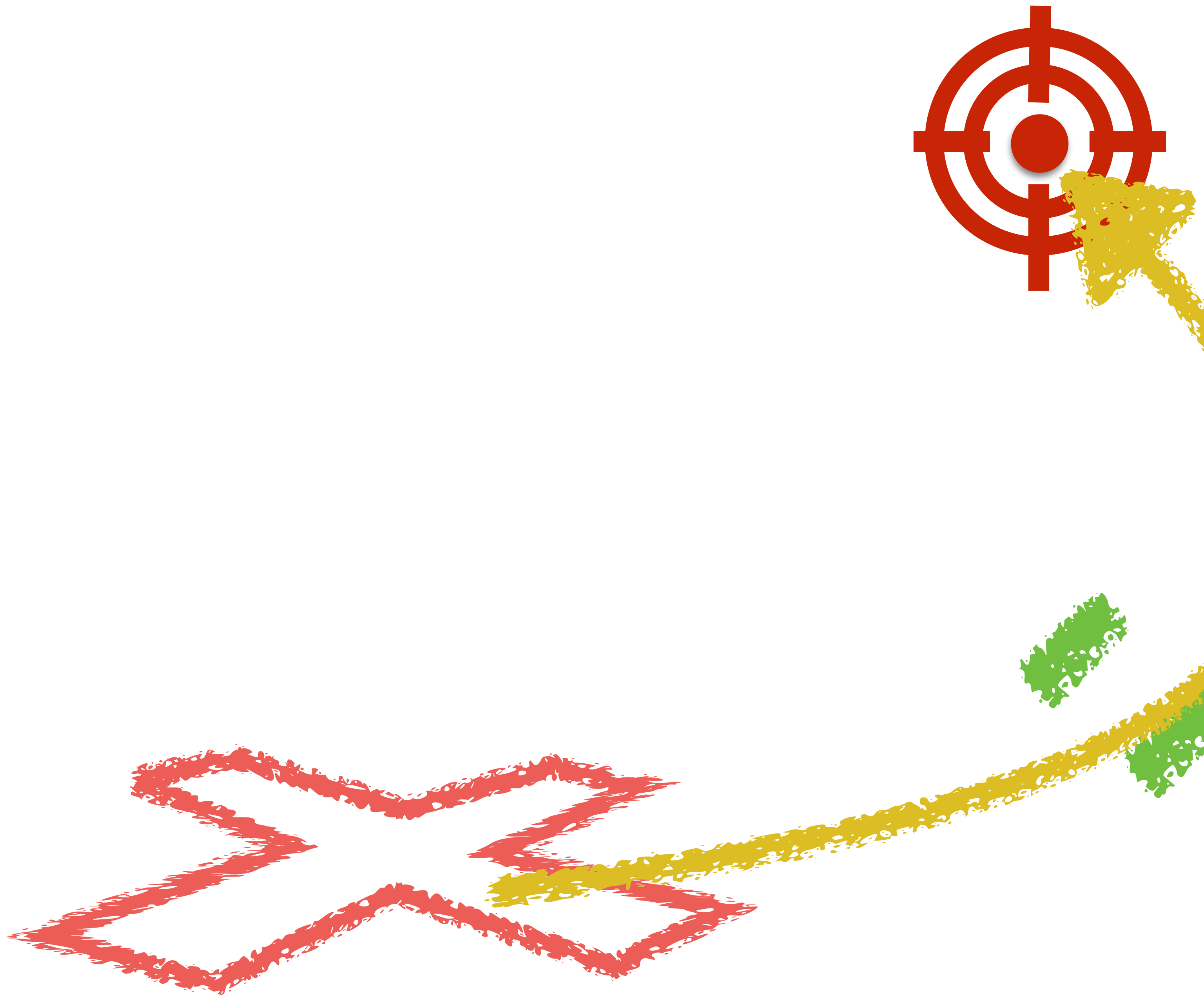
The Programming Process



The Programming Process

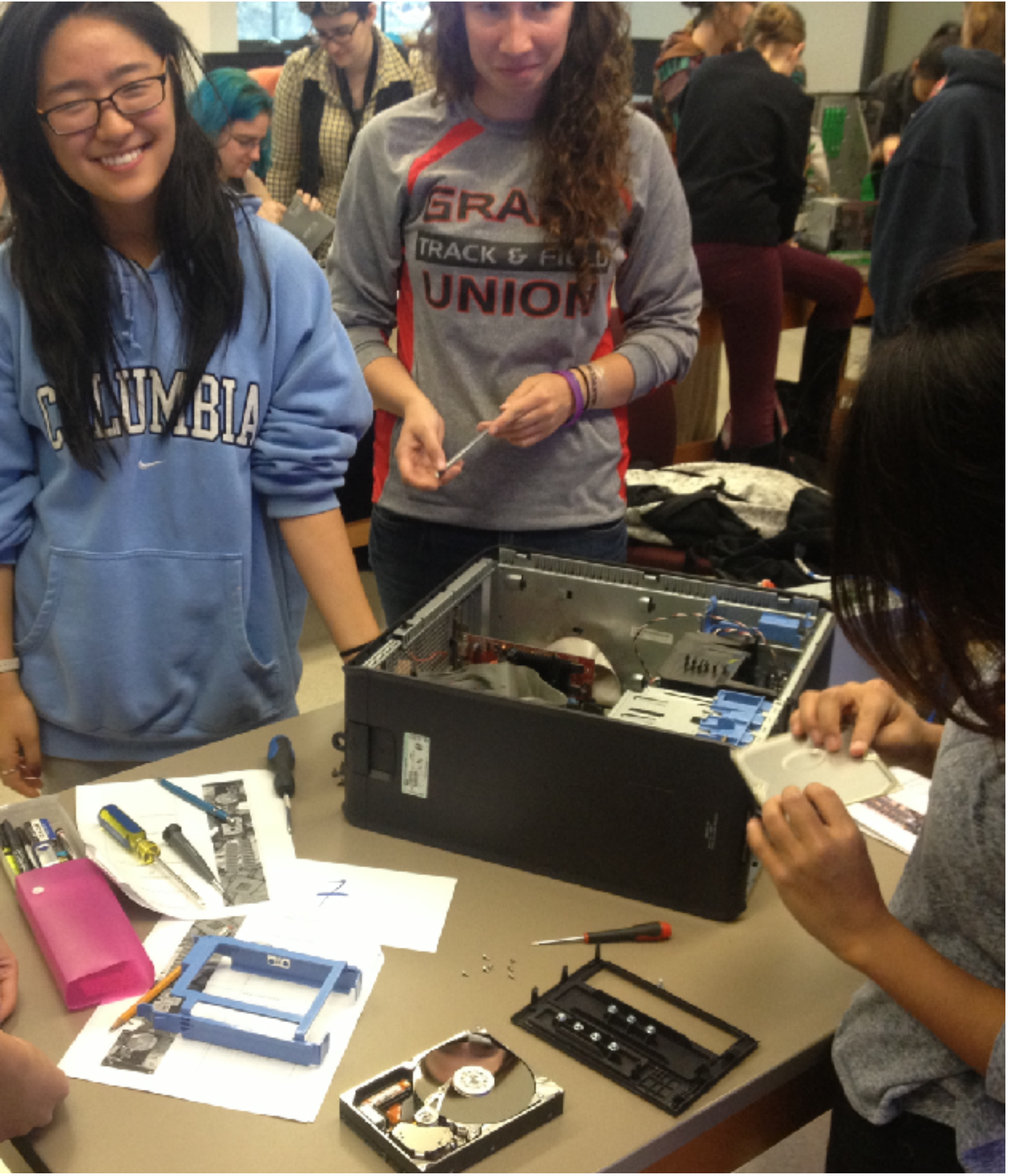


The Programming Process



Back to the Memory

What does the memory really look like?

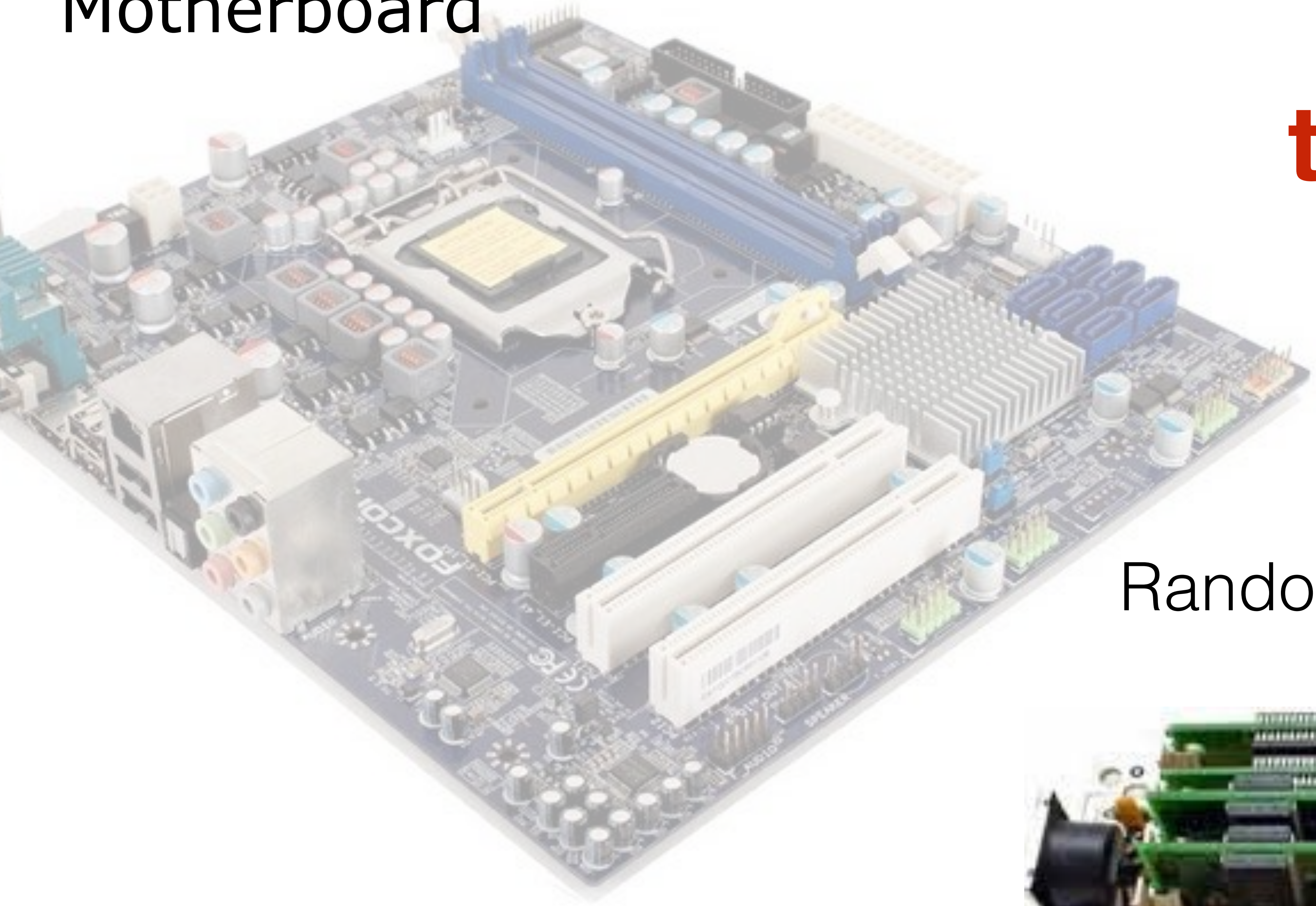


What does the memory really look like?

Motherboard

What does the memory really look like?

Motherboard

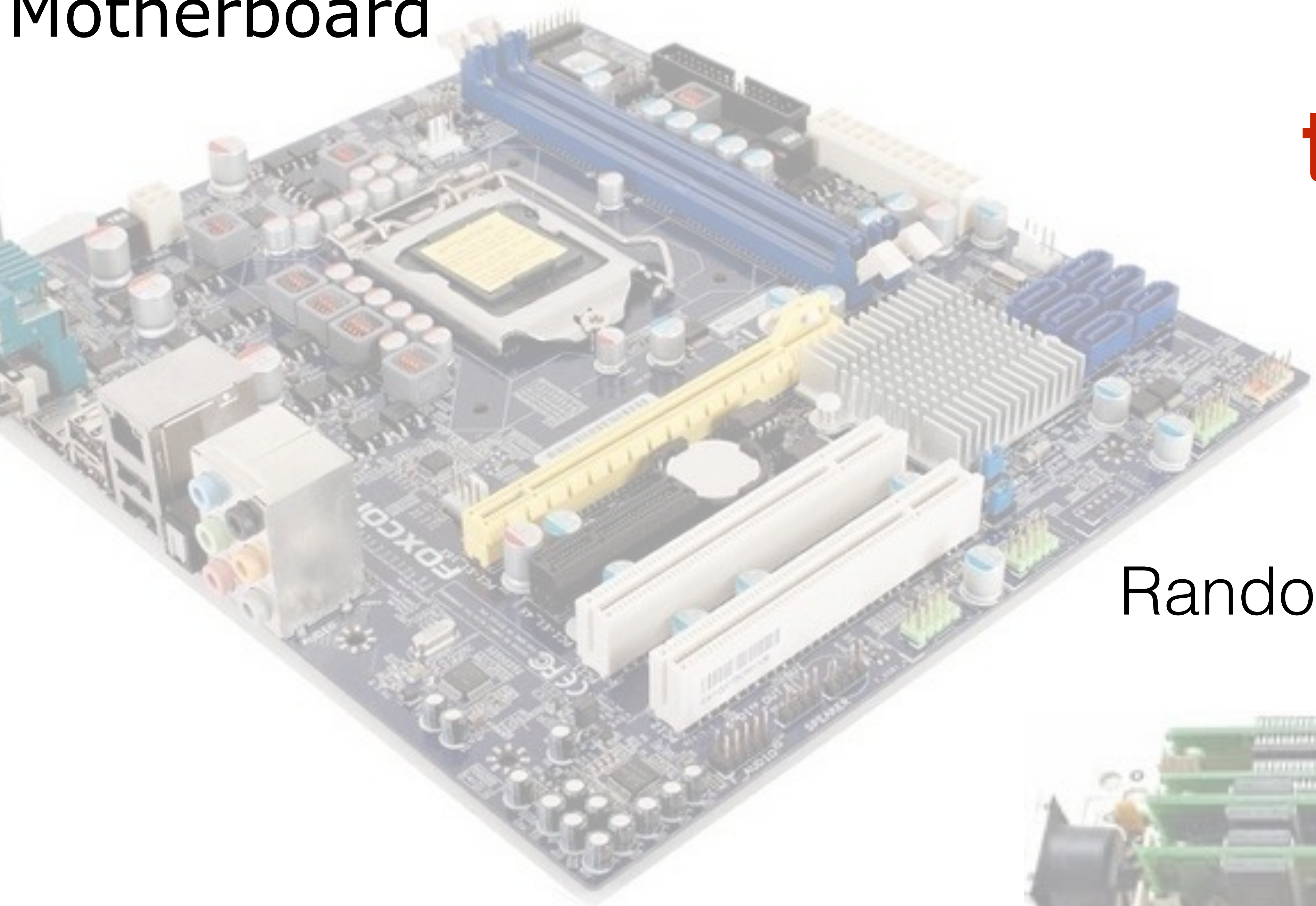


What does the memory really look like?

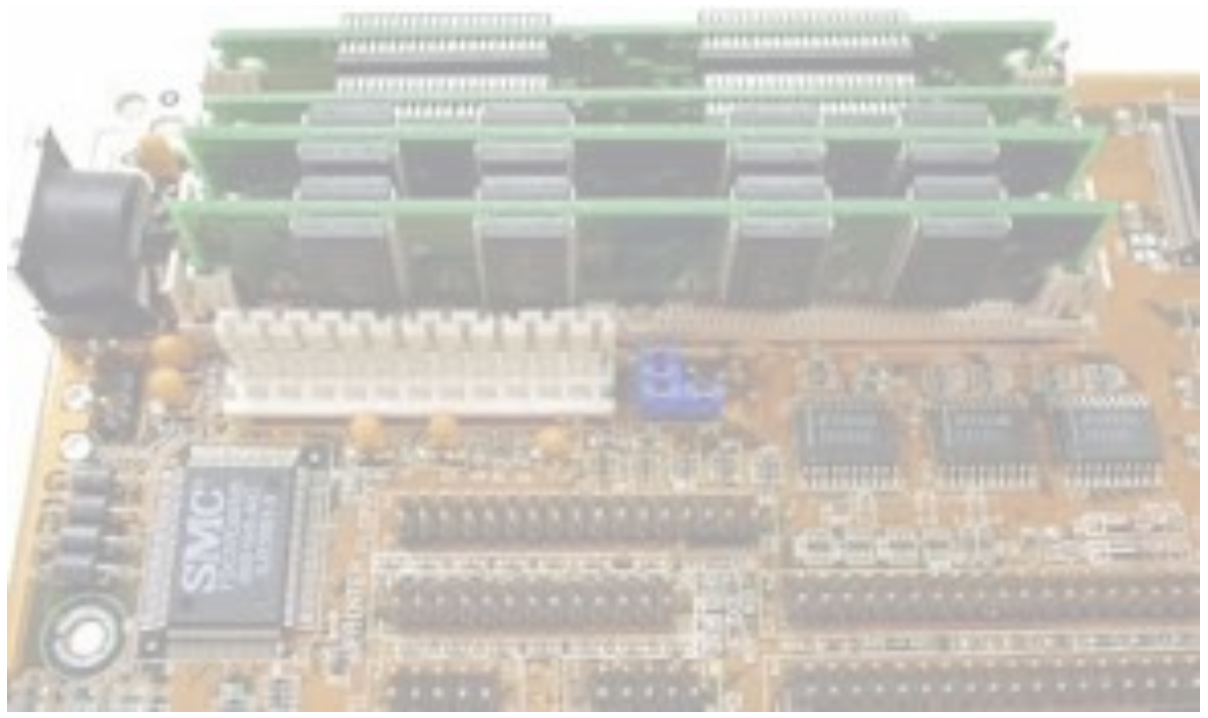
Random Access Memory (RAM)

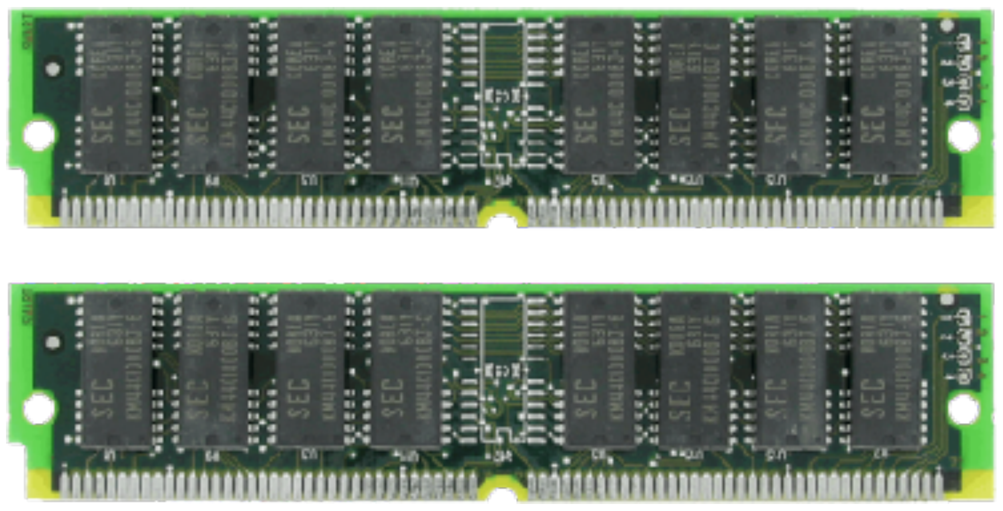
Motherboard

What does the memory really look like?



Random Access Memory (RAM)

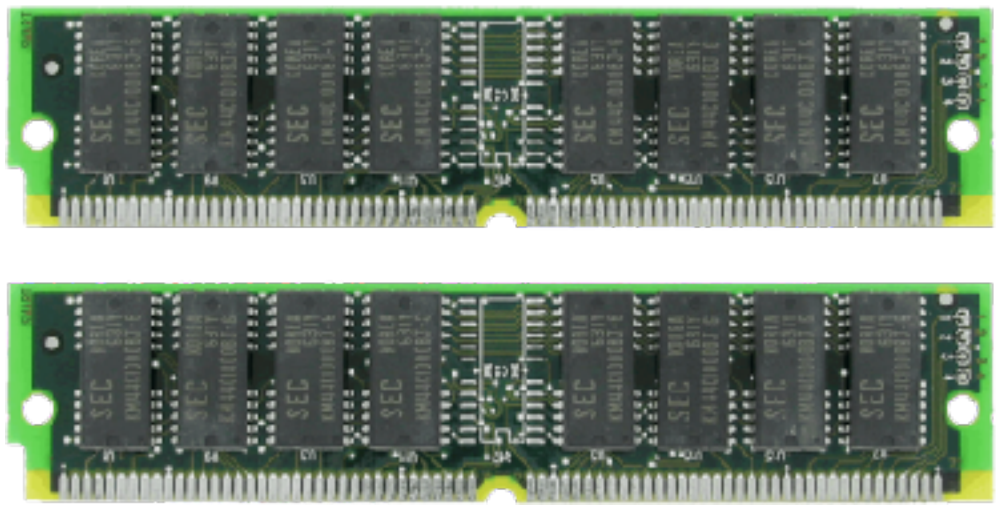




Single In-line Memory Module (SIMM)

What does the memory really look like?

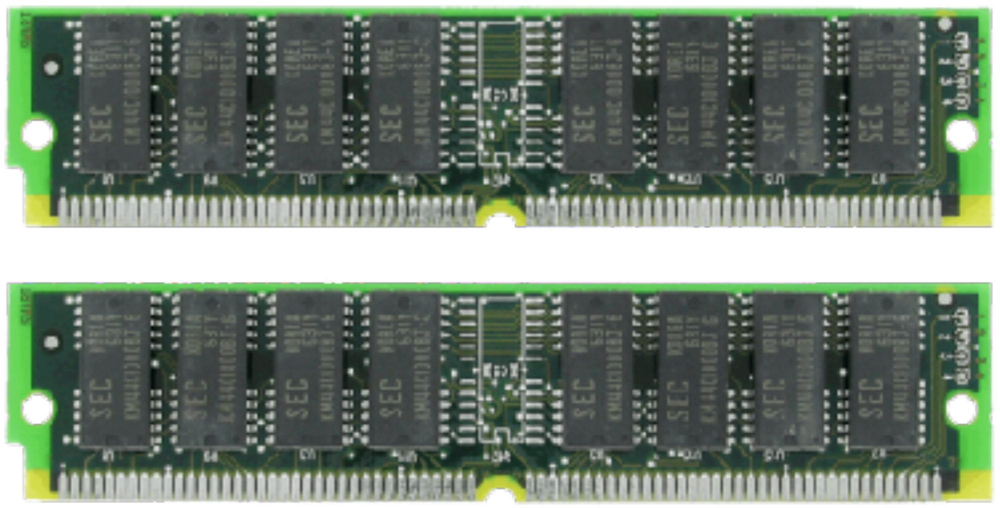
- RAM: 4, 8, 12, 16 GigaBytes
- **Giga** = billion: 10^9 bytes
- In RAM: room for approximately **2 billion** integers



- 1 number takes **4 bytes**
- 1 character takes **1 bytes (sometimes 2 bytes)**

How big is 2 Billion?

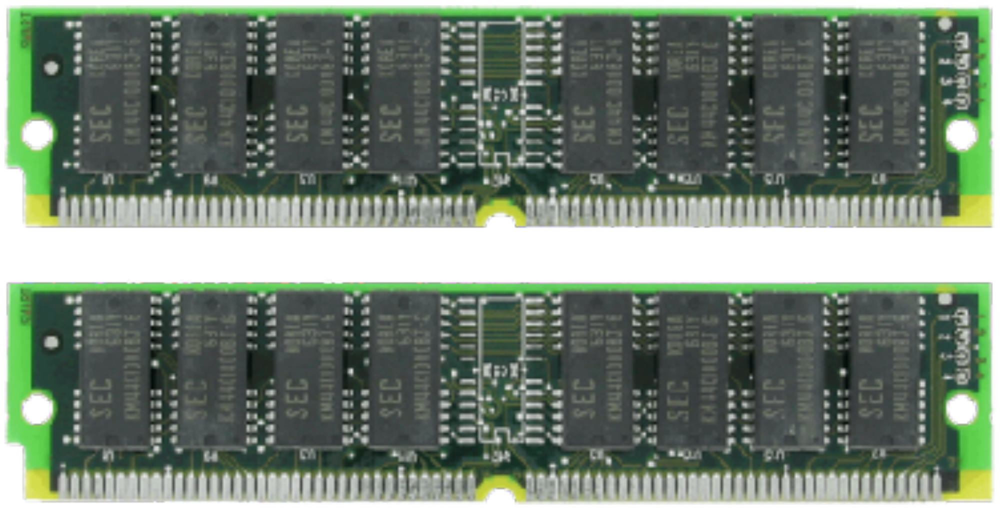
2 billion integers



How tall are 2 billion quarters

How big is 2 Billion?

2 billion integers

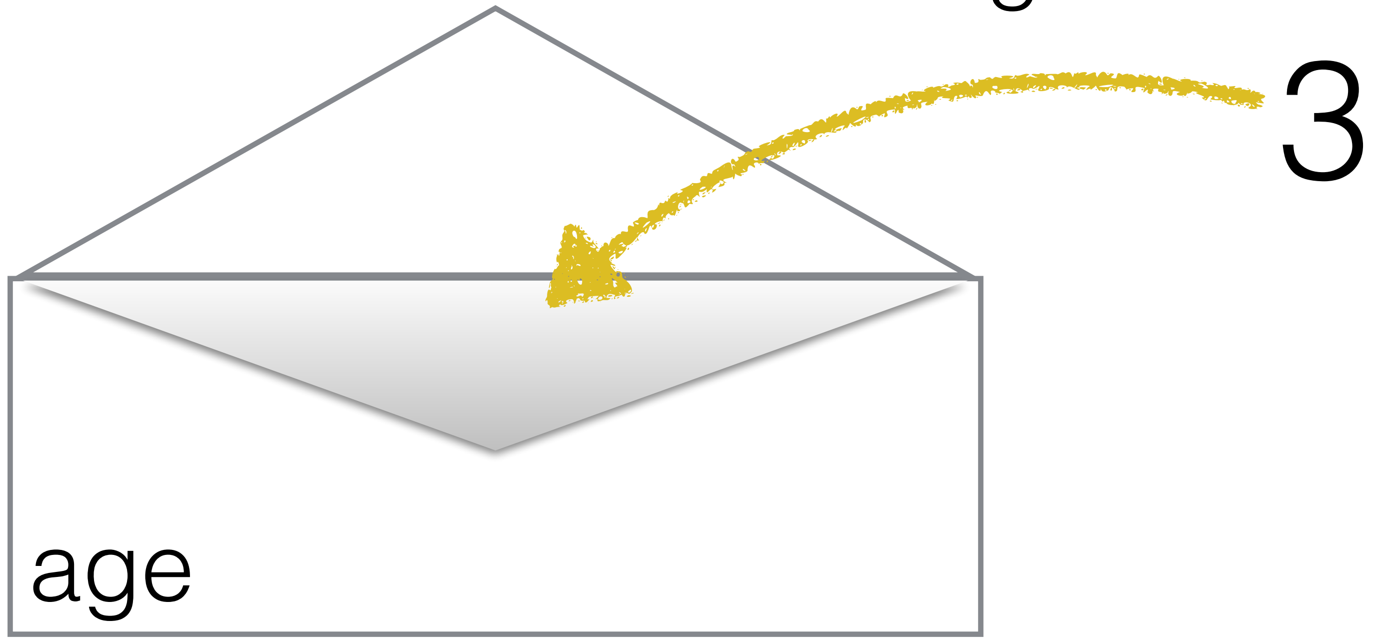


2 miles, or 3.2 km !

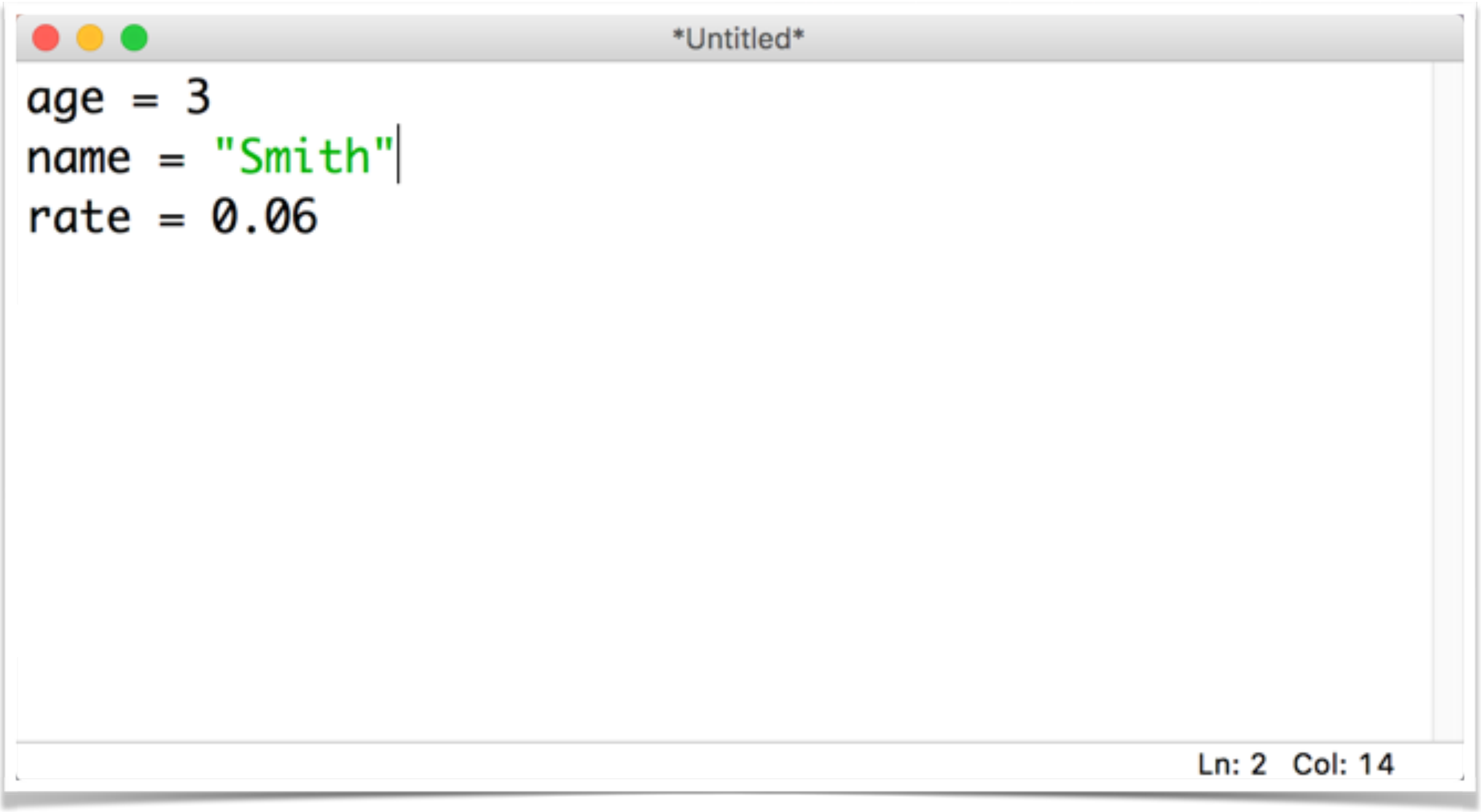
How tall are 2 billion quarters

Variables and Assignment

age = 3

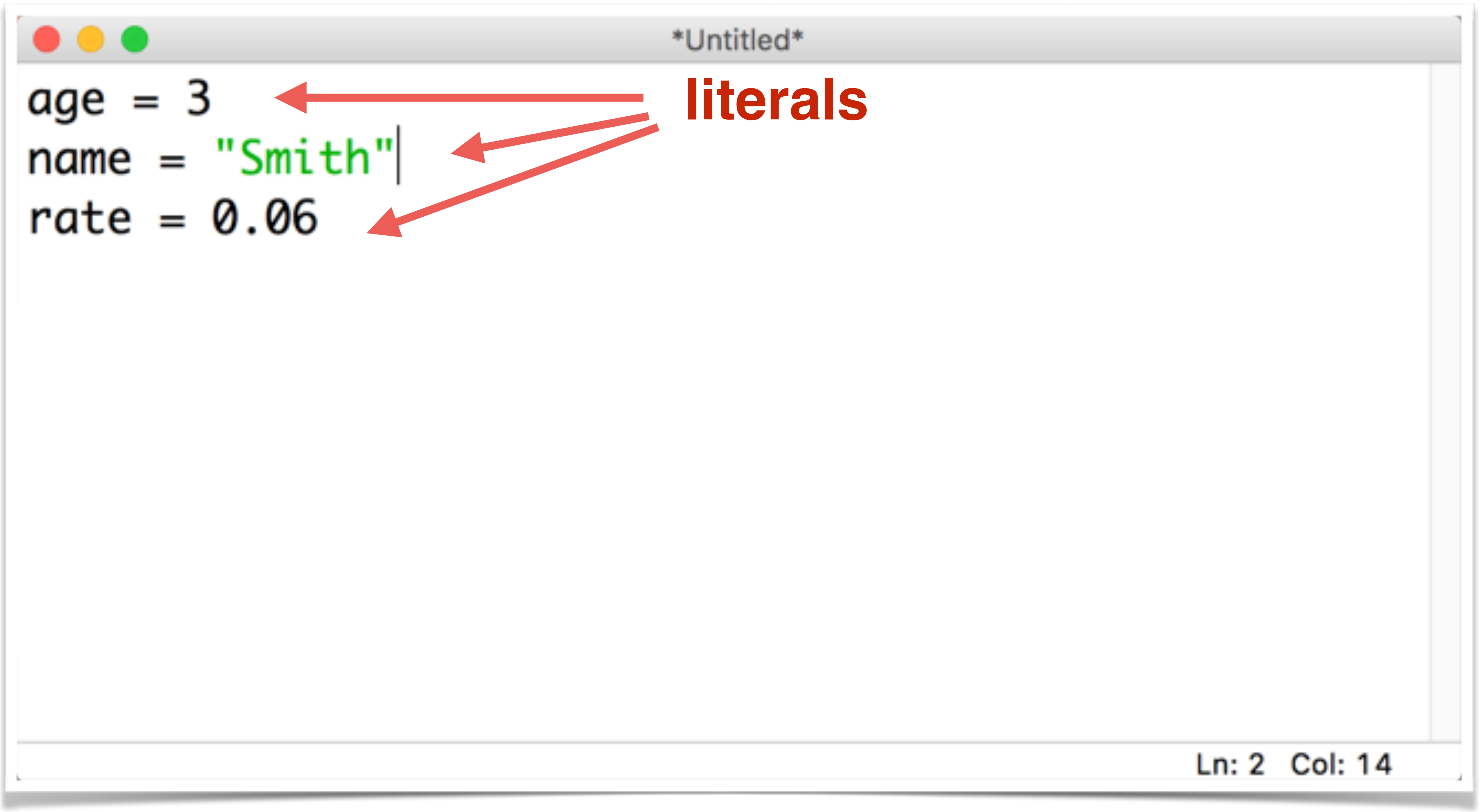


Variables and Assignment



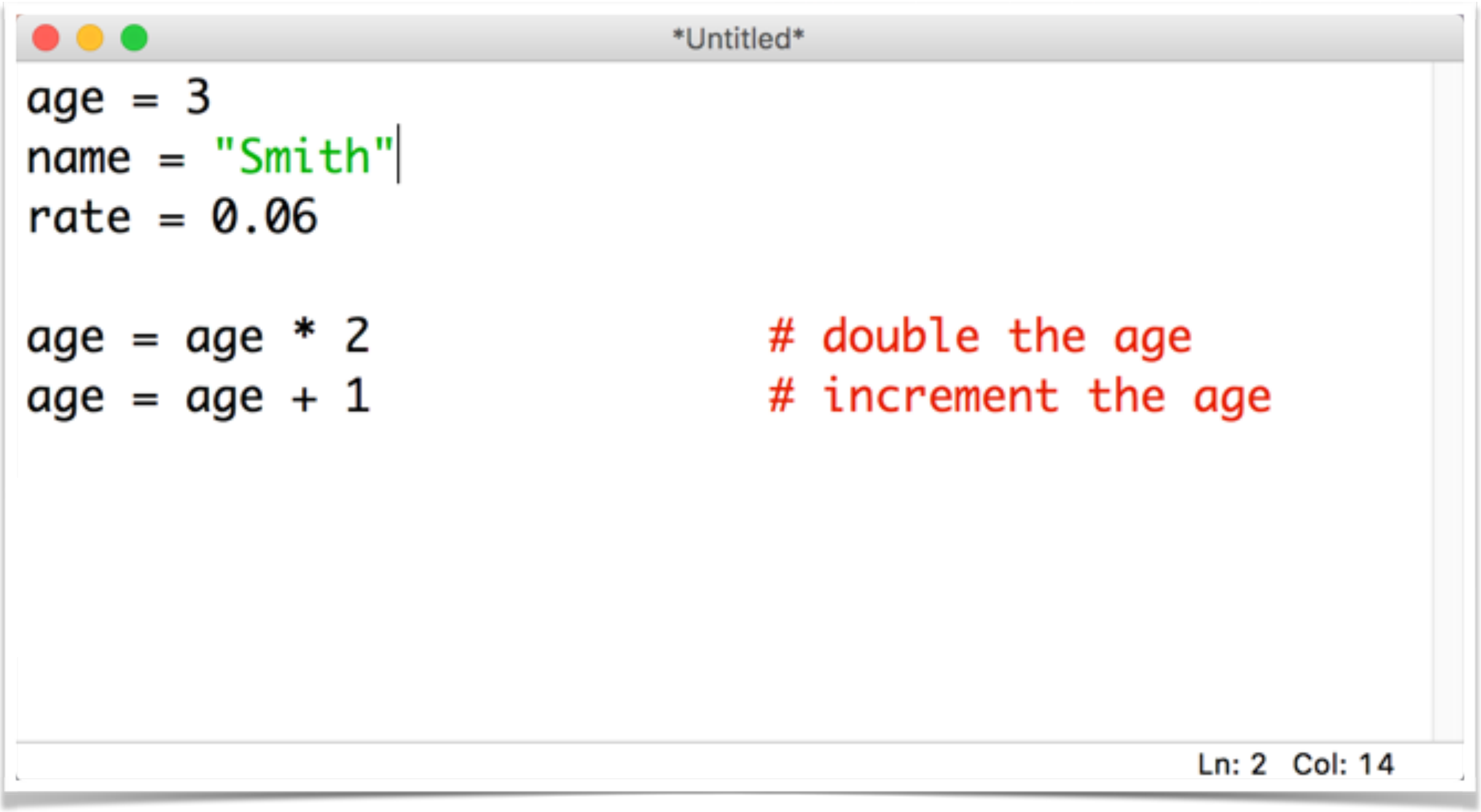
```
age = 3
name = "Smith"
rate = 0.06
```

Variables and Assignment



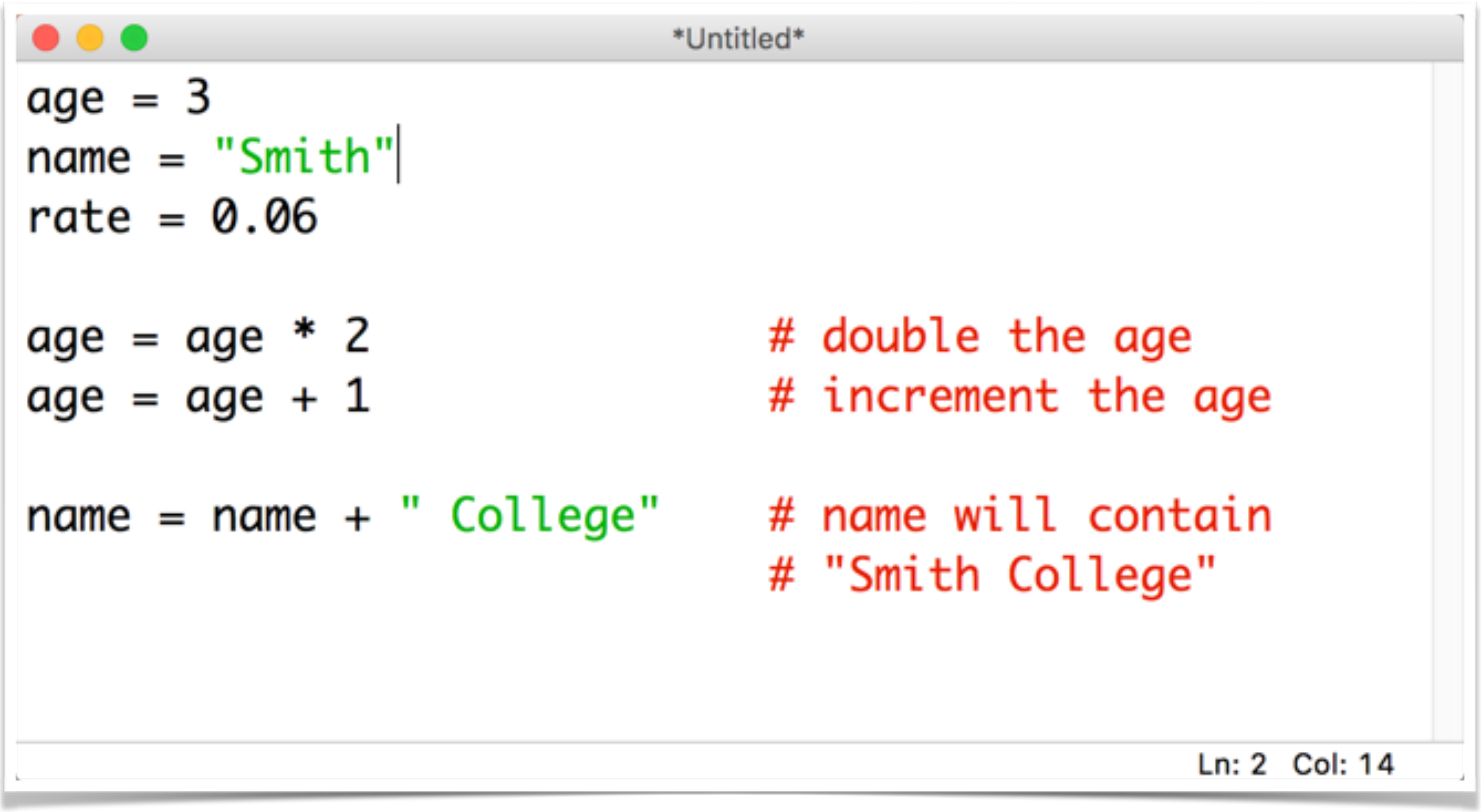
```
age = 3
name = "Smith"
rate = 0.06
```

Variables and Assignment



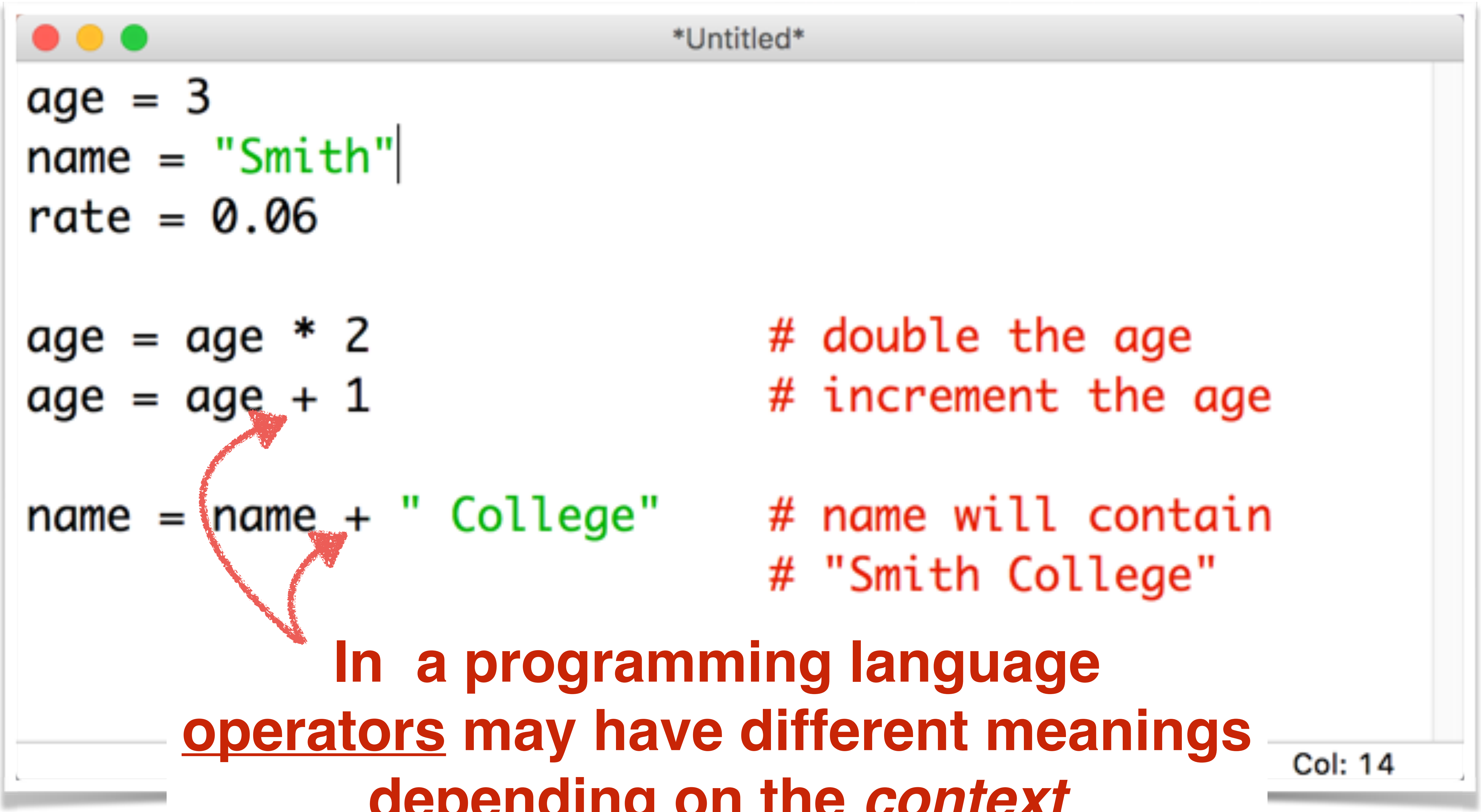
```
age = 3
name = "Smith"
rate = 0.06

age = age * 2                # double the age
age = age + 1                # increment the age
```

Variables and Assignment



```
age = 3
name = "Smith"
rate = 0.06

age = age * 2                 # double the age
age = age + 1                 # increment the age

name = name + " College"      # name will contain
                              # "Smith College"
```

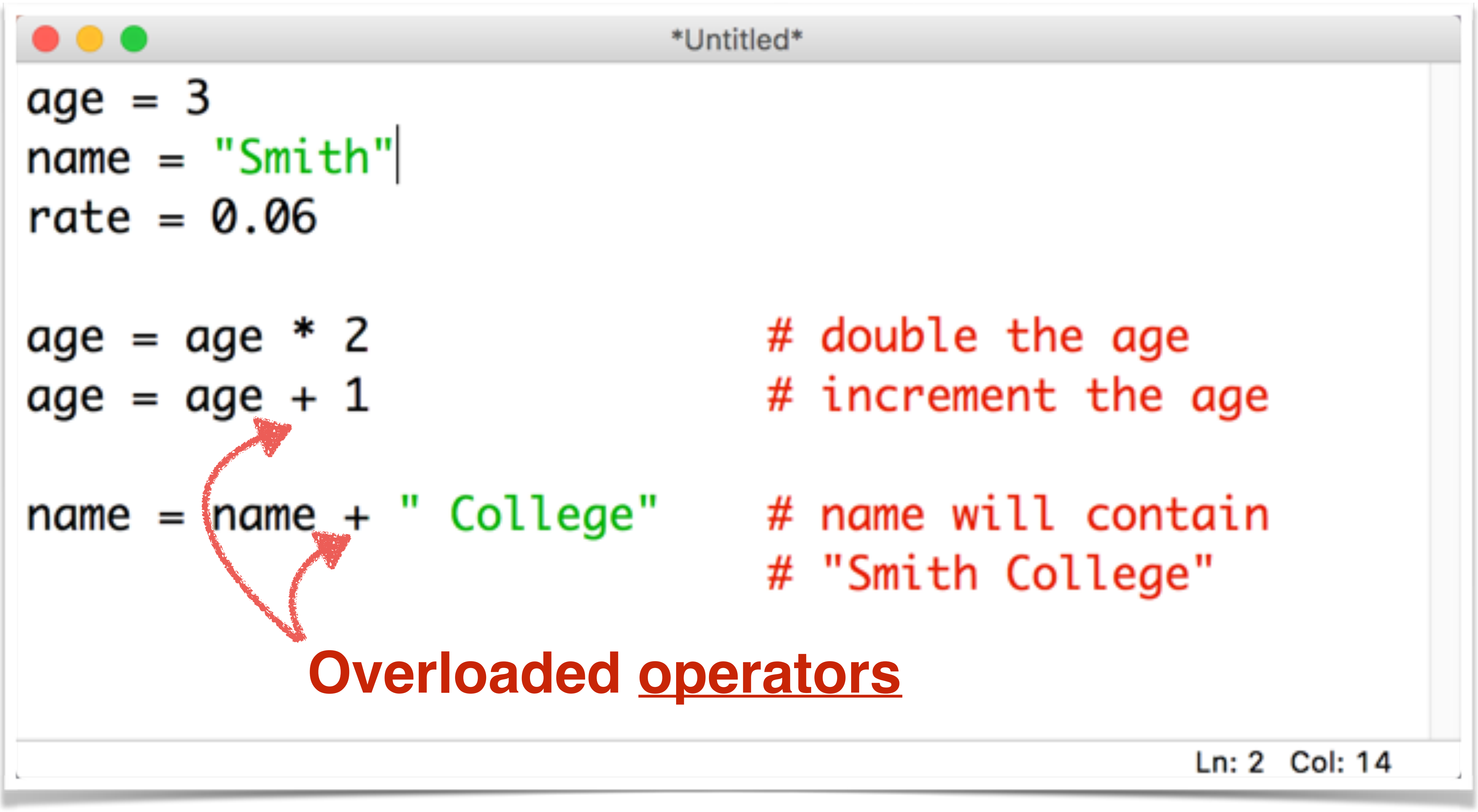
Variables and Assignment



```
age = 3
name = "Smith"
rate = 0.06

age = age * 2                  # double the age
age = age + 1                  # increment the age

name = name + " College"       # name will contain
                               # "Smith College"
```

In a programming language <u>operators</u> may have different meanings depending on the *context*

Variables and Assignment



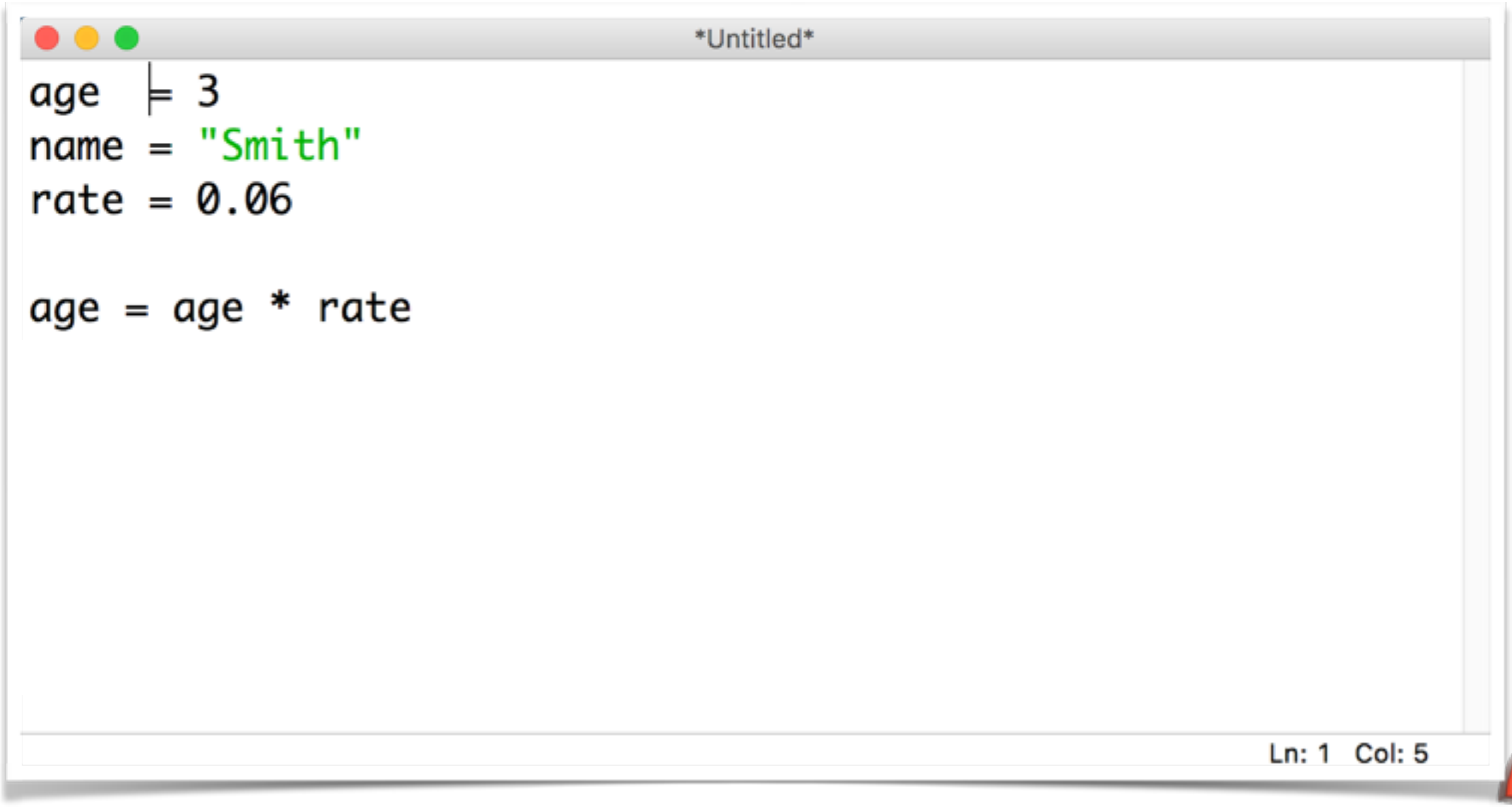
```
age = 3
name = "Smith"
rate = 0.06

age = age * 2                     # double the age
age = age + 1                     # increment the age

name = name + " College"          # name will contain
                                  # "Smith College"
```

Overloaded operators

Exercises

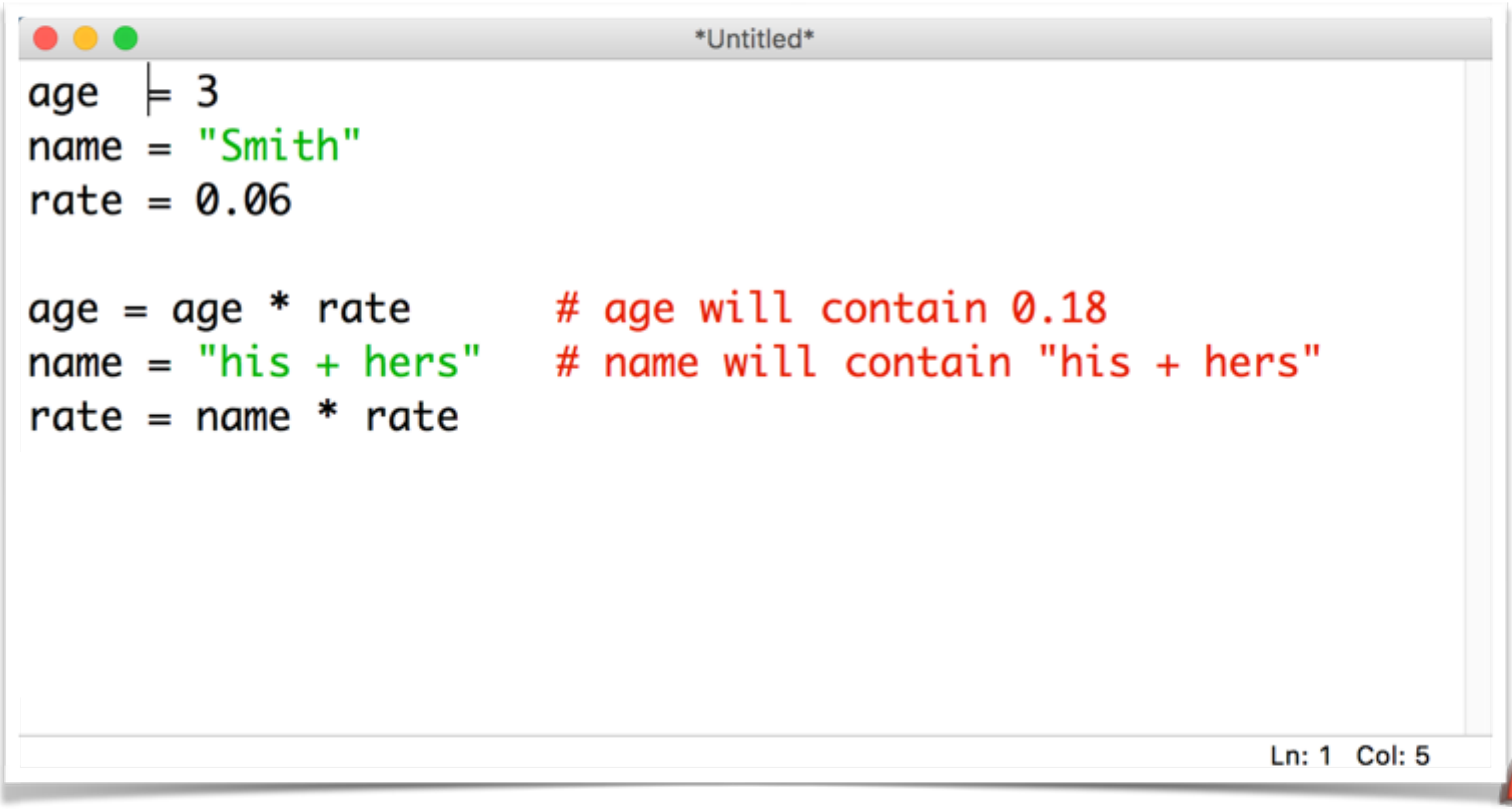
Exercises

Guess what Python will do



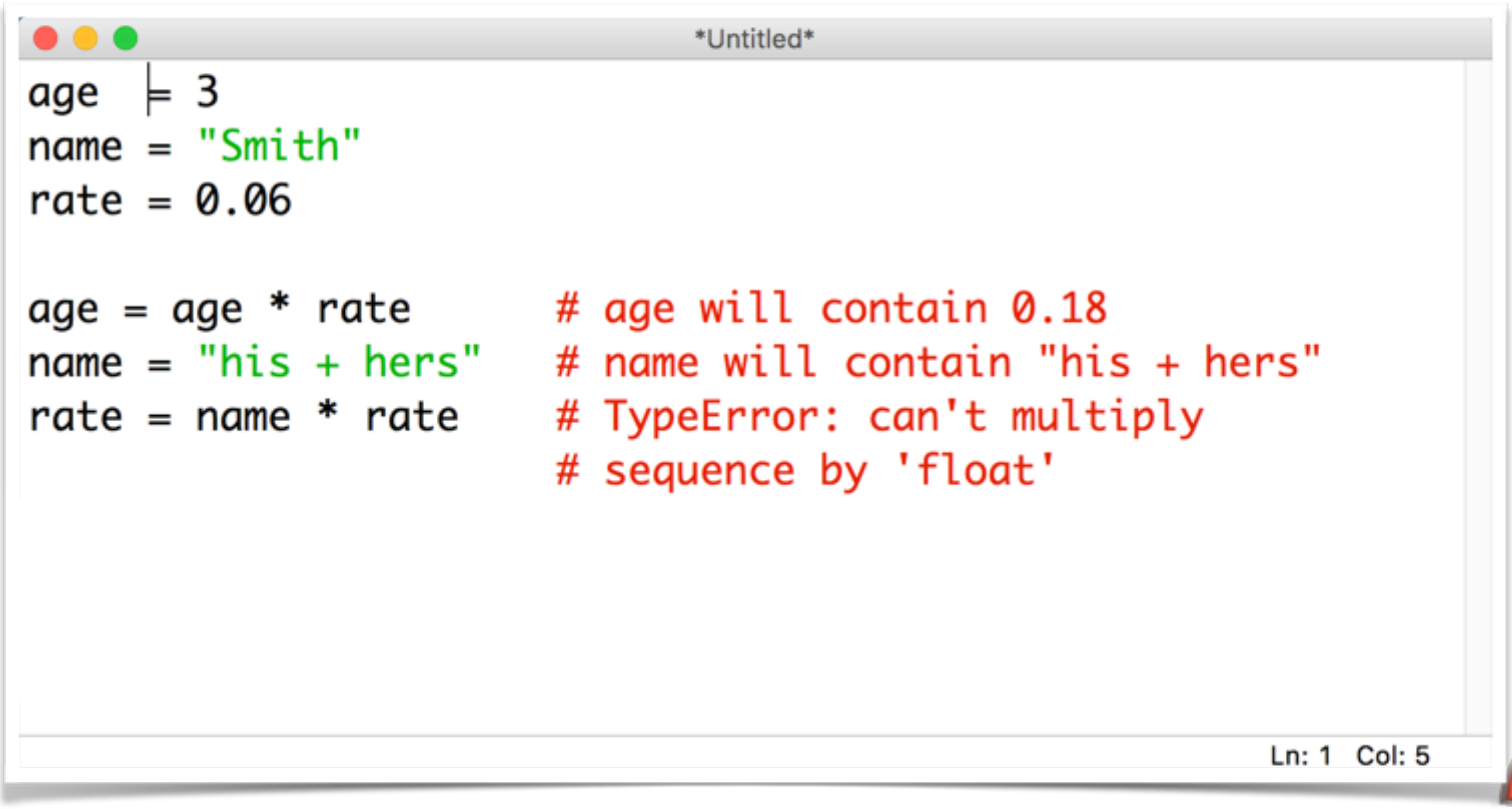
```
age  = 3
name = "Smith"
rate = 0.06

age = age * rate
```

Exercises
Guess what Python will do



```
age  = 3
name = "Smith"
rate = 0.06

age = age * rate      # age will contain 0.18
name = "his + hers"   # name will contain "his + hers"
rate = name * rate
```

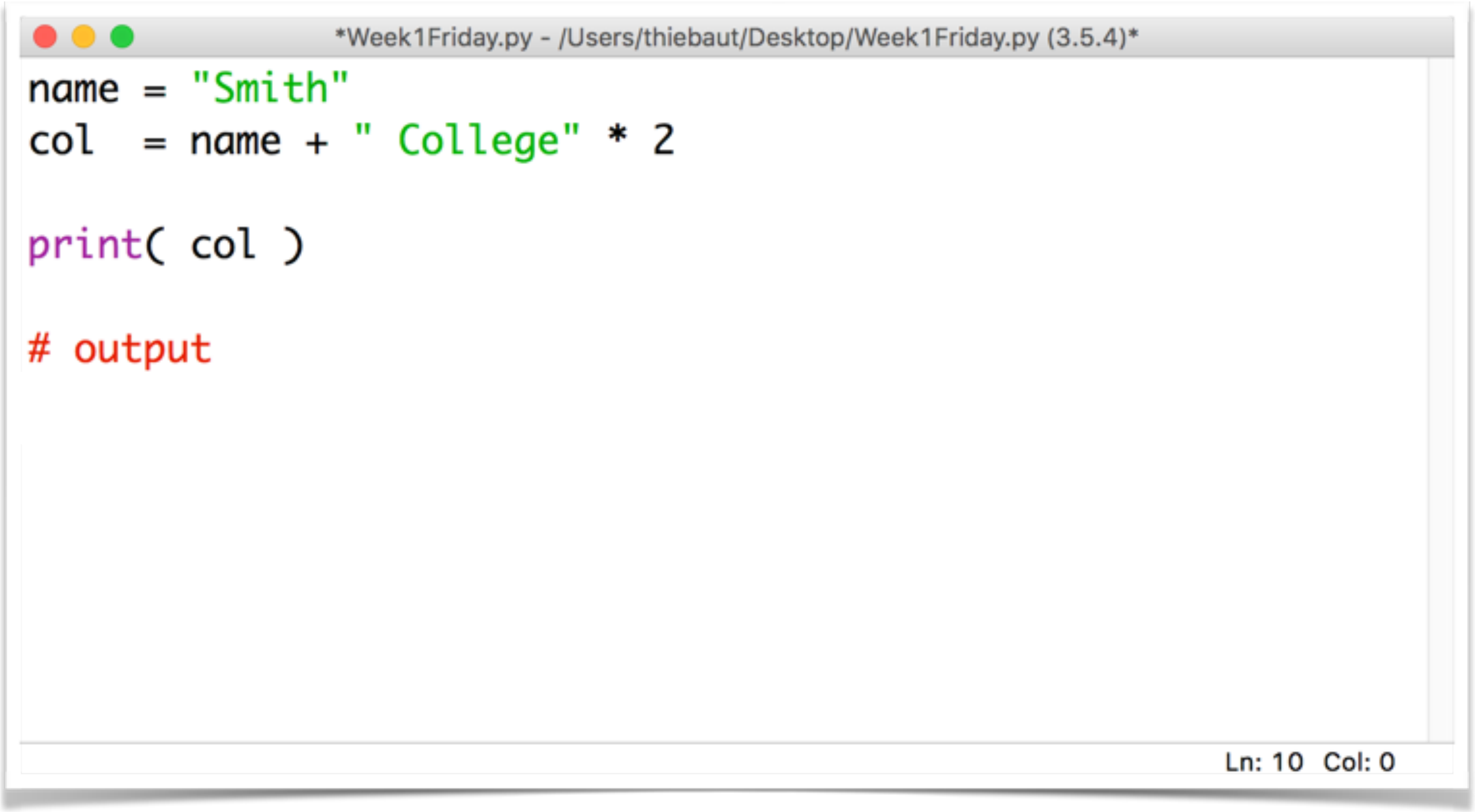
Exercises
Guess what Python will do



```
age  = 3
name = "Smith"
rate = 0.06

age = age * rate       # age will contain 0.18
name = "his + hers"    # name will contain "his + hers"
rate = name * rate     # TypeError: can't multiply
                       # sequence by 'float'
```

Exercises

Guess what Python will do



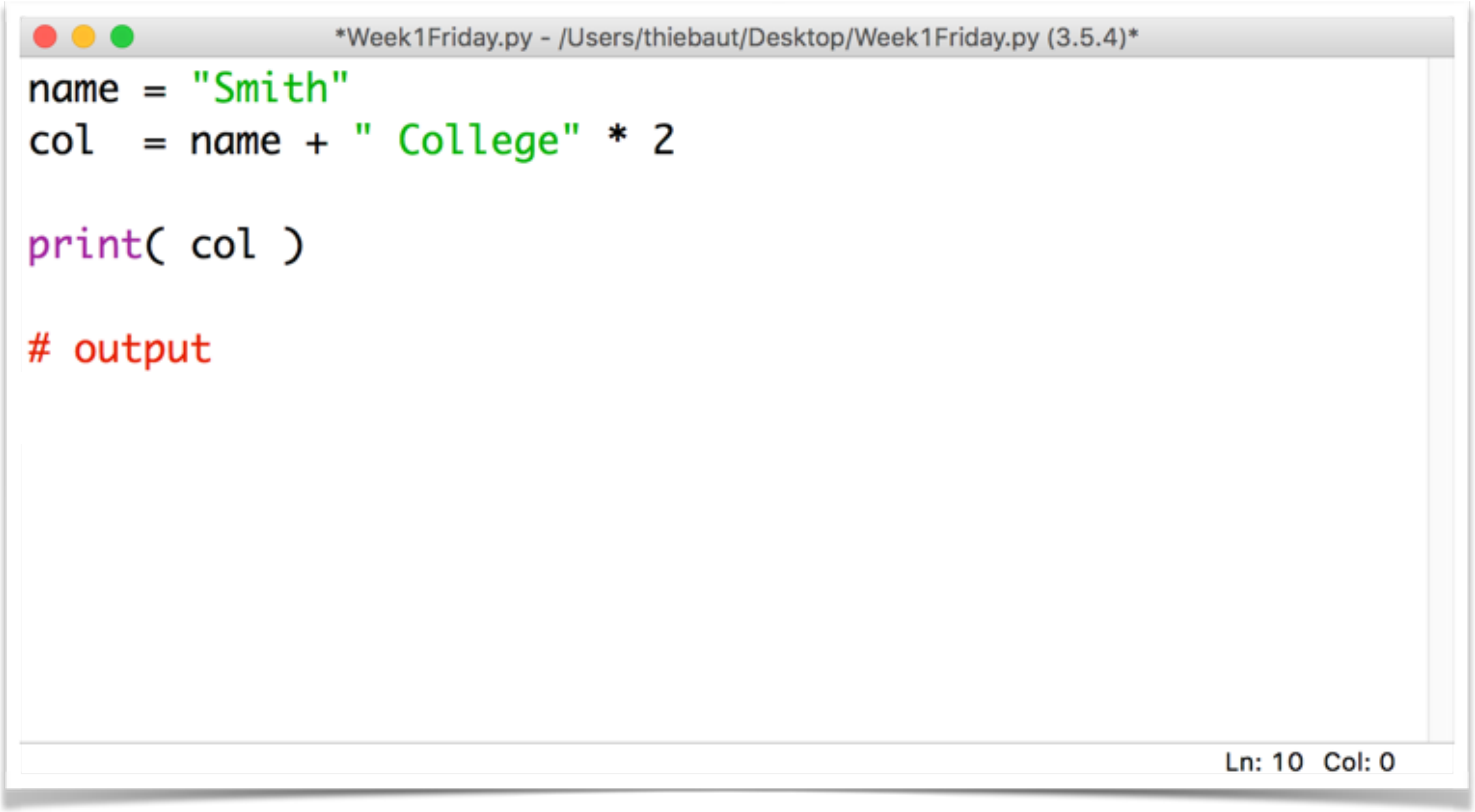
Week1Friday.py - /Users/thiebaut/Desktop/Week1Friday.py (3.5.4)

```
name = "Smith"
col  = name + " College" * 2

print( col )

# output
```

Ln: 10   Col: 0

Exercises

Guess what Python will do



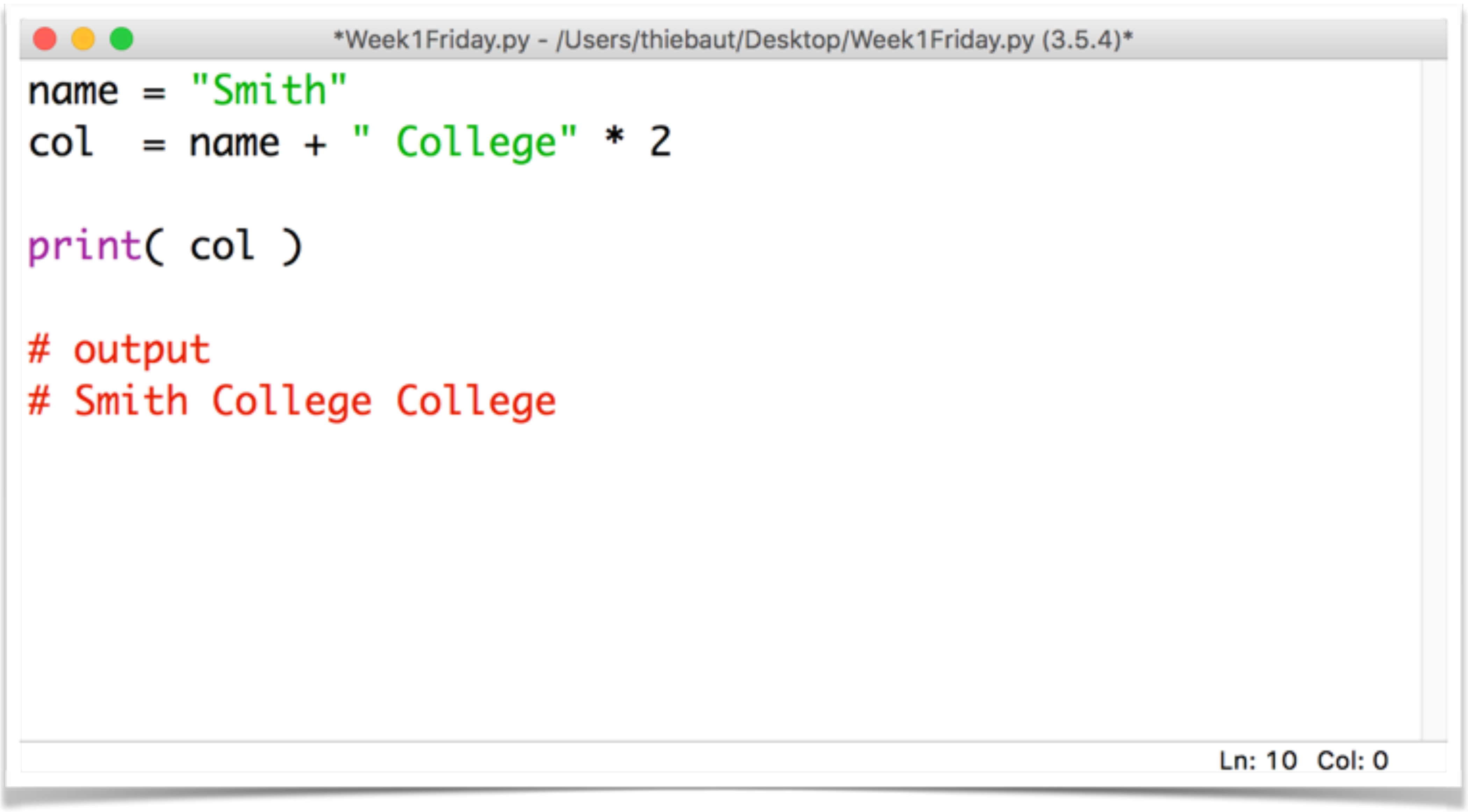
Week1Friday.py - /Users/thiebaut/Desktop/Week1Friday.py (3.5.4)

```
name = "Smith"
col  = name + " College" * 2

print( col )

# output
```

Ln: 10 Col: 0

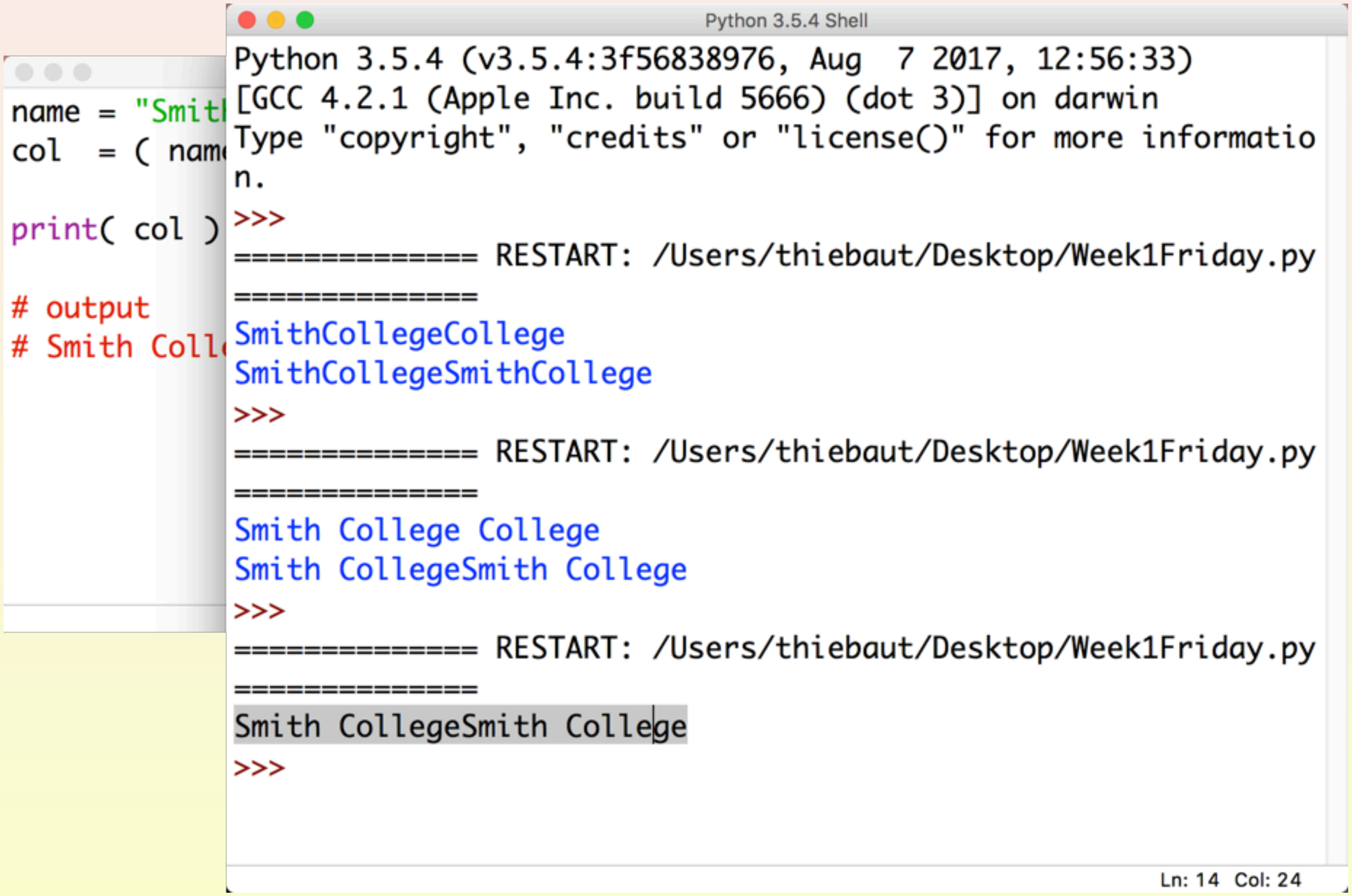
Exercises

Guess what Python will do


Week1Friday.py - /Users/thiebaut/Desktop/Week1Friday.py (3.5.4)

```
name = "Smith"
col  = name + " College" * 2

print( col )

# output
# Smith College College
```

Using the Shell…

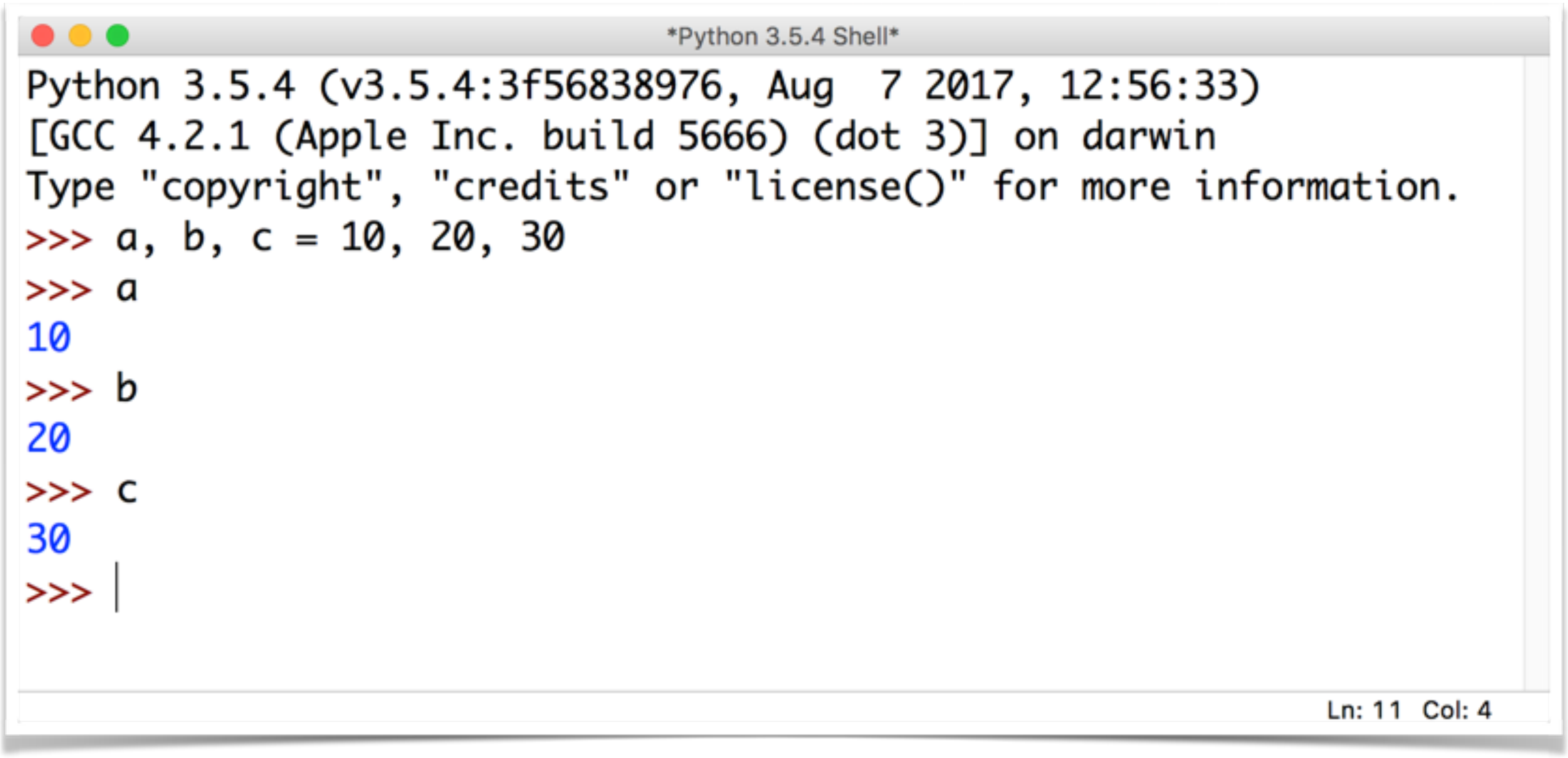


```
name = "Smit
col  = ( nam

print( col )

# output
# Smith Coll
```

```
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 informatio
n.
>>>
============== RESTART: /Users/thiebaut/Desktop/Week1Friday.py
==============
SmithCollegeCollege
SmithCollegeSmithCollege
>>>
============== RESTART: /Users/thiebaut/Desktop/Week1Friday.py
==============
Smith College College
Smith CollegeSmith College
>>>
============== RESTART: /Users/thiebaut/Desktop/Week1Friday.py
==============
Smith CollegeSmith College
>>>
```

Simultaneous Assignments

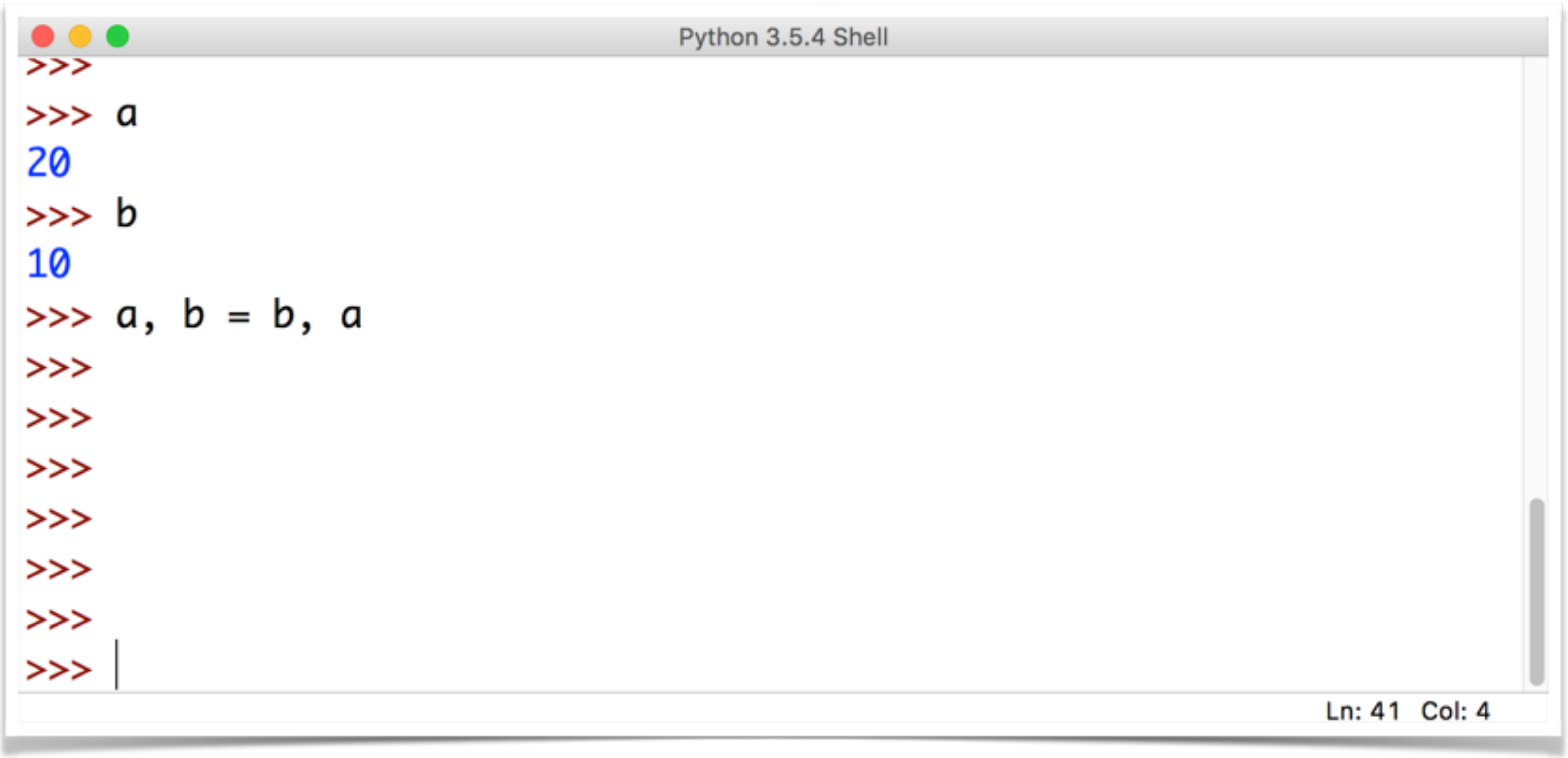
Simultaneous Assignments


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.
>>> a, b, c = 10, 20, 30
>>> a
10
>>> b
20
>>> c
30
>>> 
```

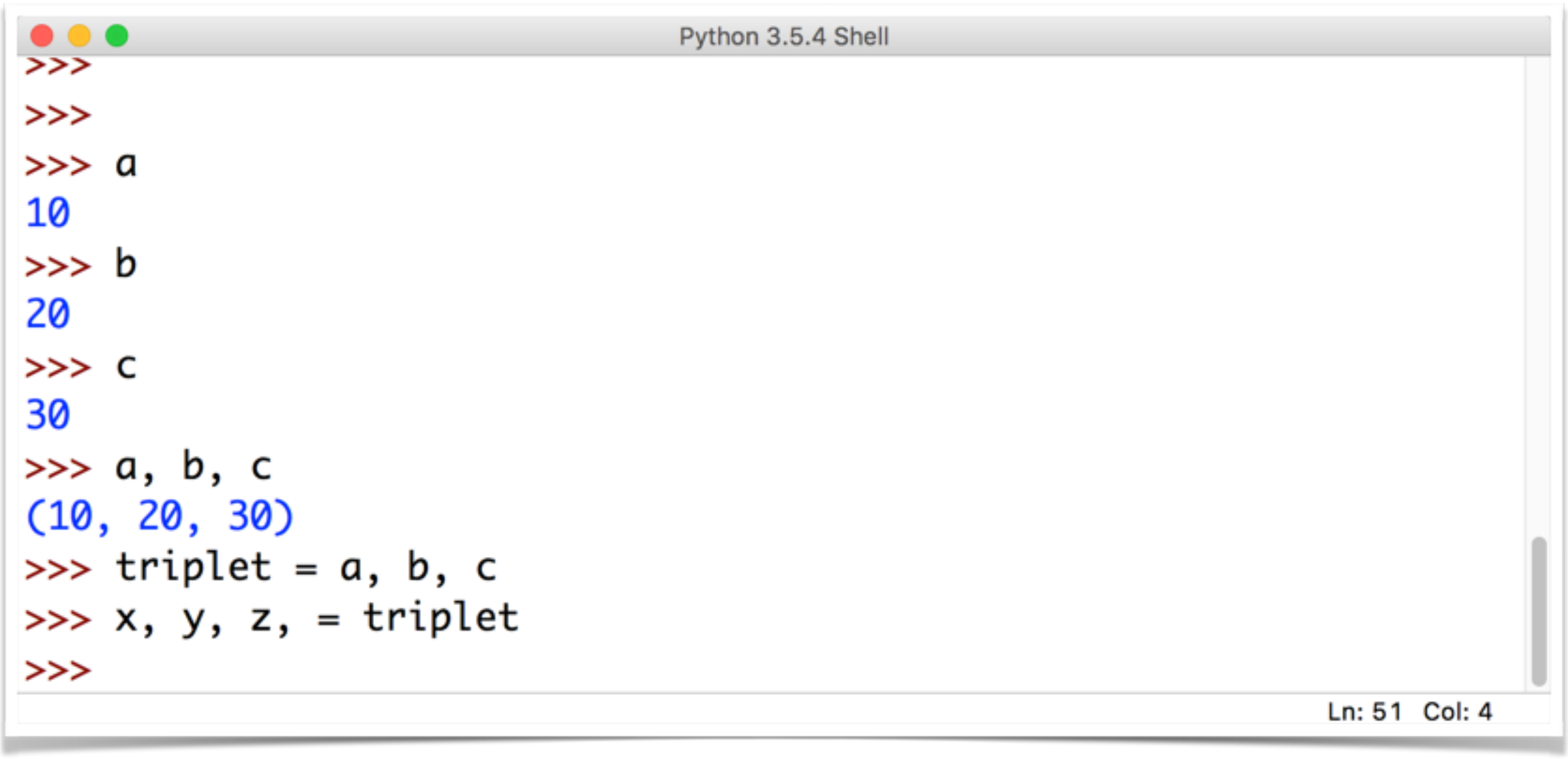
Ln: 11 Col: 4

Swapping Variables



```
>>>
>>> a
20
>>> b
10
>>> a, b = b, a
>>>
>>>
>>>
>>>
>>>
>>>
>>> |
```

Lists and Variables



```
>>>
>>>
>>> a
10
>>> b
20
>>> c
30
>>> a, b, c
(10, 20, 30)
>>> triplet = a, b, c
>>> x, y, z, = triplet
>>>
```

```
a, b, c = 10, 20, 30
                        # a = 10, b = 20, c = 30

triplet = a, b, c       # triplet = (10, 20, 30)

x, y, z = triplet       # x = 10
                        # y = 20
                        # z = 30
```

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

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

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

```
for can in [
```

```
]:
    open( can )
    drink( can )
    throwAway( can )
```

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

Sequence

```
for can in [
```

```
]:
     open( can )
     drink( can )
     throwAway( can )
```

Many actions repeated, each group for each can

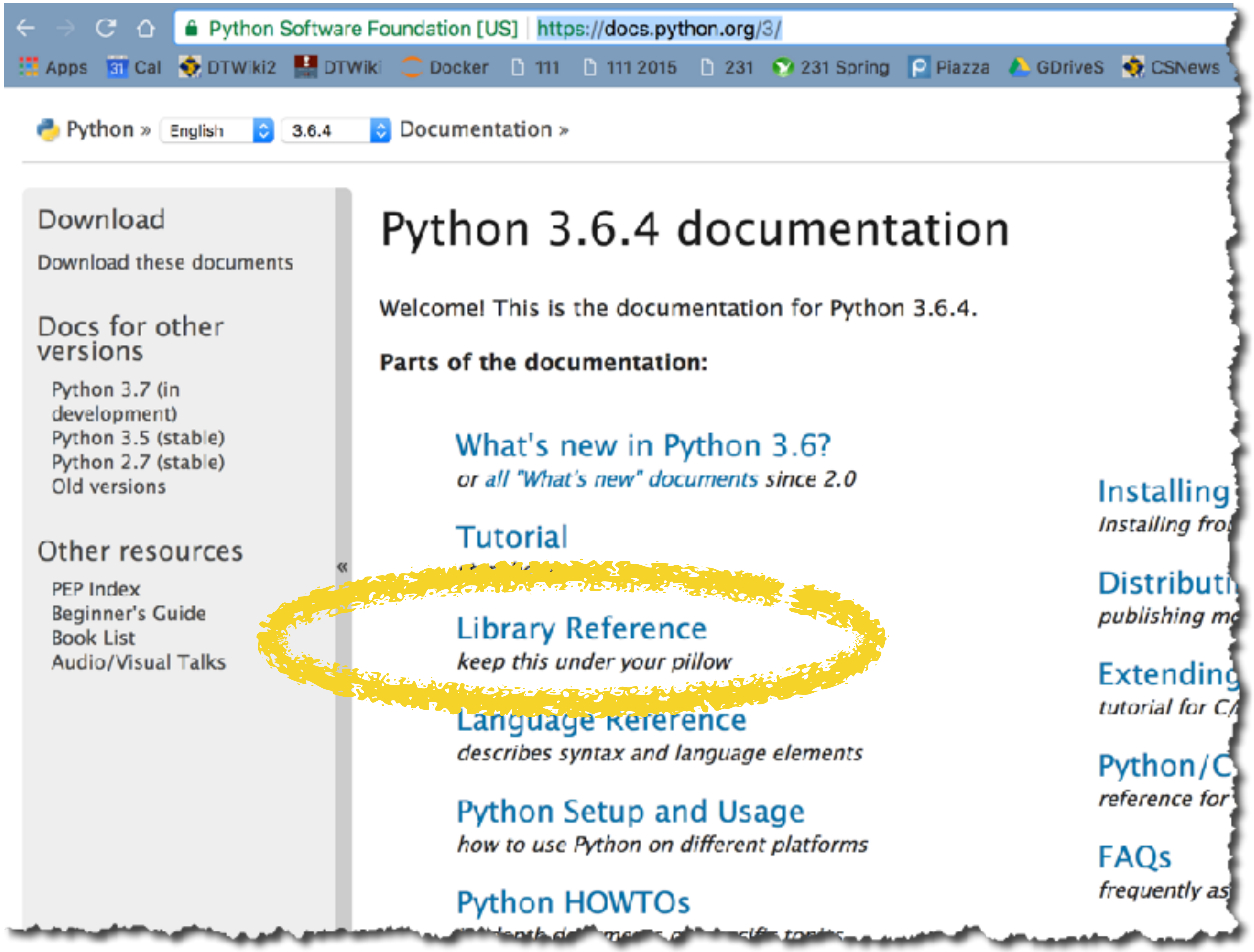
```
for <var> in <sequence>:
     <body>
```

```
for name in [ "Alex", "Max", "Rui" ]:
     open( can )
     drink( can )
     throwAway( can )
```

```
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

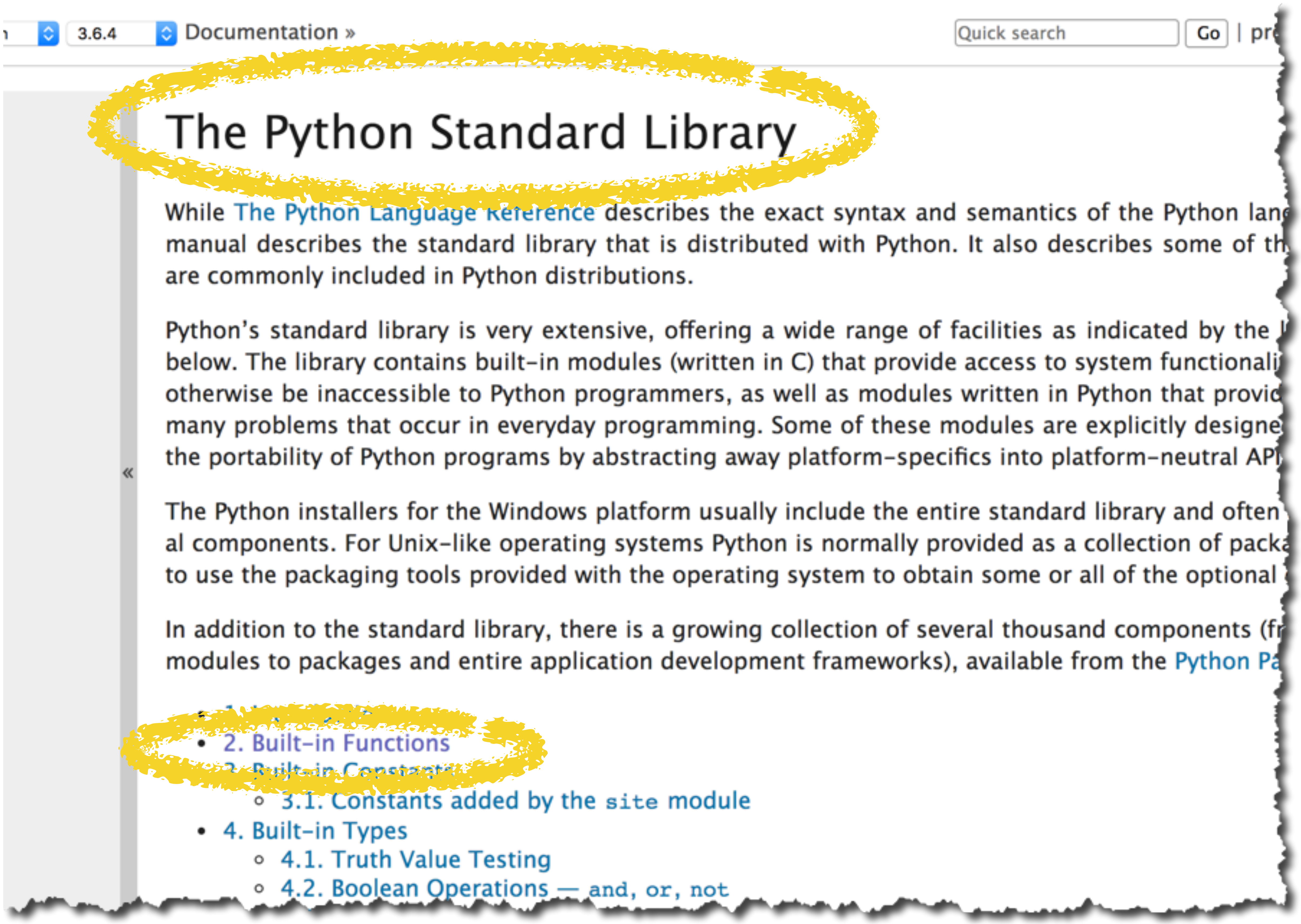
Installing
Installing fro

Distributi
publishing m

Extending
tutorial for C/

Python/C
reference for

FAQs
frequently as



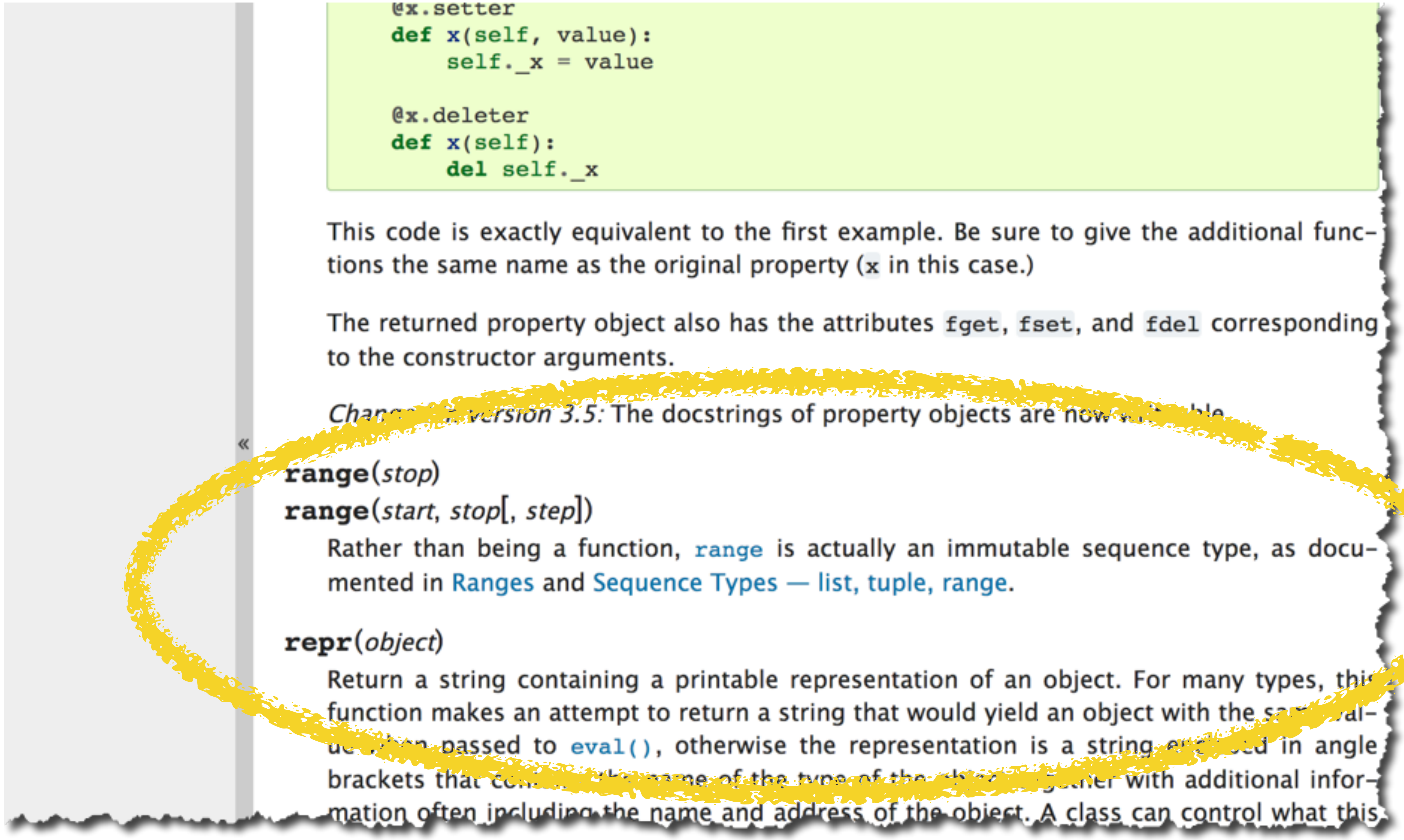
The Python Standard Library

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
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 AP

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 packa
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



```
    @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

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