CS 150: Review

Cynthia Taylor
Oberlin College
May 7th 2014

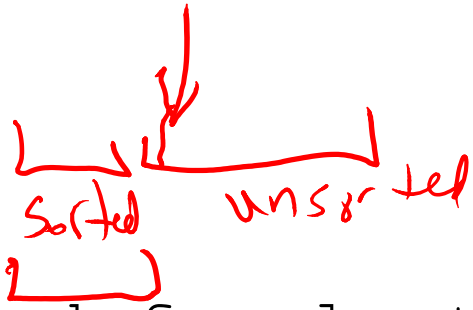
Open Discussion on Community and Academic Environments in the Math and CS Departments

- 12:15pm on Thursday May 8th
- Free pizza if you sign up in advance at http://bit.ly/womacs.
- A conversation about how to make the Math and CS departments more accessible to all.
- Hosted by Women in Math and Computer Science

How can I get extra practice for the final?

- Project Euler – linked to from blackboard
- Make sure you can do all of the shorter lab problems on paper
- Make sure you can answer all of the clicker questions and do all of the coding problems from class
- Make up problems similar to practice final, labs, class problems
  - Post on blackboard!!

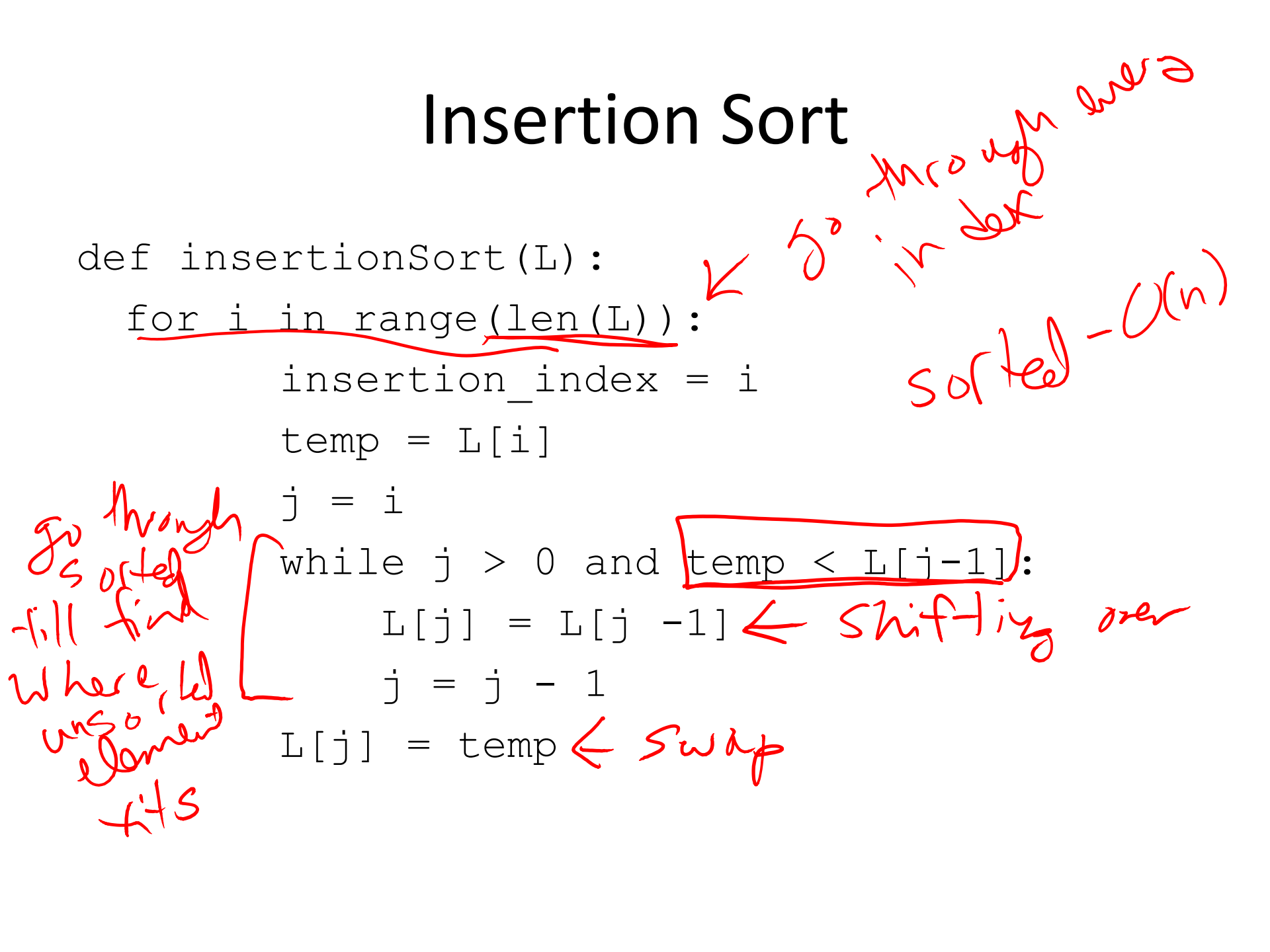
Selection Sort




```
def selectionSort(L):
    for i in range(len(L)):
        min_index = i
        for j in range(i, len(L)):
            if L[j] < L[min_index]:
                min_index = j
        temp = L[i]
        L[i] = L[min_index]
        L[min_index] = temp
```

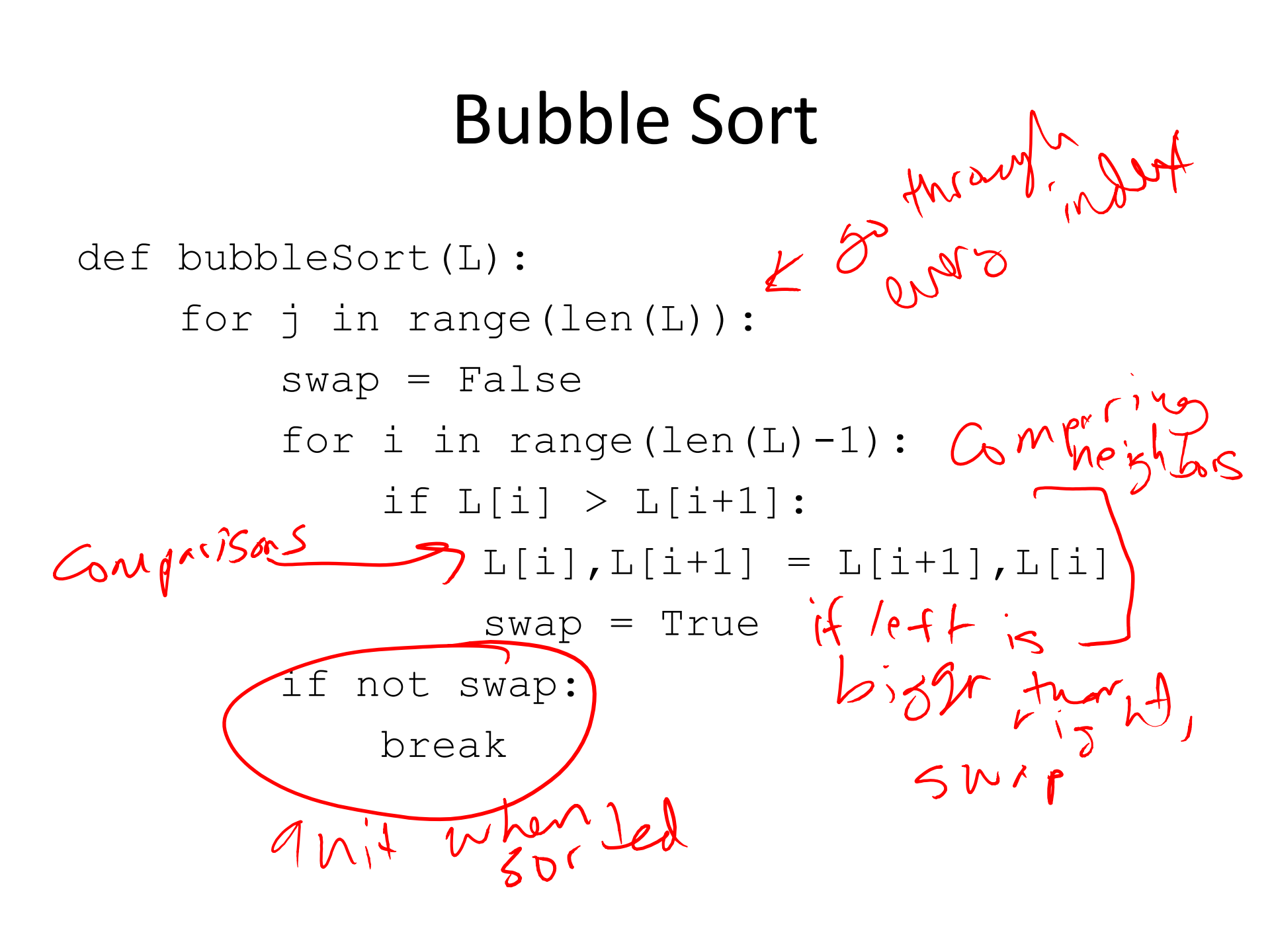
Insertion Sort

```
def insertionSort(L):
    for i in range(len(L)):
        insertion_index = i
        temp = L[i]
        j = i
        while j > 0 and temp < L[j-1]:
            L[j] = L[j -1]
            j = j - 1
        L[j] = temp
```



Bubble Sort

```
def bubbleSort(L):
    for j in range(len(L)):
        swap = False
        for i in range(len(L)-1):
            if L[i] > L[i+1]:
                L[i],L[i+1] = L[i+1],L[i]
                swap = True
        if not swap:
            break
```



Merge Sort

```
def mergeSort(A):
    if len(A) > 1:
        m = len(A)//2
        print(A)
        B = mergeSort(A[:m])
        C = mergeSort(A[m:])
        A = Merge(B,C)
    return A
```

Which will perform better on a mostly sorted list?

A. Selection Sort

B. Insertion Sort

C. They will perform the same

D. I don’t know

Write pseudocode to do a binary search over a sorted list.

3

L, e

```
def binSearch(L,e):
    m = len(L)//2
    if e<L[m]:
        L = L[:m]
    return binSearch(L)
    if e>L[m]
        ret bS(L[m:])
    if e==L[m]:
        ret m
```

```
if len(L)==1
and e!=L[m]:
    ret -1
```

Assume you have a very large number, n, and a list, P, that consists of the primes from 2 to the first prime larger than the square root of n. Use a process pool with 4 processes to determine if n is prime.

Write code where 10 processes each add 1 to a RawValue r 10 times, ending with r.value equal to 100. Make sure you use the lock to ensure you end up with the correct result.

Create a subclass of S, R, which says "Wheee!" instead of "Hi", and eats pizza instead of kale. Its sleep should remain the same.

```
class S:
  def talk(self):
    return "Hi!"

  def nom(self):
    return "Delicious kale!"

  def sleep(self):
    return "zzzzzz"
```

class R(S):
def talk(self):
return "Whoeeee!"

def nom(self):

return "delicious pizza"

__init__(self)
self.talk = "Hi"

```
class S:
    def __init__(self, x):
        self.x = x

class T(S):
    def __init__(self, x, b):
        self.b = b
        S.__init__(self, x)
```

Next Time

- Class Wrap Up
- READING ON BLACKBOARD for Friday

- Final Exam:
  - Wednesday the 14th, 7pm, King 306