Lecture Handout #7
February 9, 2016

Data Source: Historic grade data at UIUC

Austin Walters filed a Freedom of Information Act (FOIA) request to get historical grade distributions for all classes at UIUC from 2010-2014.

CS 205 Workbook Branch

A new release branch has been created for this experience. You can find the initial files in the branch `exp_historicGrades`.

Recall that every workbook project is set up with three directories:

-
-
-

Inside of py/compute.py, add the lines to read the CSV file:

```
f = open("res/Fall.2014.csv")
reader = csv.DictReader(f)
data = [row for row in reader]
```

The next line of Python will loop through each row in the data file:

```
for row in data:
```

After this line, what code can be used to:

- Access the average grade?
- Access the full course name (eg: "CS 205")?

Puzzle #1:

Find the row with the minimum GPA. What course was it and who was the instructor?

Your Courses

Let's consider your courses! What non-special topic course are you taking this semester (eg: "CS 205")?

My Courses	

In order to search for these courses, create an array of dictionaries of these courses. Each dictionary should contain one element called `courseNumber`. For example, if your courses are CS 125 and CS 173:

```
myCourses = [
  { "courseNumber": "CS 125" },
  { "courseNumber": "CS 173" }
]
```

The rationale for this setup is that, when we find a `myCourses` inside of data, we can add to that dictionary. One entry may become:

```
  { "courseNumber": "CS 125",
    "averageGrade": 3.605 }
```

Puzzle #2:

Find the grade information all of your courses from Fall 2014. Which one of your courses had the lowest average grade in Fall 2014?

Advanced questions/ideas about the data?

-
-
-
-
-

Histograms: Categorizing continuous data in buckets

One classic technique to understanding the bigger picture about the data is to use a histogram. The workbook already has a histogram visualization already created; you just need to create the Python code to output the data in the way that the visualization expects it!

Specifically, the visualization is looking for the following JSON:

```
[
  { "rangeTitle": "A+", "count": 385 },
  { "rangeTitle": "A", "count": 472 },
  { "rangeTitle": "A-", "count": 215 },
  ...
]
```

Since each element of the array is a dictionary, **it is okay to have additional elements in each dictionary**. It is up to you to decide how you want to "count" each range; there are multiple ways that one might think of counting the data.

Puzzle #3:

Create the JSON expected by the visualization and save it as "res/histogram.json". If your format is correct, you will see the visualization in the workbook!