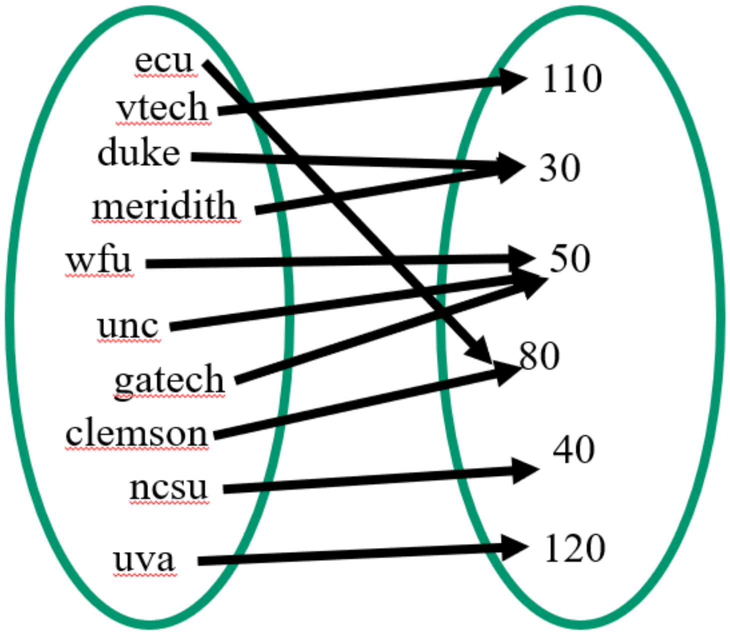
CompSci 101
Introduction to Computer Science



November 9, 2017

Prof. Rodger

Announcements

- Assign 7 due Monday
- APT 7 due Tuesday
- Exam 2 Thursday, November 16
  - See practice exams from Fall 16 and Spring 17

- Today:
  - More problem solving with dictionaries
  - Finish problem from last time

Be in the know….
ACM, compsci mailing lists

- Association of Computing Machinery (ACM)
  - Professional organization for computer science
  - Duke Student ACM Chapter – join for free
- Join duke email lists to find out info on jobs, events for compsci students
  - lists.duke.edu – join lists:
    - compsci – info from compsci dept
    - dukeacm – info from student chapter

Review Dictionaries

- Map keys to values
  - Counting: count how many times a key appears
    - Key to number
  - Store associated values
    - Key to list or set
- Get all
  - Keys, values or (key,value) pairs
- What question do you want to answer?
  - How to organize data to answer the question

Dictionary problems Number of students in Photo clubs bit.ly/101f17-1109-1

```
d = {'duke':30, 'unc':50, 'ncsu':40}


d['duke'] = 80
d.update({'ecu':40, 'uncc':70})
print  d.values()
```

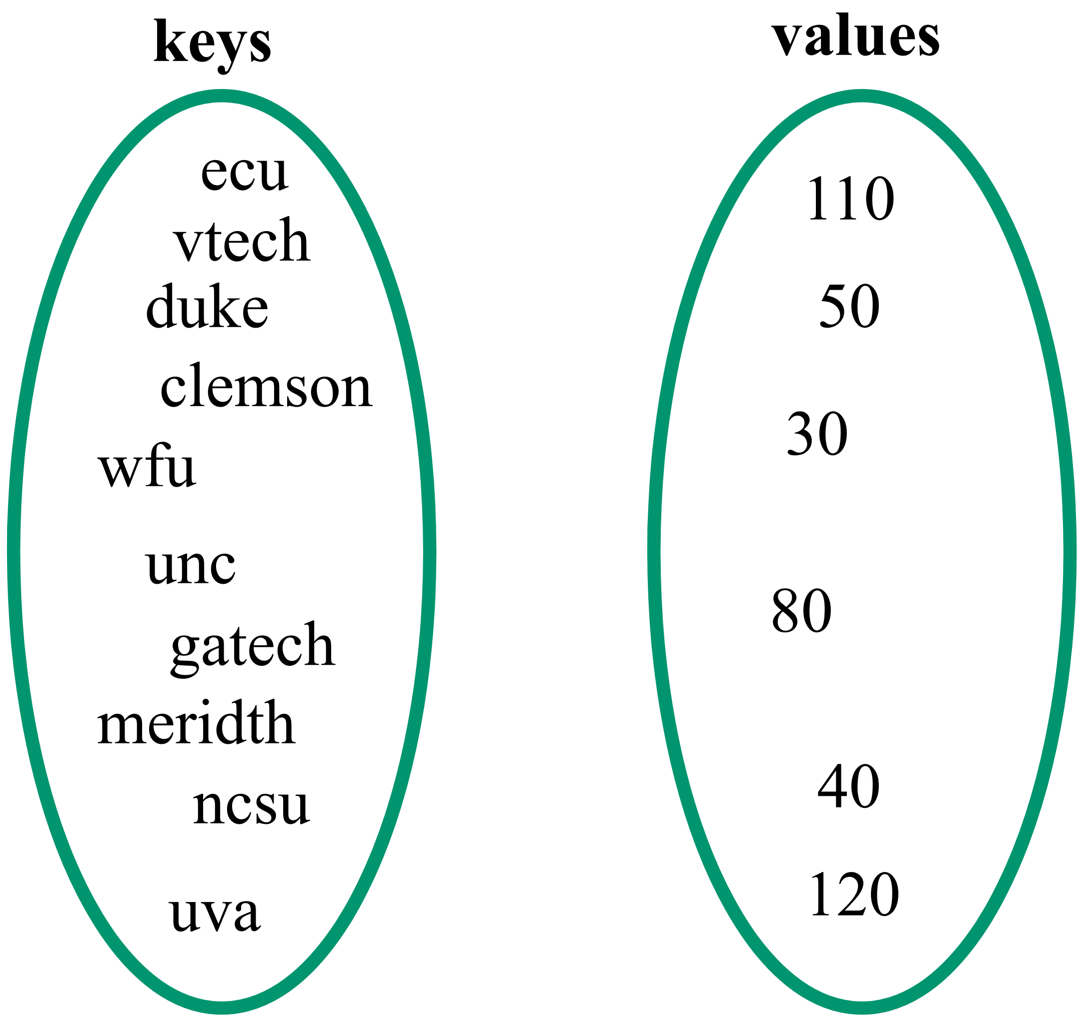
Dictionary problems – part 2 bit.ly/101f17-1109-2

- Consider the Python dictionary below maps schools to number of students in the Photo Club at their school

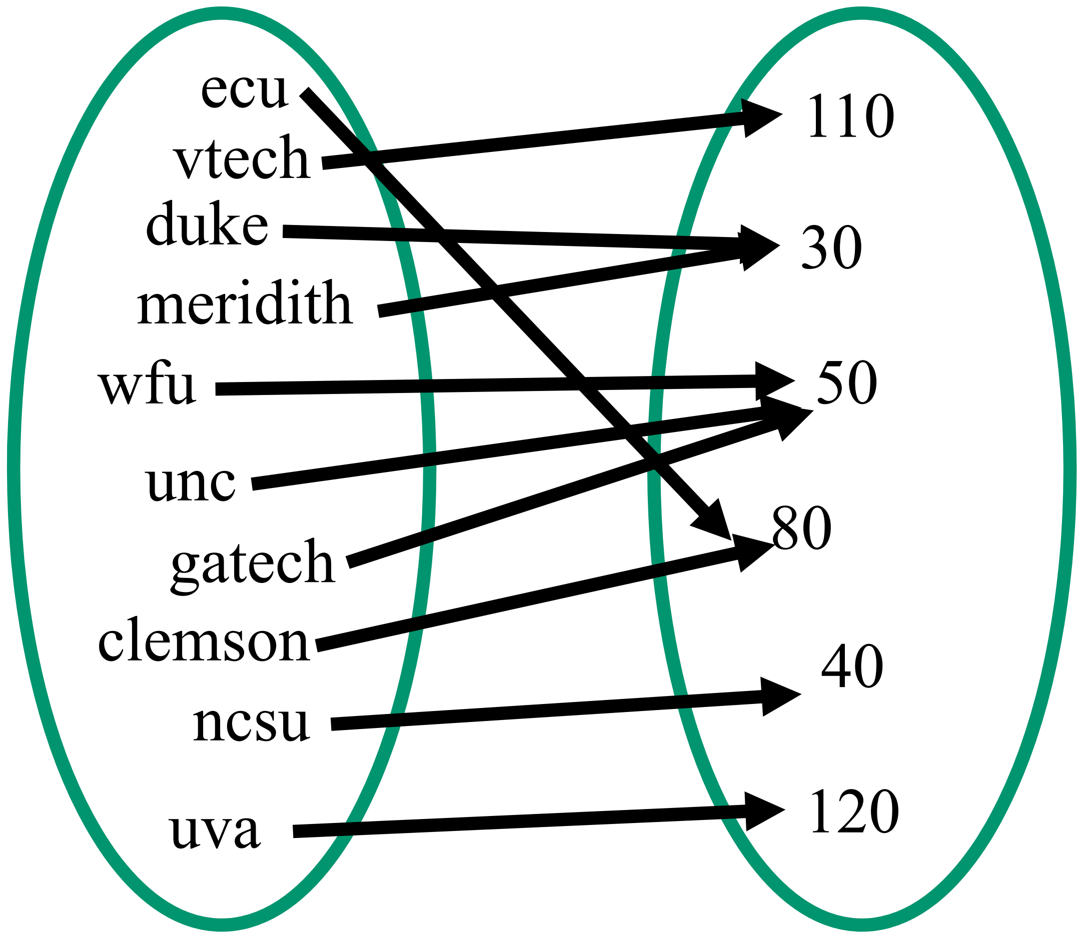
```
d = {'duke':30, 'unc':50, 'ncsu':40, 'wfu':50,
'ecu': 80, 'meridith':30, 'clemson':80,
'gatech':50, 'uva':120, 'vtech':110}
```

Dictionary to answer which schools have X students? … which schools have groups of students 1-49, 50-99, etc?

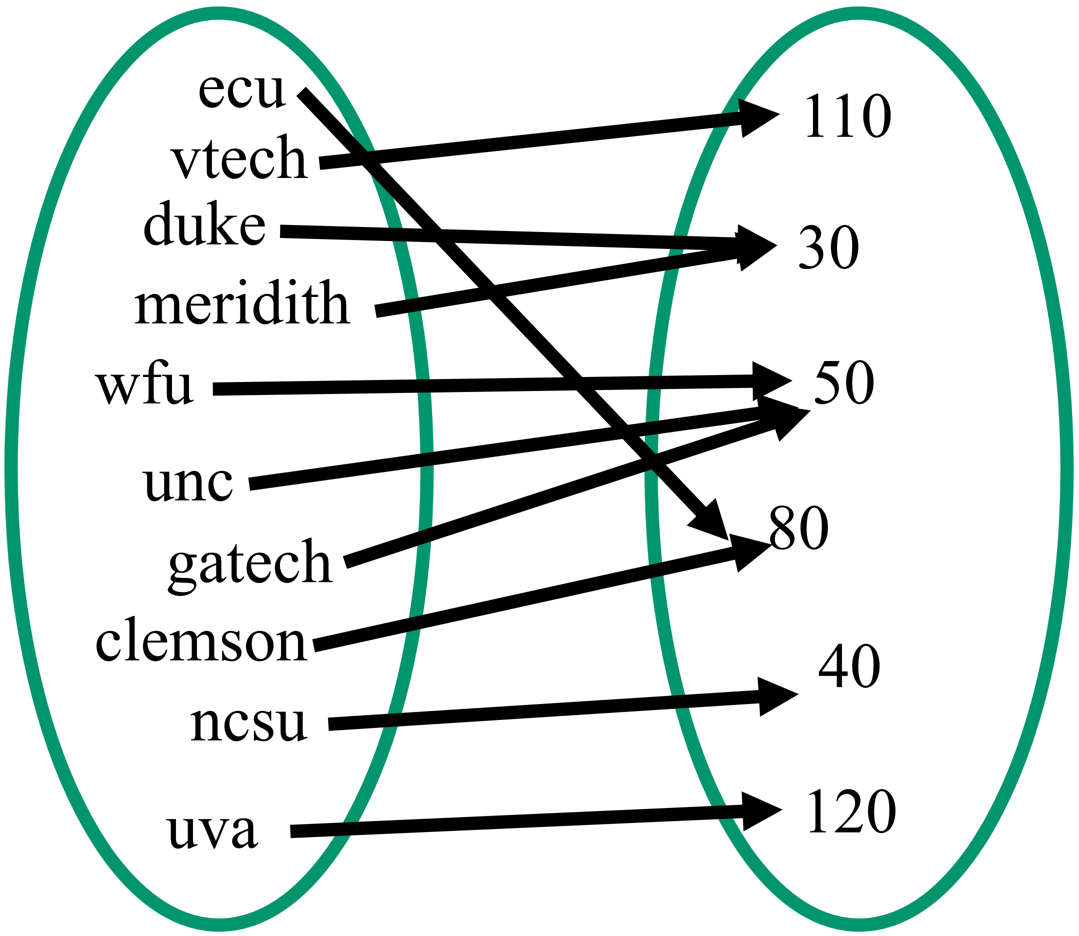
Dictionary of schools to number students



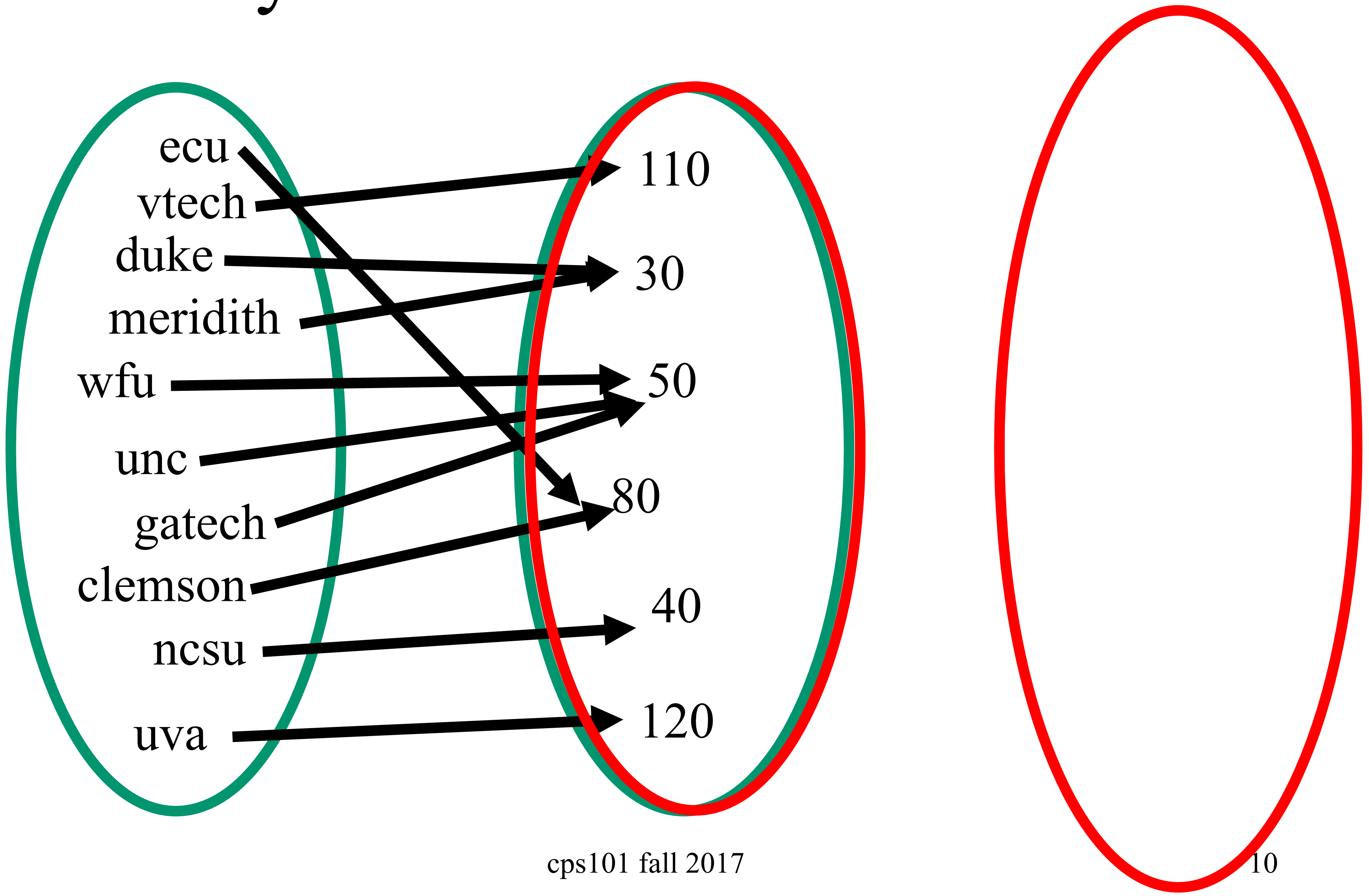
Dictionary of schools to number students



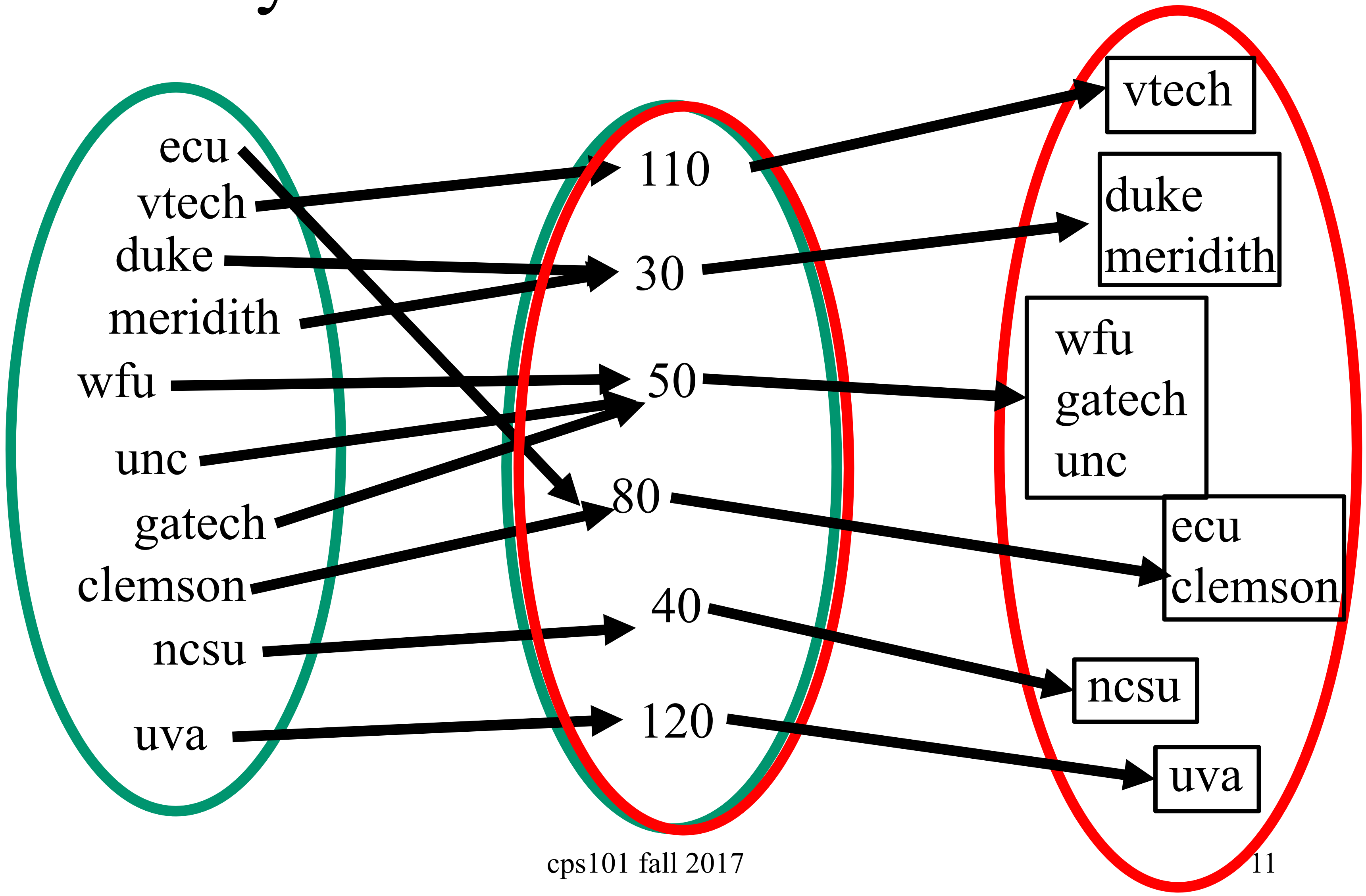
Dictionary of schools to number students
Dictionary of number students to schools



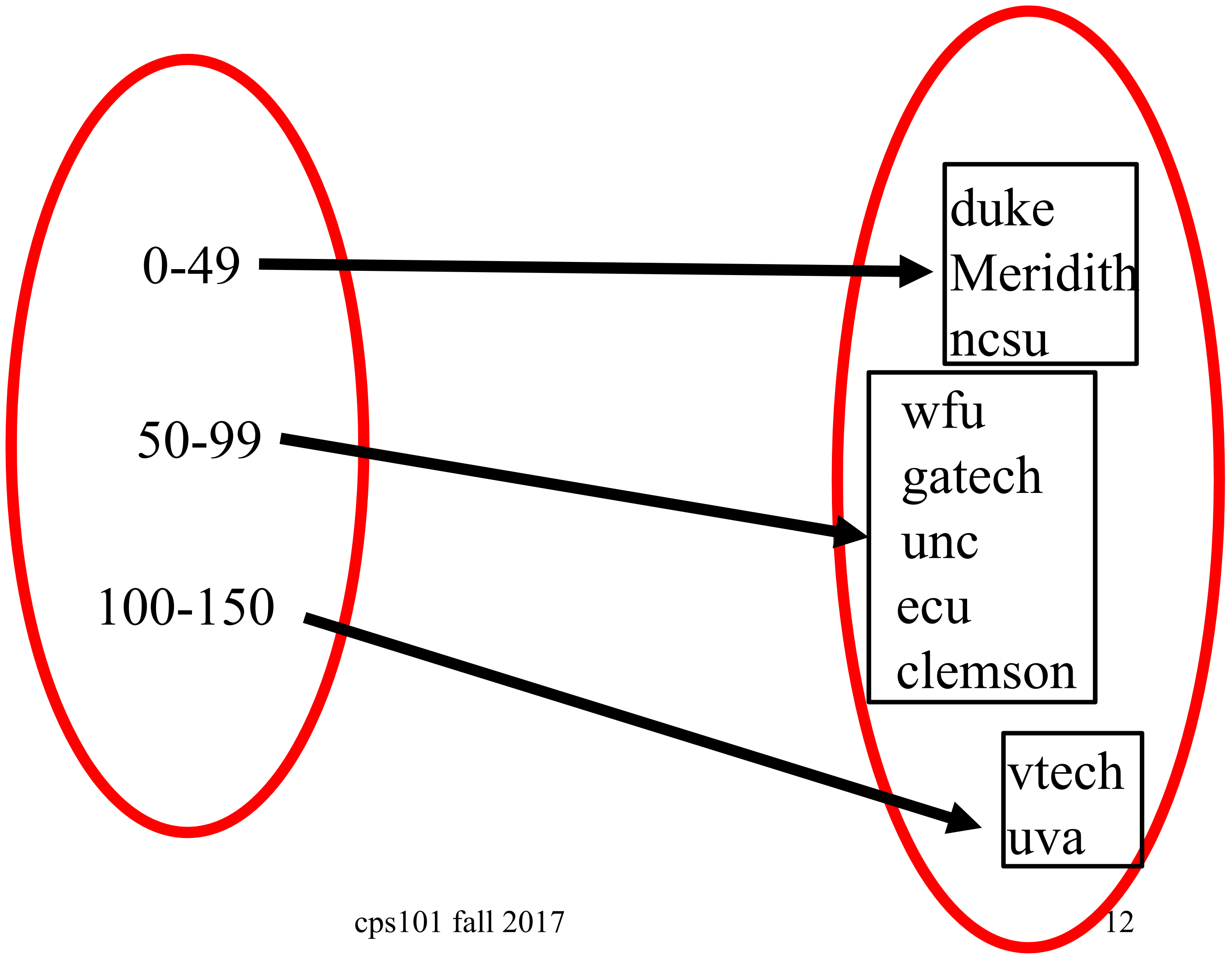
Dictionary of schools to number students
Dictionary of number students to schools



Dictionary of schools to number students
Dictionary of number students to schools



Dictionary of number groups to list of schools



Inverted Dictionary bit.ly/101f17-1109-3

- Start with dictionary of keys to values
  - *Schools to number of students*
- Use it to build an inverted dictionary of values to keys (actually list of keys)
  - *Number of students to list of schools*
- Lets look at the code

Dictionary Song problem bit.ly/101f17-1109-4

songs = [*"Hey Jude:Let it be:Day Tripper",*
"Let it be:Drive my car:Hey Jude",
"I want to hold your hand:Help!:Day Tripper",
"Born to run:Thunder road:She's the one",
"Hungry heart:The river:Born to run",
"The river:Thunder road:Drive my car",
"Angie:Start me up:Ruby Tuesday",
"Born to run:Angie:Drive my car"]

Building the dictionary d

"Hey Jude:Let it be:Day Tripper"

Building the dictionary d

"Hey Jude:Let it be:Day Tripper“

d[“Hey Jude”] = [1, 0, 0]

d[“Let it be”] = [0, 1, 0]

Building the dictionary d

"Hey Jude:Let it be:Day Tripper"

d["Hey Jude"] = [1, 0, 0]

d["Let it be"] = [0, 1, 0]

d["Day Tripper"] = [0, 0, 1]

Building the dictionary d

"Let it be:Drive my car:Hey Jude"

d[“Hey Jude”] = [1, 0, 0]

d[“Let it be”] = [0, 1, 0]

[1, 1, 0]

d[“Day Tripper”] = [0, 0, 1]

Building the dictionary d

"Let it be:Drive my car:Hey Jude"

d["Hey Jude"] =	[1, 0, 0]
d["Let it be"] =	[1, 1, 0]
d["Day Tripper"] =	[0, 0, 1]
d["Drive my car"] =	[0, 1, 0]

Building the dictionary d

"Let it be:Drive my car:Hey Jude"

d["Hey Jude"] = [[11,,00,,01]]

d["Let it be"] = [1, 1, 0]

d["Day Tripper"] = [0, 0, 1]

d["Drive my car"] = [0, 1, 0]

Building the dictionary d

"I want to hold your hand:Help!:Day Tripper"

d[“Hey Jude”] =	[1, 0, 1]
d[“Let it be”] =	[1, 1, 0]
d[“Day Tripper”] =	[0, 0, 1]
d[“Drive my car”] =	[0, 1, 0]
d[“I want to hold your hand”] =	[1, 0, 0]

Building the dictionary d

"I want to hold your hand:Help!:Day Tripper"

d[“Hey Jude”] =	[1, 0, 1]
d[“Let it be”] =	[1, 1, 0]
d[“Day Tripper”] =	[0, 0, 1]
d[“Drive my car”] =	[0, 1, 0]
d[“I want to hold your hand”] =	[1, 0, 0]
d[“Help!”] =	[0, 1, 0]

Building the dictionary d

"I want to hold your hand:Help!:Day Tripper"

d[“Hey Jude”] =	[1, 0, 1]
d[“Let it be”] =	[1, 1, 0]
d[“Day Tripper”] =	[0, 0, 2]
d[“Drive my car”] =	[0, 1, 0]
d[“I want to hold your hand”] =	[1, 0, 0]
d[“Help!”] =	[0, 1, 0]

Building the dictionary d

"I want to hold your hand:Help!:Day Tripper"

d["Hey Jude"] =	[1, 0, 1]
d["Let it be"] =	[1, 1, 0]
d["Day Tripper"] =	[0, 0, 2]
d["Drive my car"] =	[0, 1, 0]
d["I want to hold your hand"] =	[1, 0, 0]
d["Help!"] =	[0, 1, 0]

APT EmailsCourse
bit.ly/101f17-1109-5

You are given a list of strings of course information, where each string is in the format "coursename:person:email". Your task is to determine the course with the most people and to return the emails of those people in the largest course. The emails should be returned as a string with the emails in alphabetical order. If there is more than one largest course, return the emails of such course that comes first in alphabetical order.

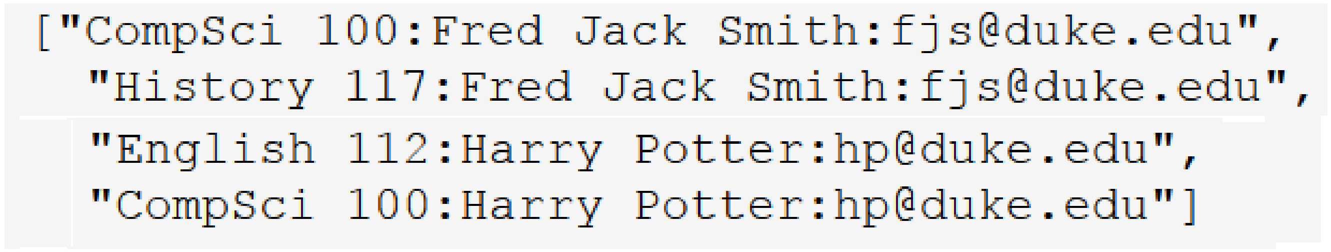
```
["CompSci 100:Fred Jack Smith:fjs@duke.edu",
  "History 117:Fred Jack Smith:fjs@duke.edu",
  "CompSci 102:Arielle Marie Johnson:amj@duke.edu",
  "CompSci 100:Arielle Marie Johnson:amj@duke.edu",
  "CompSci 006:Bertha White:bw@duke.edu",
  "Econ 051:Bertha White:bw@duke.edu",
  "English 112:Harry Potter:hp@duke.edu",
  "CompSci 100:Harry Potter:hp@duke.edu"]

Returns "amj@duke.edu fjs@duke.edu hp@duke.edu
```

Step 1 – Work small example by hand

```
["CompSci 100:Fred Jack Smith:fjs@duke.edu",
  "History 117:Fred Jack Smith:fjs@duke.edu",
  "English 112:Harry Potter:hp@duke.edu",
  "CompSci 100:Harry Potter:hp@duke.edu"]
```

Step 1 – Work small example by hand



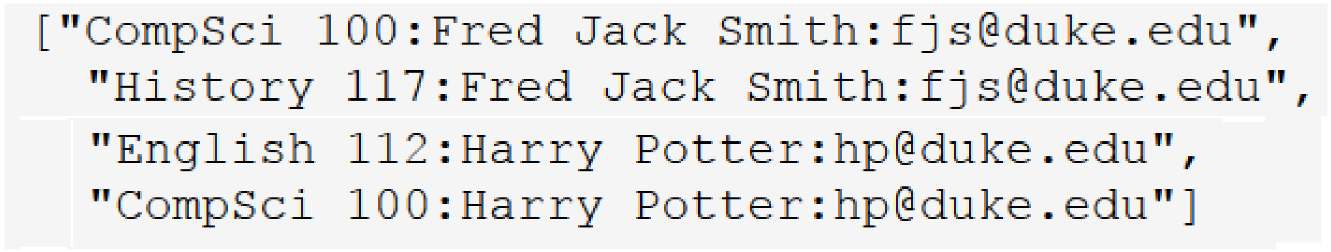
```
["CompSci 100:Fred Jack Smith:fjs@duke.edu",
  "History 117:Fred Jack Smith:fjs@duke.edu",
  "English 112:Harry Potter:hp@duke.edu",
  "CompSci 100:Harry Potter:hp@duke.edu"]
```

CompSci 100 => fjs@duke.edu

History 117 => fjs@duke.edu

English 112 => hp@duke.edu

Step 1 – Work small example by hand



```
["CompSci 100:Fred Jack Smith:fjs@duke.edu",
  "History 117:Fred Jack Smith:fjs@duke.edu",
  "English 112:Harry Potter:hp@duke.edu",
  "CompSci 100:Harry Potter:hp@duke.edu"]
```

CompSci 100 => fjs@duke.edu, hp@duke.edu

History 117 => fjs@duke.edu

English 112 => hp@duke.edu

Answer is: fjs@duke.edu, hp@duke.edu

Step 2 – Write down what you did

- Extracted out CompSci 101, and email
- Mapped CompSci 101 to fjs@duke.edu
- Extracted out History 117 and email
- Mapped History 117 to fjs@duke.edu
- Extacted out English 112 and email
- Mapped English 112 to hp@duke.edu
- Extracted out CompSci 101 and email
- Mapped CompSci 101 to another, hp@duke.edu

Step 3 – Generalize, find patterns

- Initialize structure for answer
- Initialize structure for mapping items
- For each item in the given list
  - Extract out course
  - Extract out the email
  - Map the course to email (need a list of emails)
- Find largest list of emails
- Sort email list and return

Step 4 – try another example

Step 5– Translate to code