

# CMPT 120

## Intro to CS & Programming I

### WEEK 8 (Mar. 3-7)

— *Jérémie O. Lumbroso* —

#### Lecture 20:

Strings are immutable? And what are lists??

<http://www.sfu.ca/~jlumbros/Courses/CMPT120/>

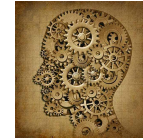
Strings are immutable? Really? But what have we been doing?

# IMMUTABILITY OF STRINGS

# Immutability?

- “**Immutability**” means something cannot be modified/changed
- Strings are objects that cannot be changed

# But wait a minute?

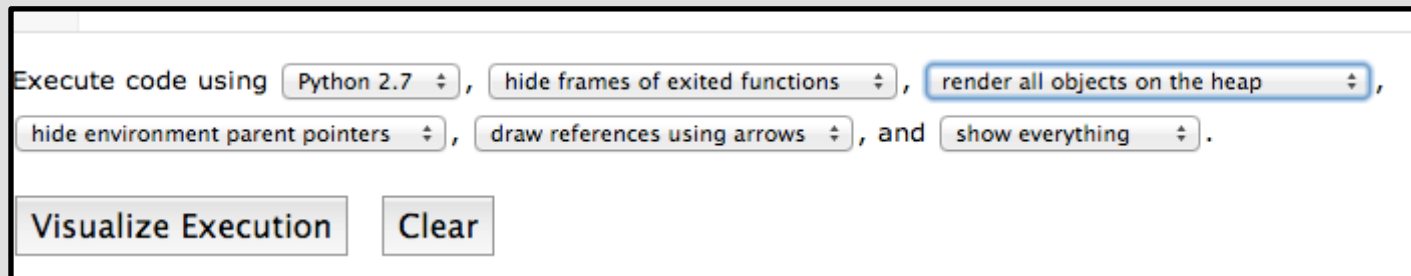
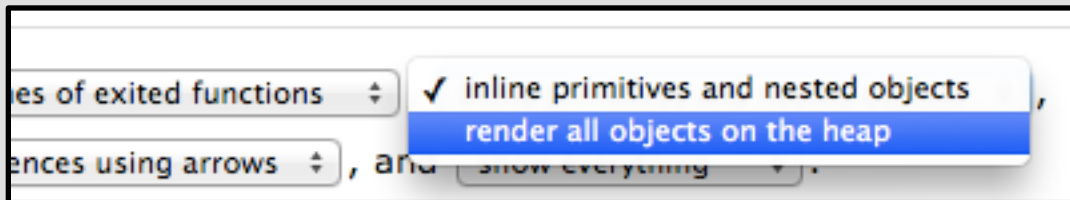


```
>>> a = "hello David"
>>> print a
hello David
>>> a = a[0:6] + "Hien"
>>> print a
hello Hien
```

- A** The string contained in `a` in the first line was modified
- B** The string contained in `a` in the first line was not modified
- C** The variable `a` in the first line was modified
- D** I am, like, confused / not sure what the question is?

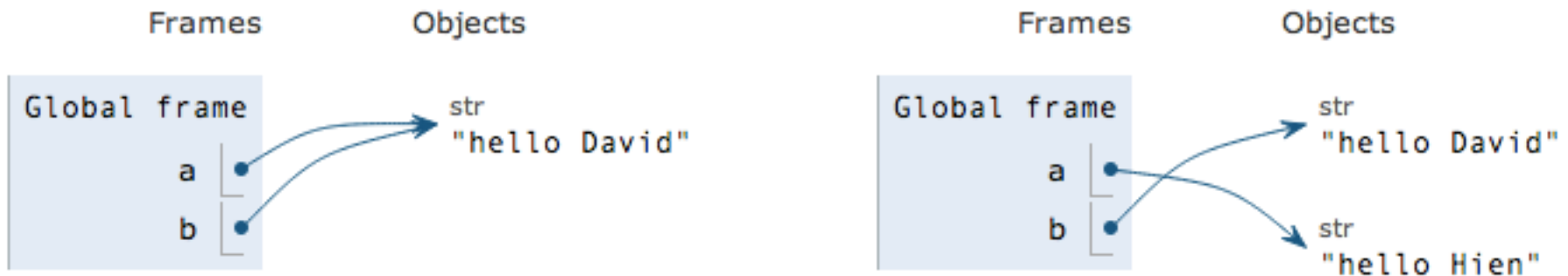
# Setting in Python Tutor

- To visualize some of the examples in this lecture, it is useful to change a setting
- Change “inline primitives and nested objects” to “render all objects on the heap”



# Strings are not modified, just copied

- In the previous example, the string is **copied**
- We can see this in Python Tutor by adding a second variable `b` to contain the original string



- When `a` is modified, a copy of "hello " is produced and then "Hien" is added to it
- `b` still has an arrow (“points”) to the original string

# Changing a Character?



Previously seen example, where we want to change the character of a string: we want to change "Hello Mike" to "Hallo Mike"

```
a = "hello Mike"
a[1] = "a"
print a
```

 1

```
a = "hello Mike"
a = a[0] + "a" + a[2:]
print a
```

 2

```
a = "hello Mike"
a = a.replace("e", "a")
print a
```

 3

```
a = "hello Mike"
a = a.replace("e", "a", 1)
print a
```

 4

A

1, 2 and 3 will work

B

2, 3 and 4 will work

C

2 and 3 will work

D

No clue 'bout 4, but 1 is error

E

1 and 2 work

# Let's Try Again

Can we write the first example like this:

```
>>> a = "hello David"
>>> print a
hello David
>>> a = a[0:6] + "Hien"
>>> print a
```

```
>>> a = "hello David"
>>> a[6:] = "Hien"
>>> print a
```

???

- A** New example (on the right) works and prints Hello Hien
- B** The string is not modified and it prints Hello David
- C** Python does not like this and gives an error
- D** Do you really know anybody named Hien?



# No Modification

- In both examples (replacing a character or replacing a substring), we are trying to use slices to modify the string
- This does not work, because strings are immutable
- That's why when we were creating functions (`replace_ch` or `replace_substr`) we were **creating a new string `new_s` instead of trying to modify the parameter**

# Distinction: about Iterators

- The fact that strings cannot be changed has nothing to do with the fact that values cannot be changed

```
s = "bonjour"
for ch in s:
    ch = ch.upper()
    print ch
print s

r = range(5)
for k in r:
    k = k + 1
    print k
print r
```

- The iteration variable `ch` or `k` receives a copy of the item it is looking at
- **Any change to the iteration variable is not “transferred” to the original value**

Preliminary experimentation...

# LISTS

# Required Reading

Chapter 10, “Lists” of Think Python  
for the next few lectures

<http://www.greenteapress.com/thinkpython/>



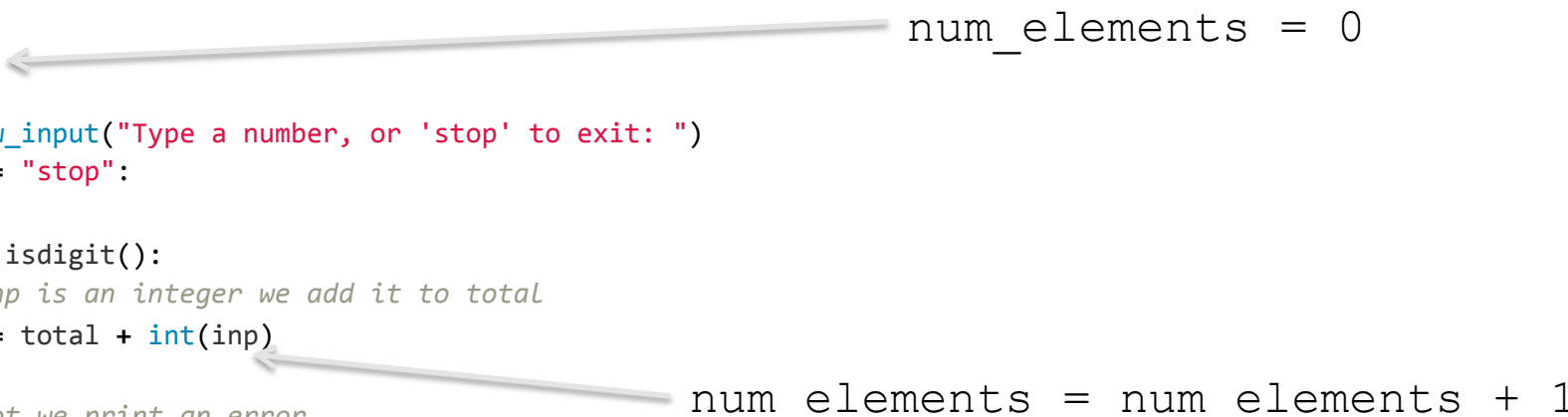
# And now some experimentation...

# Average of Numbers

```
total = 0
num_elements = 0
while True:
    inp = raw_input("Type a number, or 'stop' to exit: ")
    if inp == "stop":
        break
    elif inp.isdigit():
        # if inp is an integer we add it to total
        total = total + int(inp)
    else:
        # if not we print an error
        print "Invalid input was ignored."

# Once we're out, print out total
print "Your numbers added up to", total

print "Average:", total/float(num_elements)
```



And with lists?

# Pacing and Understanding

How well did you understand today?



- A** Too easy, this lecture is way below my abilities
- B** Everything went at a good pace, and I am fine
- C** Too fast, but I will catch up on my own
- D** Too fast, and I need you to slow down
- E** I really do not think I can handle this