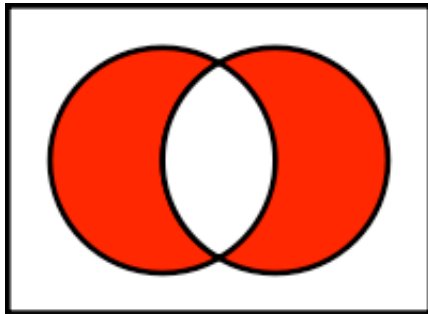


CompSci 101

Introduction to Computer Science



October 9, 2014

Prof. Rodger

Thanks to Prof. Azhar and Yossra Hamid for
giving this lecture!

Announcements

- Reading for next time on calendar page
 - en.wikibooks.org/wiki/Python_Programming/Sets
 - RQ
- APT 4 is due today
 - APT 5 is out today
- Exam 1 was handed out Tuesday, grades are on Sakai, you will need to see Prof. Rodger next week to get your test back
- Today Sets
- Prof. Rodger is at a conference this week
 - <http://gracehopper.org/>

Python Sets

- Set – unordered collection of distinct items
 - Unordered – can look at them one at a time, but cannot count on any order
 - Distinct - one copy of each
- Operations on sets:
 - Modify: add, clear, remove
 - Create a new set: difference(-), intersection(&), union (|), symmetric_difference(^)
 - Boolean: issubset <=, issuperset >=
- Can convert list to set, set to list

Summary (from wikibooks)

- `set1 = set()` # A new empty set
- `set1.add("cat")` # Add a single member
- `set1.update(["dog", "mouse"])` # Add several members
- `set1.remove("cat")` # Remove a member - error if not there
- `print set1`
- `for item in set1:` # Iteration AKA for each element
 `print item`
- `print "Item count:", len(set1)` # Length AKA size AKA item count
- `isempty = len(set1) == 0` # Test for emptiness
- `set1 = set(["cat", "dog"])` # Initialize set from a list
- `set3 = set1 & set2` # Intersection
- `set4 = set1 | set2` # Union
- `set5 = set1 - set3` # Set difference
- `set6 = set1 ^ set2` # Symmetric difference (elements in either
set but not both)
- `issubset = set1 <= set2` # Subset test
- `issuperset = set1 >= set2` # Superset test
- `set7 = set1.copy()` # A shallow copy (copies the set, not the
elements)
- `set8.clear()` # Clear AKA empty AKA erase

Creating and changing a set

```
colorList = ['red', 'blue', 'red', 'red', 'green']  
colorSet = set(colorList)  
smallList = list(colorSet)  
colorSet.clear()  
colorSet.add("yellow")  
colorSet.add("red")  
colorSet.add("blue")  
colorSet.add("yellow")  
colorSet.add("purple")  
colorSet.remove("yellow")
```

- See setsEasy.py

Set Operations

```
UScolors = set(["red", "white", "blue"])
dukeColors = set(["blue", "white"])
print dukeColors.union(UScolors)
print dukeColors | UScolors
print dukeColors.intersection(UScolors)
print dukeColors & UScolors
print dukeColors.difference(UScolors)
print dukeColors - UScolors
print UScolors - dukeColors
print dukeColors ^ UScolors
print UScolors ^ dukeColors
```

- See setsEasy.py

Set Examples

bit.ly/101fall14-1009-01

```
poloClub = set(['Mary', 'Laura', 'Dell'])
```

```
rugbyClub = set(['Fred', 'Sue', 'Mary'])
```

Question 1:

```
print [w for w in poloClub.intersection(rugbyClub)]
```

Question 2:

```
print [w for w in poloClub.union(rugbyClub)]
```

More Set Examples

bit.ly/101fall14-1009-02

```
lista = ['apple', 'pear', 'fig', 'orange', 'strawberry']
```

```
listb = ['pear', 'lemon', 'grapefruit', 'orange']
```

```
listc = [x for x in lista if x in listb]
```

```
listd = list(set(lista)|set(listb))
```

Question 1:

```
print listc
```

Question 2:

```
print listd
```

More Set Examples

```
s = set(lista)    lista = ['apple', 'pear', 'fig', 'orange', 'strawberry']  
t = set(listb)    listb = ['pear', 'lemon', 'grapefruit', 'orange']
```

```
problem1 = (s-t) | (t-s)
```

```
print problem1
```

```
problem2 = (s|t) - (s&t)
```

```
print problem2
```

```
problem3 = (s|t|(s&t))
```

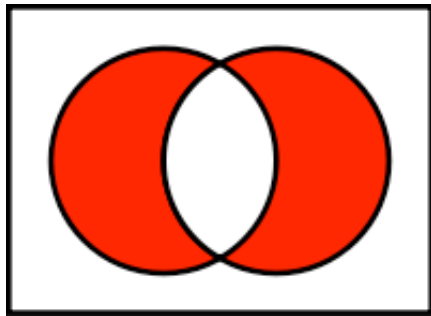
```
print problem3
```

Set Operations from pictures

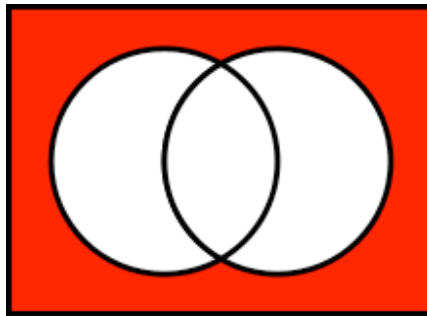
bit.ly/101fall14-1009-03

Question: Which picture is which operation?

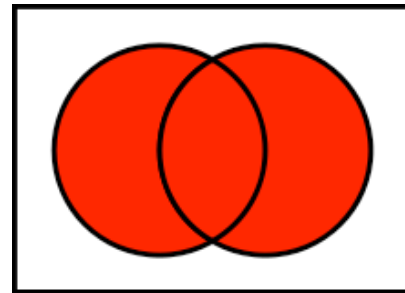
A)



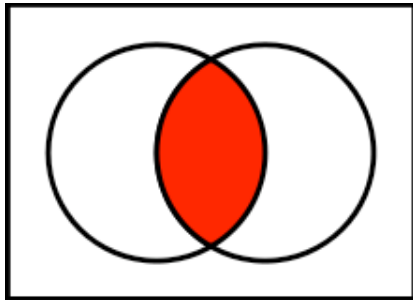
C)



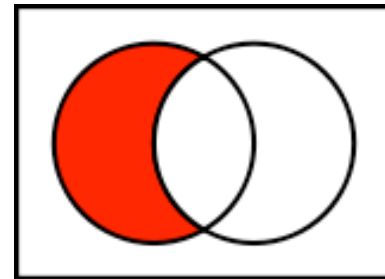
D)



B)



E)



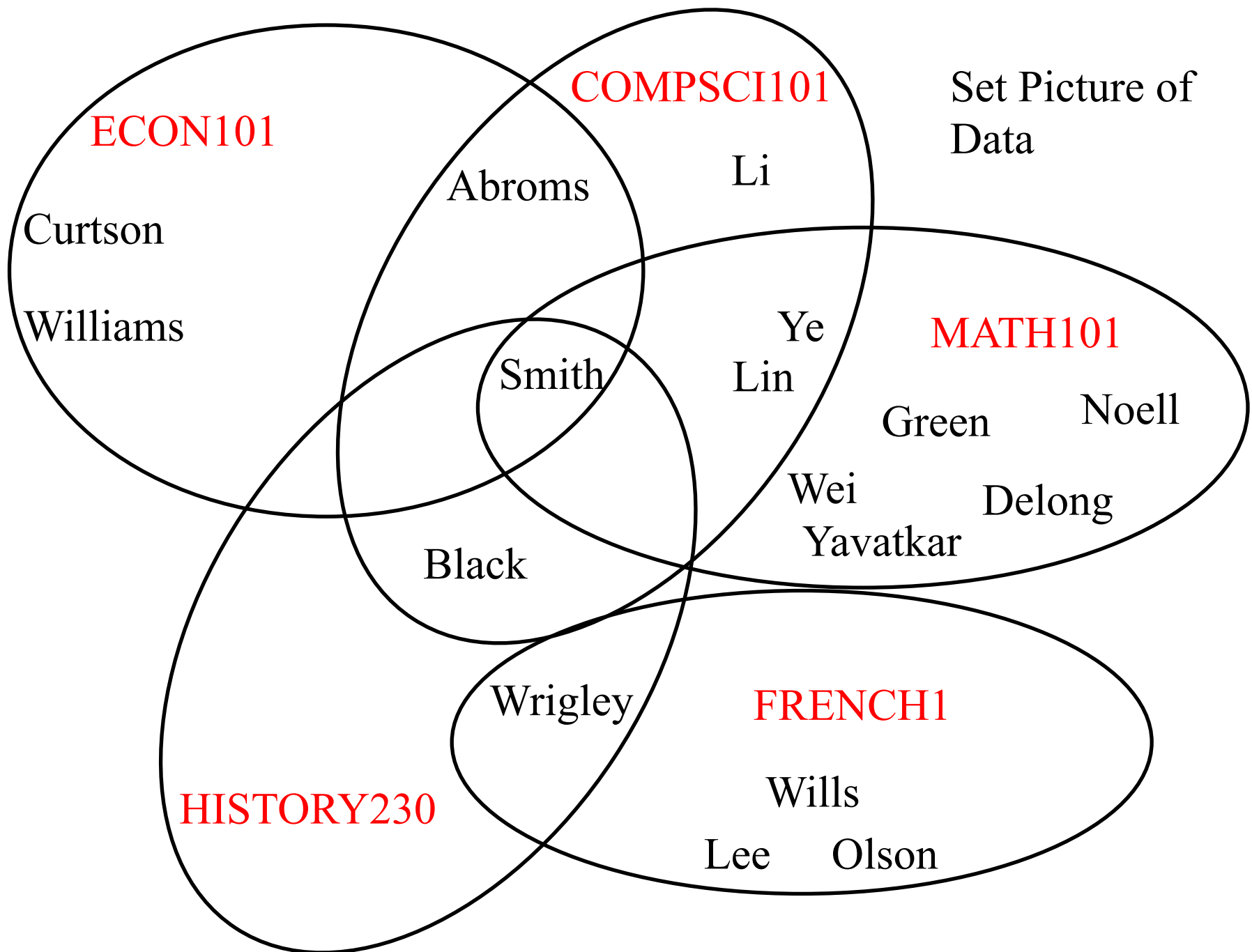
Problems — snarf setExample.py

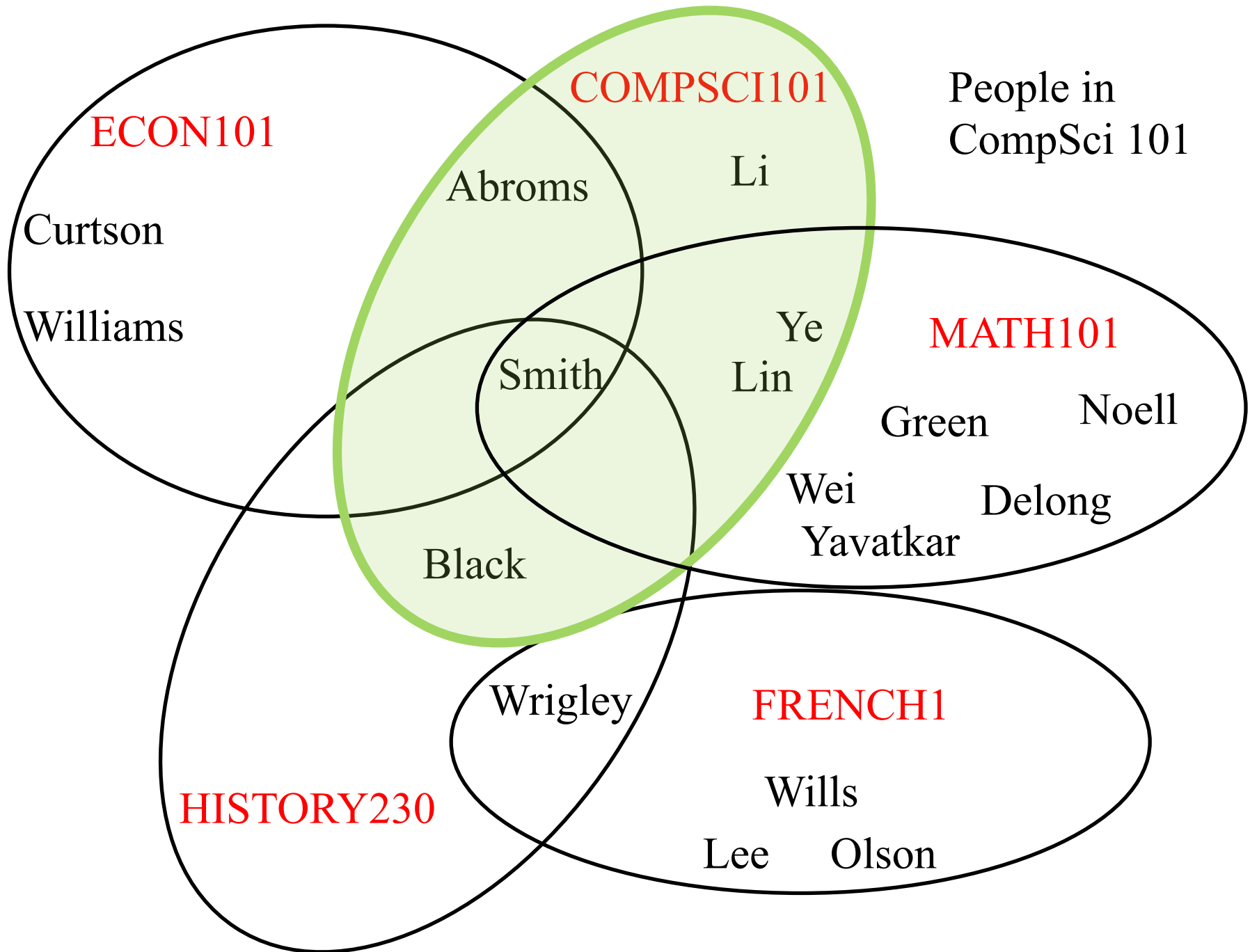
- Given a list of strings that have the **name of a course (one word)**, followed by **last names** of people in the course:
 - Convert list into lists of strings of names for each course
 - Find total number of people taking any course
 - Find number of people taking just one course

*["econ101 Abrams Curtson Williams Smith",
"history230 Black Wrigley Smith", ...]*

Data for example

[compsci101 Smith Ye Li Lin Abroms Black“,
“math101 Green Wei Lin Williams DeLong
Noell, Ye, Smith”,
“econ101 Abroms Curtson Williams Smith”,
“french1 Wills Wrigley Olson Lee”,
"history230 Black Wrigley Smith"]





ECON101

Curtson

Williams

COMPSCI101

Abrams

Li

People Taking
both Math
And CompSci

MATH101

Ye

Lin

Green

Noell

Delong

Wei

Yavatkar

Intersection

Smith

Black

HISTORY230

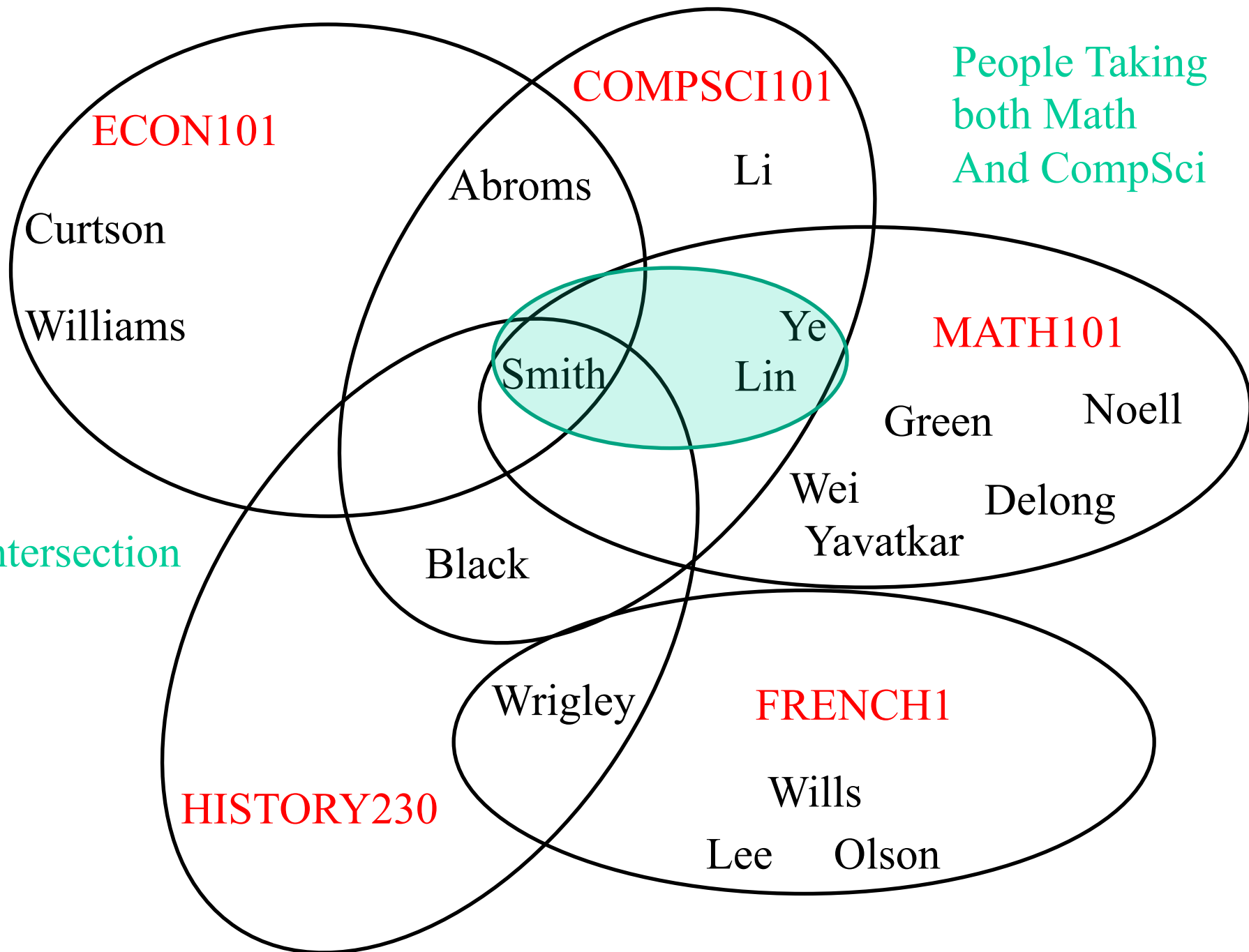
Wrigley

FRENCH1

Wills

Lee

Olson



Part 1 — processList

bit.ly/101fall14-1009-04

- Given a list of strings that have the name of a course (one word), followed by last names of people in the course:
 - Convert list into lists of strings of names for each course

*["econ101 Abrams Curtson Williams Smith",
"history230 Black Wrigley Smith", ...]*

*[['Abrams', 'Curtson', 'Williams', 'Smith'],
['Black', 'Wrigley', 'Smith', ...]]*

Part 1: processList

Input: **list** of strings

*[“compsci101 Smith Ye Li Lin
Abroms Black”,
“math101 Green Wei Lin
Williams DeLong Noell Ye
Smith”,
“econ101 Abroms Curtson
Williams Smith”,
“french1 Wills Wrigley Olson
Lee”,
“history230 Black Wrigley
Smith”]*

Return: **list** of lists (of strings)

*[[“Smith”, “Ye”, “Li”, “Lin”,
“Abroms”, “Black”],
[“Green”, “Wei”, “Lin”,
“Williams”, “DeLong”,
“Noell”, “Ye”, “Smith”],
[“Abroms”, “Curtson”,
“Williams”, “Smith”],
[“Wills”, “Wrigley”, “Olson”,
“Lee”],
[“Black”, “Wrigley”,
“Smith”]]*

Part 2 — peopleTakingCourses

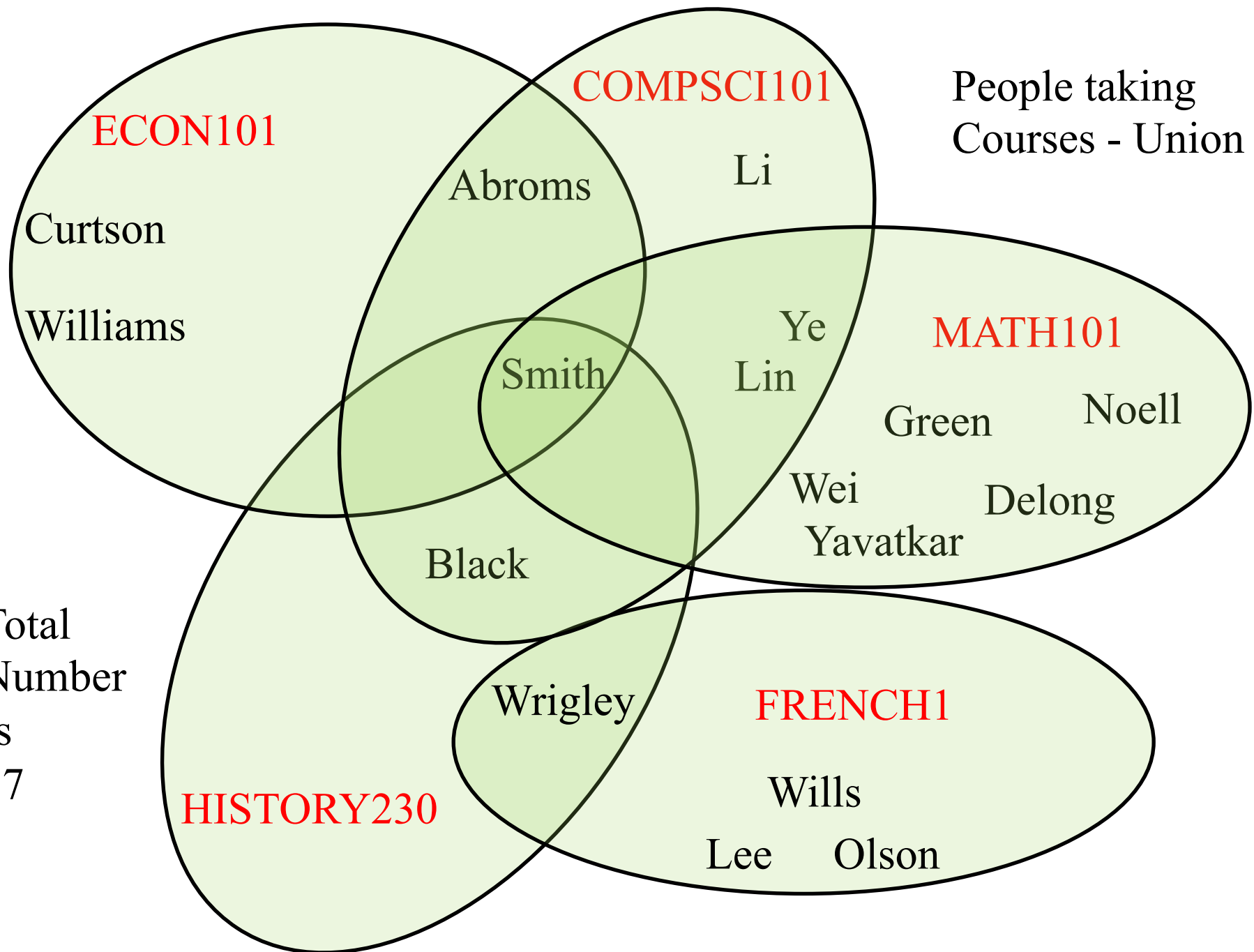
bit.ly/101fall14-1009-05

- Given a list of lists (of strings) that have the last names of people in the course:
 - **Find total number of people taking any course**

*["econ101 Abrams Curtson Williams Smith",
"history230 Black Wrigley Smith", ...]*

*[['Abrams', 'Curtson', 'Williams', 'Smith'],
['Black', 'Wrigley', 'Smith', ...]]*

6...



Part 2 – peopleTakingCourses

Input: **list** of lists (of strings)

*[[“**Smith**”, “**Ye**”, “**Li**”,
“**Lin**”, “**Abroms**”, “**Black**”],
[“**Green**”, “**Wei**”, “**Lin**”,
“**Williams**”, “**DeLong**”,
“**Noell**”, “**Ye**”, “**Smith**”],
[“**Abroms**”, “**Curtson**”,
“**Williams**”, “**Smith**”],
[“**Wills**”, “**Wrigley**”,
“**Olson**”, “**Lee**”],
[“**Black**”, “**Wrigley**”,
“**Smith**”]]*

Return: **list** of strings (of all
unique students)

*[“**Smith**”, “**Ye**”, “**Li**”, “**Lin**”,
“**Abroms**”, “**Black**”, “**Green**”,
“**Wei**”, “**DeLong**”, “**Noell**”,
“**Curtson**”, “**Wills**”, “**Olson**”,
“**Lee**”, “**Wrigley**”,
“**Williams**”]*

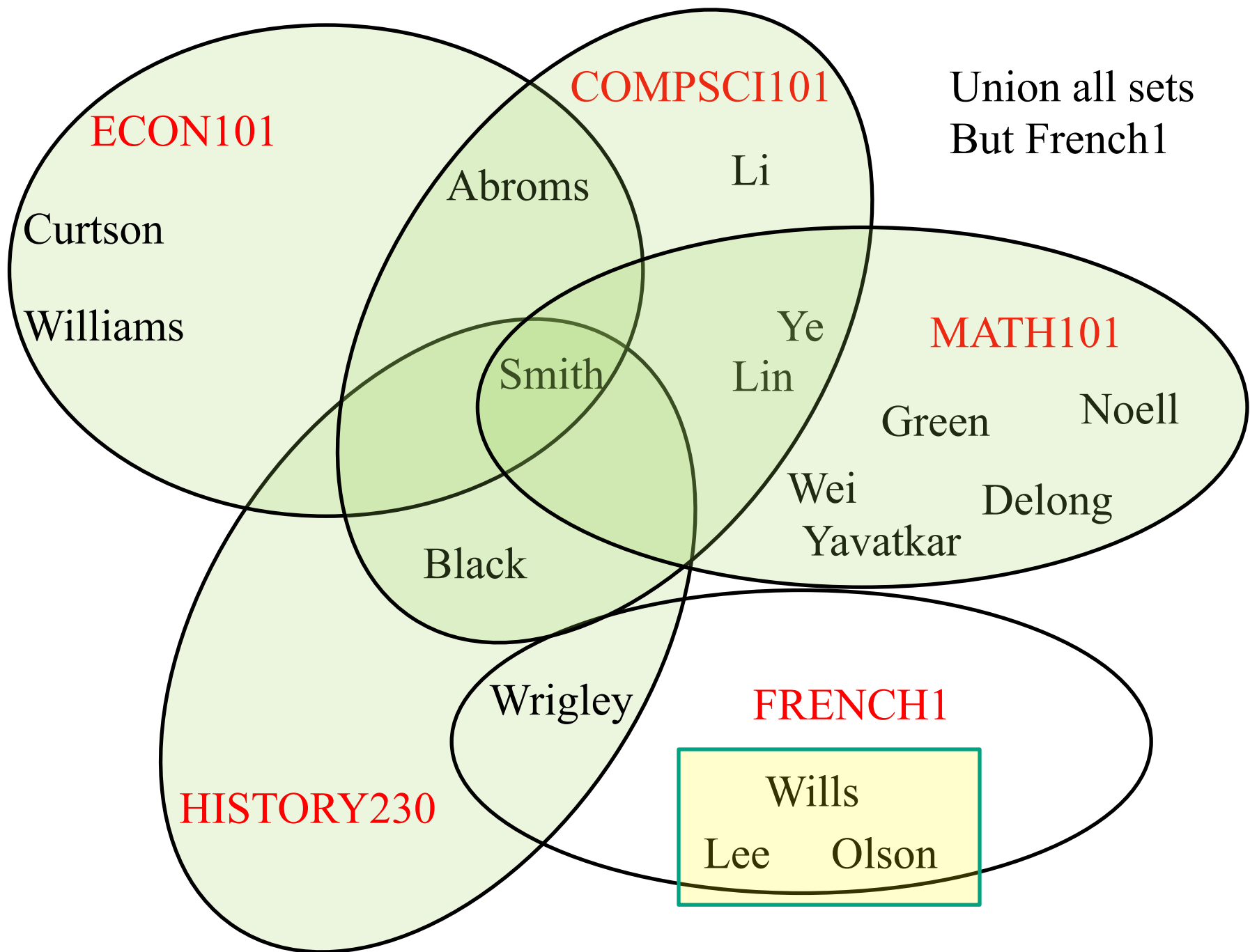
*Length of list = total people
taking courses*

Part 3 — unionAllSetsButMe

bit.ly/101fall14-1009-06

- Given a list of lists (of strings) that have the last names of people in the course:
 - Find number of people taking just one course
 - BUT FIRST, lets write this helper method
 - **Return the union of all sets except current set**

*["econ101 Abrams Curtson Williams Smith",
"history230 Black Wrigley Smith", ...]*



Part 3 – unionAllSetsButMe

Input: **list** of lists (of strings)

index of current set

*[[“Smith”, “Ye”, “Li”,
“Lin”, “Abroms”, “Black”],
[“Green”, “Wei”, “Lin”,
“Williams”, “DeLong”,
“Noell”, “Ye”, “Smith”],
[“Abroms”, “Curtson”,
“Williams”, “Smith”],
[“Wills”, “Wrigley”, “Olson”,
“Lee”],
[“Black”, “Wrigley”,
“Smith”]]*

Return: **set** – union of all lists

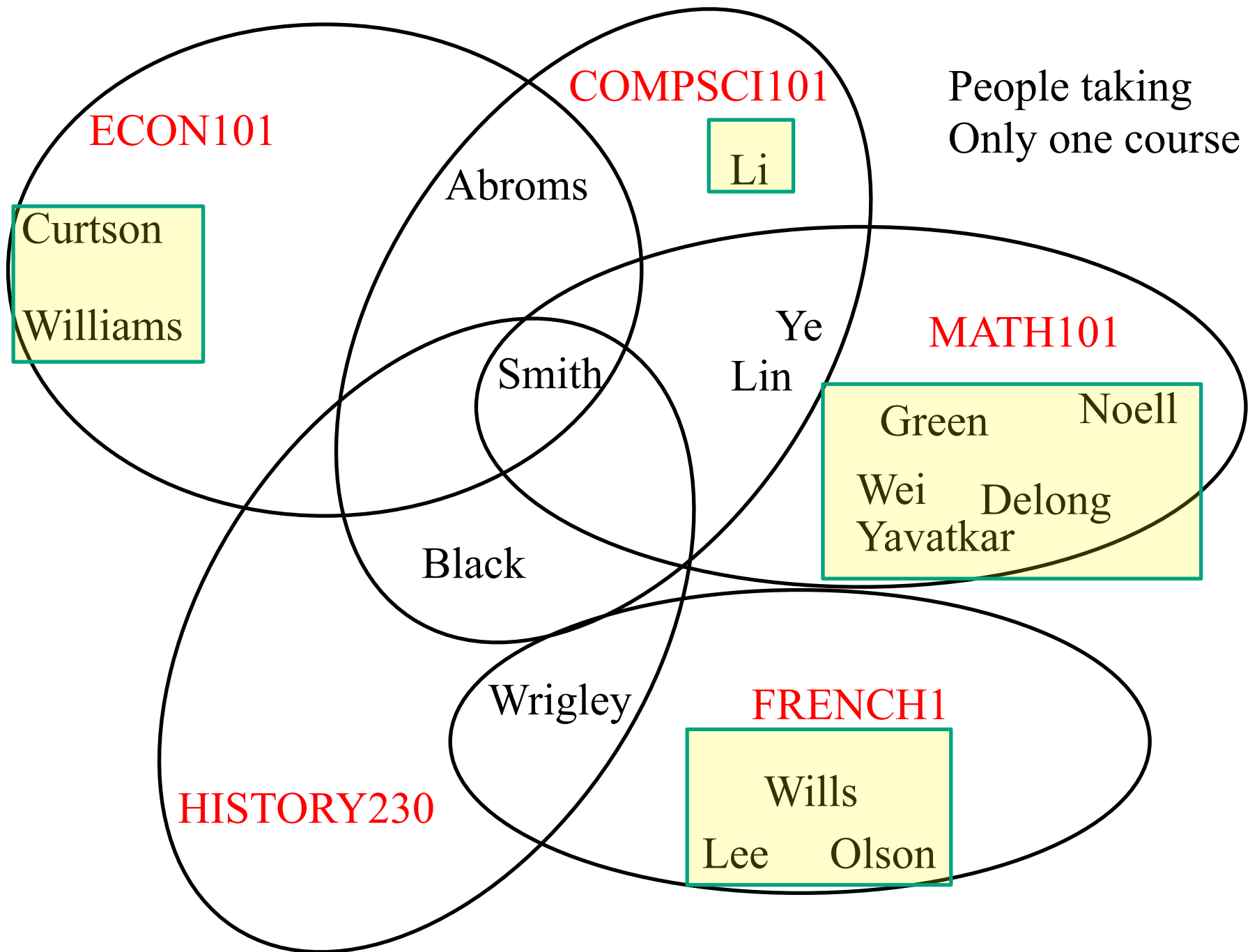
*[“Smith”, “Ye”, “Li”, “Lin”,
“Abroms”, “Black”, “Green”,
“Wei”, “DeLong”, “Noell”,
“Curtson”, “Williams”,
“Wrigley”]*

Part 4 — peopleTakingOnlyOneCourse

bit.ly/101fall14-1009-07

- Given a list of lists (of strings) that have the last names of people in the course:
 - **Find number of people taking just one course**

*["econ101 Abrams Curtson Williams Smith",
"history230 Black Wrigley Smith", ...]*



Part 4 – peopleTakingOnlyOneCourse

Input: **list** of lists (of strings)

```
[ [“Smith”, “Ye”, “Li”,  
  “Lin”, “Abroms”, “Black”],  
  [“Green”, “Wei”, “Lin”,  
    “Williams”, “DeLong”,  
    “Noell”, “Ye”, “Smith”],  
  [“Abroms”, “Curtson”,  
    “Williams”, “Smith”],  
  [“Wills”, “Wrigley”,  
    “Olson”, “Lee”],  
  [“Black”, “Wrigley”,  
    “Smith”] ]
```

Return: **list** of strings
(students only taking one
course)

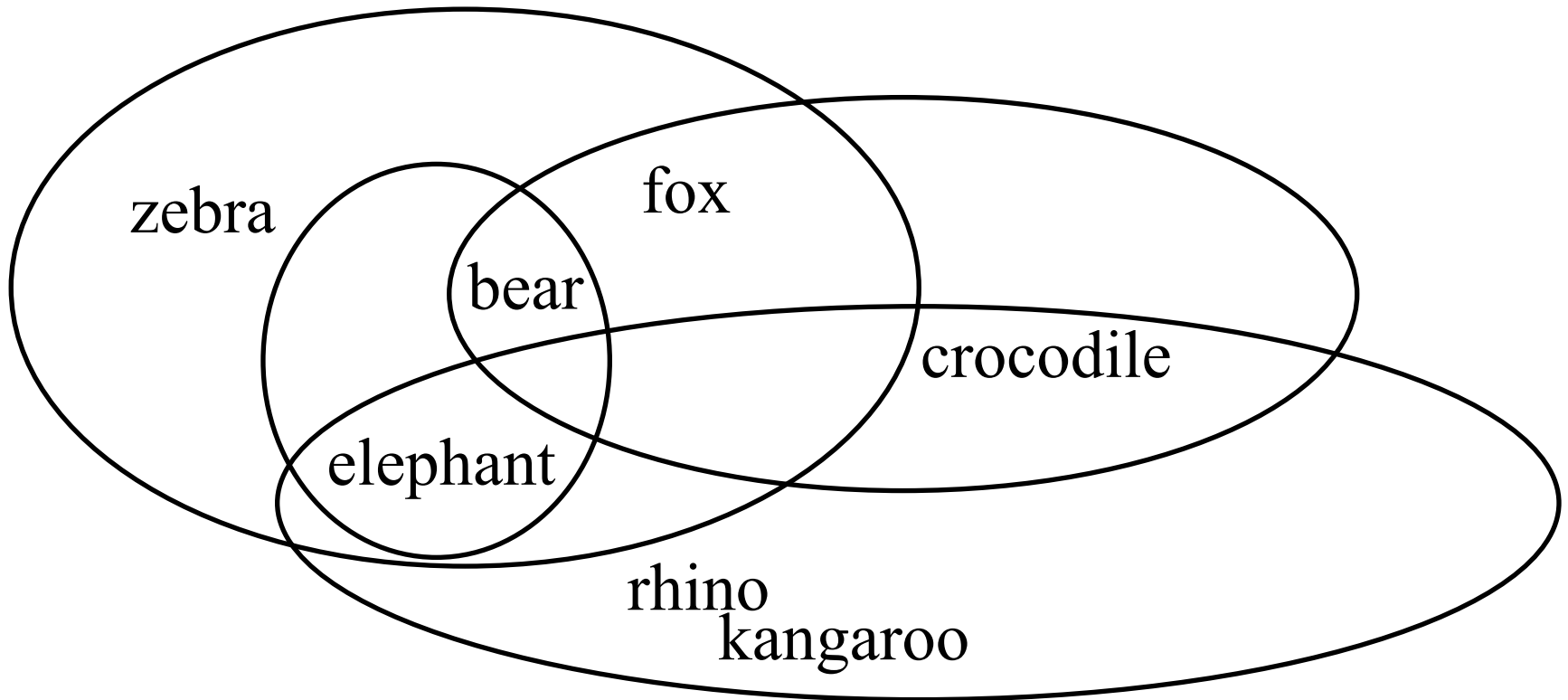
```
[ “Li”, “Green”, “DeLong”,  
  “Noell”, “Curtson”, “Wills”,  
  “Olson”, “Lee”]
```

APT - UniqueZoo

- How do you solve this problem?
- How is it similar to the problem we just solved

Example Data for UniqueZoo

["zebra bear fox elephant", "bear crocodile fox",
"rhino elephant crocodile kangaroo", "elephant
bear"]



UniqueZoo – two zoos have unique animals

