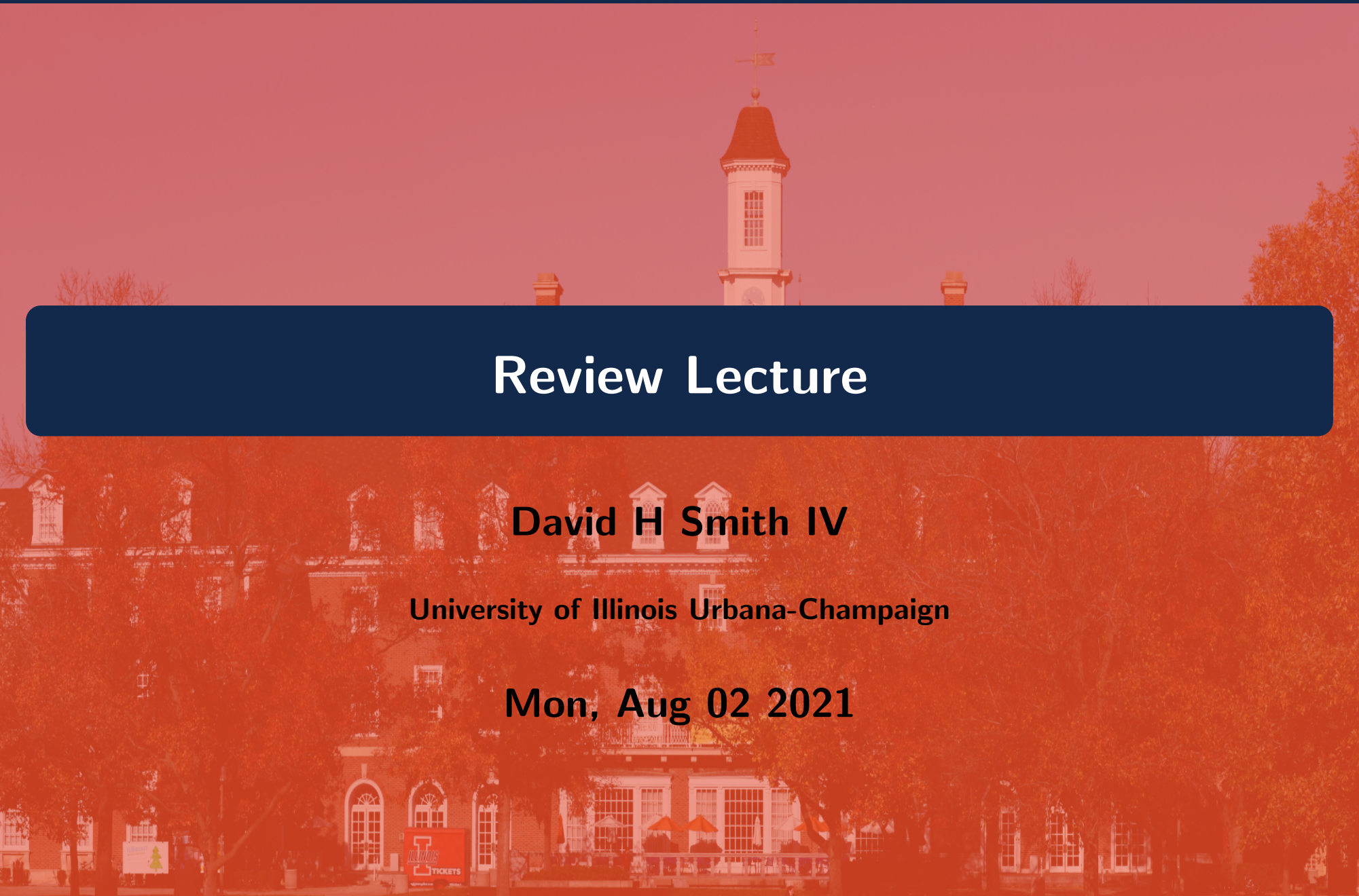

# Review Lecture

**David H Smith IV**

**University of Illinois Urbana-Champaign**

**Mon, Aug 02 2021**

# Reminders

# Reminders

- Practice final is up and attempting it is the final homework (250 pts).
- The final "Lecture" is going to be an office hours where you can swing by and attempt the practice final.
- The final is this Saturday at 1pm so please register for it on the scheduler.
- I'll be available all week for questions, even outside of the usual office hours, so feel free to ping me on Discord.
- Please review the gradebook to make sure everything looks correct.
- **Please be sure to fill out the ICES forms if you haven't already.**

# Patterns

# Patterns of Interest

- Counting pattern
- Sum/Total Pattern
- Computing a Sum/Total Over Specific Elements
- Finding single item in collection
- Finding best in collection
- Filtering a collection
- Histogram

# Counting Pattern

```
def count(collection):
  counter = 0
  for item in collection:
    if <item meets condition>:
      counter += 1
  return counter
```

# Computing a Sum/Total

```
1 def sum(collection):
2   total = 0
3   for item in collection:
4     total += item
5   return total
```

# Computing a Sum/Total Over Specific Elements

```
1 def sum(collection):
2   total = 0
3   for item in collection:
4     if <condition>:
5       total += item
6   return total
```

# Dictionaries: Computing a Histogram

Creating a count map of items in a collection is a common dictionary pattern:

```
1 def get_item_counts(some_list):
2   counts = {}
3   for item in some_list:
4     if item not in counts:
5       counts[item] = 1
6     else:
7       counts[item] += 1
```

# Finding (single thing) in a Collection

```
def find_thing(collection):
  for thing in collection:
    if <thing meets condition>:
      return thing
```

```
def find_thing(collection):
  found = None
  for thing in collection:
    if <thing meets condition>:
      found = thing
      break
  return found
```

# Finding best in collection

```
def find_best(collection):
  currentbest = ??
  for thing in collection:
    if <thing is better than current best>:
      currentbest = thing
  return currentbest
```

- If we're searching over a list and we want to return the largest or smaller number: `currentbest = stufflist[0]`

# Finding best in collection

```
def find_best(collection):
  currentbest = ??
  for thing in collection:
    if <thing is better than current best>:
      currentbest = thing
  return currentbest
```

- If we're searching over a list and we want to return the largest or smaller number: `currentbest = stufflist[0]`
- If we're searching over a list of strings and we want to return the longest string: `currentbest = stufflist[0]` or `currentbest = ""`

# Finding best in collection

```
def find_best(collection):
  currentbest = ??
  for thing in collection:
    if <thing is better than current best>:
      currentbest = thing
  return currentbest
```

- If we're searching over a list and we want to return the largest or smaller number: `currentbest = stufflist[0]`
- If we're searching over a list of strings and we want to return the longest string: `currentbest = stufflist[0]` or `currentbest = ""`
- If you know the list contains only non-negative integers: `currentbest = -1`

# Filtering a collection

```
def filter(collection):
  new_list = []

  for thing in collection:
    if <thing meets criteria>:
      newlist.append(thing)

  return new_list
```

# General Review: Common Points of Error

# Poll Questions: Functions

How many functions will the function call `count_letters\("Hello, 105!"\)` iterate?

```
1 def count_letters(s):
2   count = 0
3   for c in s:
4     count += 1
5     return count
```

A. 0
B. 1
C. 10
D. 11

# Poll Question: Find and Slicing

What does the following code produce?

```
1 def foo(x):
2   f = x.find(',')
3   s = x.find(',', s + 1)
4   return x[f:(s + 1)]
5 foo("This, small sentence, is a test.")
```

- A Error on Line 2
- B Error on line 3
- C Error on line 4
- D ', small sentence,'

# Poll Question: Find and Slicing

Which of these lines of code need to be used to fix it such that it produces the output `, small sentence,`?

```
1 def foo(x):
2   f = x.find(',')
3   s = x.find(',', s + 1)
4   return x[f:(s + 1)]
5 foo("This, small sentence, is a test.")
```

A. 3: `s = x.find(',', s + 1)`

B. 3: `s = x.find(',', f)`

C. 3: `s = x.find(',', f + 1)`

D. 3: `s = x.find(',', f - 1)`

# Poll Question: Functions

What does the following function produce if called with `foo(["apples", "bananas"])`?

```
def foo(str_list):
  count = 0
  for s in str_list
    for c in s:
      if c == 'a':
        count += 1
return count
```

A. None
B. Error
C. 2
D. 4

# Filtering a collection

What is the value of new_list after running the following code?

```
x = ["This", "Is", "A", "Test"]
new_list = []
for i in x:
  if i < 2:
    new_list.append(i)
```

A. Error
B. `["Is", "A"]`
C. `["A"]`
D. `[]`

# Files

# Files

Which of the following reads all the contents of a file into a list of strings?

A. readlines
B. readall
C. read
D. readline

# Poll Question: Read Characters

Given a variable named `file_object` that contains a file object which of the following will read the next 15 character into a variable named `title`.

A. `title = file_object.read(15)`

B. `title = file_object.read(14)`

C. `title = file_object.reads(15)`

D. `title = read(file_object, 15)`

# Reading from Files

Method 1:

```
file_object = open('filename')
lines = file_object.readlines()
for line in lines:
  print(line)
file_object.close()
```

Method 2:

```
with open('filename') as inf:
  lines = inf.readlines()
  for line in lines:
    print(line)
  #automatic file close
```

# Writing to Files

Method 1:

```
file_object = open('filename', 'w')
file_object.write('thing to write')
file_objet.close()  #automatic at program end
file_object.flush() #optional
```

Method 2:

```
with open('filename', 'w') as outf:
  outf.write('thing to write')
  #automatic file close
```

# Patterns and Files

## Usual Sum/Total:

```
1 def foo(some_list):
2   total = 0
3   for item in
      some_list
4     total += item
5   return total
```

## Sum/Total Pattern w/ File:

```
1 def foo(filename):
2   file_object = open(filename)
3   lines = file_object.readlines()
4   total = 0
5   for line in lines:
6     total += int(line)
7   return total
```

# More Review

# Poll Question: Booleans

What is the correct boolean expression for if we want to determine if the list `x` only if it contains either elements `y` `z`.

A. `y or z in x`

B. `x in y or x in z`

C. `y in x or z in x`

D. `y in x and z in x`

# Filtering a collection

Which function call will produce the output `'hj'`.

```
1 def foo(x, y, z):
2     return x[y: z]
```

A. `foo('asdfghjkl', -4, -2)`
B. `foo('asdfghjkl', -3, -2)`
C. `foo('asdfghjkl', -3, -1)`
D. `foo('asdfghjkl', -5, -2)`