On your way in...

Pick-up
1. Lecture notes (side table)
2. Graded HW3 in numbered folders (front desk)

Hand-in (side table, 2 piles under & over 50)
1. HW4
Welcome to CS 134!

Introduction to Computer Science

Iris Howley

-Iterators-

Spring 2019
James Mickens, Class of 60s’ Scholars Lecture

**BLOCKCHAINS ARE A BAD IDEA**

*(More specifically, Blockchains are a very bad idea)*

Thursday, 3/7 at 8pm in Wege

Reception to follow
Iris’ Office Hours Tonight

MONDAY, FEBRUARY 4, 6-8PM
TPL205
Midterm Exam is Tuesday, March 5 in Wege Auditorium (this room)

• Choose one of 2 time slots:
  1. 5:45-7:45pm
     • If your accommodations letter provides extended time, attend this session
  2. 8-10pm

• Midterm Review Session:
  • Sunday, 3/3 7-9pm in Wege

• Iris’ Office Hours:
  • Monday, 3/4 6-8pm in TPL 205
Johnson’s MATH150 or Williams’ BIOL102?

• If you’re currently taking:
  • Stewart Johnson’s MATH 150
  • Heather Williams’ BIOL 102

• Please come talk to me after class!
• (alternatively, send me an email)
Midterm Exam is Tuesday, March 5

• Closed book exam

• Covers topics up to and including last week
  • Review your homeworks! Exercises in the book! POGIL activities!
  • Midterm Study Session Notes are on the course website

• This week’s lab will be *experiential*
NO CLASS WEDNESDAY MARCH 6
Loops

• for letter in ‘Python’:
  ▪ if letter == ‘h’:
    ◦ break
  ▪ print (‘Current Letter: ‘, letter)

What will this print?

Current Letter: p
Current Letter: y
Current Letter: t
Loops

for letter in 'Python':
    if letter == 'h':
        continue
    print ('Current Letter: ', letter)

What will this print?

Current Letter: p
Current Letter: y
Current Letter: t
Current Letter: o
Current Letter: n
TODAY’S LESSON

Iterators

(objects that return one element at a time)
Iterators

• We’ve been using iterators all along!
• The for statement calls iter() on ‘hello’ string
• iter() returns an iterator which has a __next__() method, which goes in and accesses each element in ‘hello’
  ▪ Returning one at a time!
• When it runs out of elements, it raises a StopIteration exception, so the for..loop terminates
Iterators

- \( s = 'abc' \)
- \( \text{it} = \text{iter}(s) \)
- \( \text{it} \quad <\text{str_iterator object at 0x107a58668}> \)
- \( \text{next(it)} 'a' \)
- \( \text{next(it)} 'b' \)
- \( \text{next(it)} 'c' \)
- \( \text{next(it)} \)
- Traceback (most recent call last): File "<stdin>" , line 1, in <module> next(it) StopIteration
s = 'hi!

it = iter(s)

try:
    print(next(it))
    print(next(it))
    print(next(it))
    print(next(it))

except StopIteration:
    print("ERROR. Ran outta juice!")
For..loops

• for item in mylist:
  ▪ print(item)

• try:
  ▪ it=iter(mylist)
  ▪ while True:
    ▪ item = next(it)
    ▪ print(item)
• except StopIteration:
  ▪ pass

This is really:
Python Tutorial on Iterators

• Getting to the end of our textbook!

• https://docs.python.org/3/tutorial/classes.html#iterators
ITERATORS ARE A SUBSET OF GENERATORS
Generators

def countTo(n):
    i = 1
    while i <= n:
        yield i
        i += 1

g = countTo(3)
print(next(g))  # 3
print(next(g))  # 1
print(next(g))  # 2
print(next(g))  # ERROR StopIteration
Generators

def countTo(n):
    i = 1
    while i <= n:
        yield i
        i += 1

g = countTo(3)
print(next(g))  # 1
print(next(g))  # 2
print(next(g))  # 1

def countRet(n):
    i = 1
    while i <= n:
        return i
        i += 1

print(countRet(5))  # 1
print(countRet(5))  # 1
print(countRet(5))  # 1
Can have multiple return statements

```python
def countRet(n):
    i = 1
    while i <= n:
        return i
        i+= 1
```

```python
def multRet(num):
    if num <= 0:
        return num
    else:
        return "POS"
```

Once we reach ‘return’ we never get past it!
i is never incremented!

"POS" is only returned if "return num" is never reached, i.e., when num is greater than 0.