Data 8 Foundations of Data Science
Spring 2017

MIDTERM SOLUTIONS

INSTRUCTIONS

- You have 45 minutes to complete the exam.
- The exam is closed book, closed notes, closed computer, closed calculator, except one hand-written 8.5" × 11" sheet of notes of your own creation and the official midterm exam reference guide provided with the exam.
- Mark your answers **on the exam itself**. We will *not* grade answers written on scratch paper.

Last name	
First name	
Student ID number	
BearFacts email (`_@berkeley.edu`)	
GSI	
Name of the person to your left	
Name of the person to your right	
All the work on this exam is my own. **(please sign)**	

1. (16 points) Expressions

A table named `pay` contains one row for each UC Berkeley faculty member and these columns:

- **dept**: a string, the department of the faculty member.
- **name**: a string, the first name of the faculty member.
- **role**: a string, one of: Assistant Professor, Associate Professor, Professor, or Lecturer
- **salary**: an int, last year's salary paid by the university.

dept	name	role	salary
Journalism	Jeremy	Lecturer	111,528
Economics	Christina	Professor	349,727
South & Southeast Asian Studies	Penelope	Associate Professor	127,119

`... (2056 rows omitted)`

Fill in the blanks of the Python expressions to compute the described values. You must use *all* and *only* the lines provided. The last (or only) line of each answer should evaluate to the value described. Assume that the statements `from datascience import *` and `import numpy as np` have been executed.

(a) (2 pt) The total salary amount paid to all faculty.

```
sum(pay.column('salary'))
```

(b) (3 pt) The name of the third highest paid faculty member. (Assume no two faculty have the same salary.)

```
pay.sort("salary", descending=True).column("name").item(2)
```

(c) (3 pt) The number of lecturers in the department that has the most lecturers. (One has more than the rest.)

```
max(pay.where('role', 'Lecturer').group('dept').column('count'))
```

(d) (3 pt) The average faculty salary after all faculty members get a 10% raise each year for three years.

```
np.average(pay.column('salary')) * (1.1) ** 3
```

(e) (5 pt) The table `big` (created below) only contains rows for faculty in departments that have both Lecturers and Professors. Using `big`, compute the minimum *pay gap* for any department. The *pay gap* is the absolute difference between a department's average `Lecturer` salary and its average `Professor` salary.

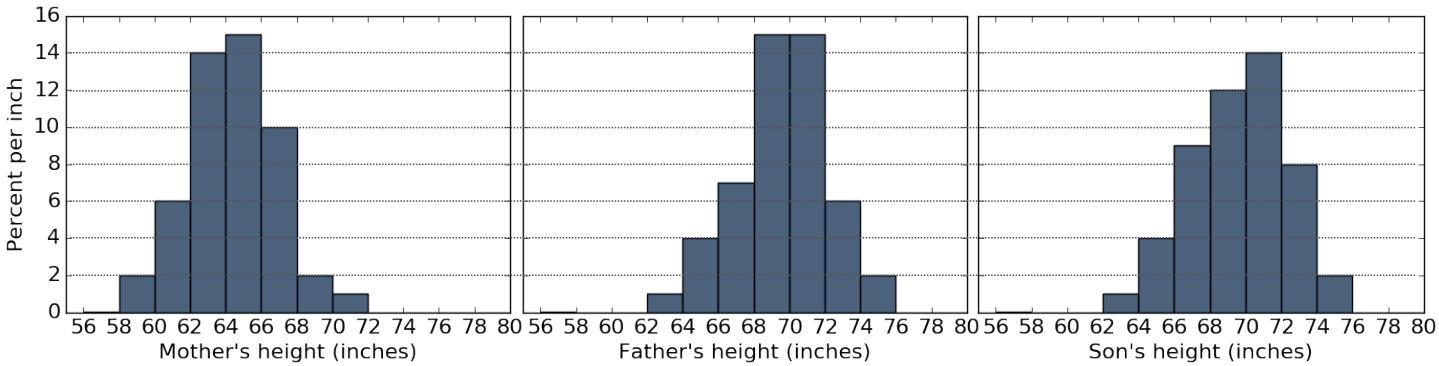
```
has_lecturer = pay.pivot('role', 'dept').where('Lecturer', are.above(0))
has_both = has_lecturer.where('Professor', are.above(0))
big = pay.where('dept', are.contained_in(has_both.column('dept')))
```

```
avg = big.pivot('role', 'dept', 'salary', np.average)
```

```
min(np.abs(avg.column('Lecturer') - avg.column('Professor')))
```

2. (14 points) Distributions

Galton measured the heights of the members of **200 families** that each included 1 mother, 1 father, and some varying number of adult sons. The three histograms of heights below depict the distributions for all mothers, fathers, and adult sons. **All bars are 2 inches wide. All bar heights are integers.** The heights of all people in the data set are included in the histograms.



(a) (8 pt) Calculate each quantity described below or write *Unknown* if there is not enough information above to express the quantity as a single number (not a range). Show your work!

i. The **percentage** of mothers that are at least 60 inches but less than 64 inches tall.

(60-64): 2 inches * (6 + 14) percent/inch = 40 percent
(66-70): 2 inches * (10 + 2) percent/inch = 24 percent;

ii. The **percentage** of fathers that are at least 64 but less than 67 inches tall.

Unknown: We can't tell how heights are distributed within a bin.

iii. The **number** of sons that are at least 70 inches tall.

Unknown: The total number of sons is unknown, so the size of any subset is unknown.

iv. The **number** of mothers that are at least 60 inches tall.

(100 percent - (2 inches * 2 percent/inch)) * 200 mothers = 192 mothers
(100 percent - (2 inches * 1 percent/inch)) * 200 fathers = 196 fathers

(b) (2 pt) If the father's histogram were redrawn, replacing the two bins from 72-to-74 and from 74-to-76 with one bin from 72-to-76, what would be the height of its bar? If it's impossible to tell, write *Unknown*.

Father's: The bin contains 6 * 2 + 2 * 2 = 16 percent, and the width is 4 inches, so the height is 4 percent/inch. Son's 5 percent/inch

(c) (4 pt) The percentage of sons that are taller than **all** of the mothers is between _____ and _____. Fill in the blanks in the previous sentence with the smallest range that can be determined from the histograms, then explain your answer below.

The tallest mother is between 70 and 72 inches. The proportion of sons above 72 inches is (8 + 2) percent/inch * 2 inches = 20 percent. The proportion of sons above 70 inches is (14 + 8 + 2) percent/inch * 2 inches = 48 percent.

3. (12 points) Probability

(a) (3 pt) A basket of 10 colored tickets contains 1 blue, 1 gold, 4 green, and 4 red tickets. If you draw 6 tickets uniformly at random with replacement, what is the chance that you draw at least one that is either blue or gold? Write your answer as a Python expression that computes the result exactly (no simulation).

```
1 - 0.8 ** 6 = 0.737856
```

(b) (5 pt) The `roll` function draws an empirical histogram of the number of results that are `k` or larger, when n fair 6-sided dice are rolled. For example, if `k` is 5, `n` is 3, and rolling 3 dice results in a 6, a 4, and a 5, then 2 of the 3 dice are 5 or larger (the 6 and the 5). Fill in the blanks to complete its implementation.

```
def roll(k, n, trials):
    """Repeatedly roll n dice and check how many results are k or larger."""
    outcomes = make_array()
    possible_results = np.arange(1. 7)
    for i in np.arange(trials):

        rolls = np.random.choice(possible_results, n)

        outcomes = np.append(outcomes, np.count_nonzero(rolls >= k))

    Table().with_column('Outcomes', outcomes).hist(bins=np.arange(30))
```

(c) (4 pt) For each of the histograms below, circle the letter of the expression that generated it, assuming `roll` is implemented correctly above.

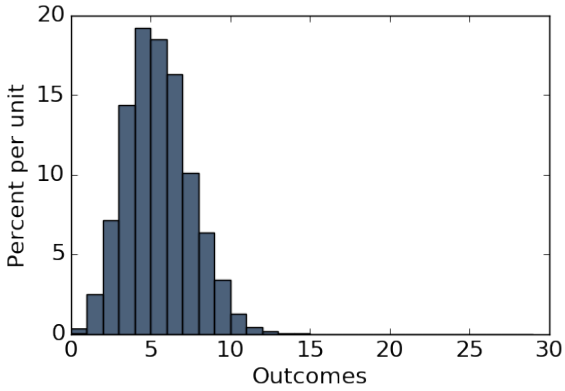
Circle one:

(A) `roll(4, 30, 5000)`

(B) *** `roll(6, 30, 5000)`

(C) `roll(4, 30, 50)`

(D) `roll(6, 30, 50)`



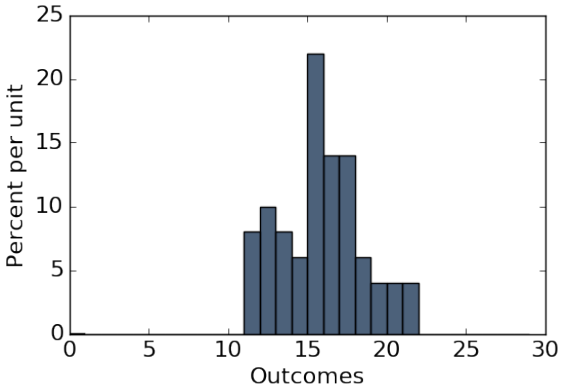
Circle One:

(A) `roll(4, 30, 5000)`

(B) `roll(6, 30, 5000)`

(C) *** `roll(4, 30, 50)`

(D) `roll(6, 30, 50)`



4. (8 points) Hypothesis Testing

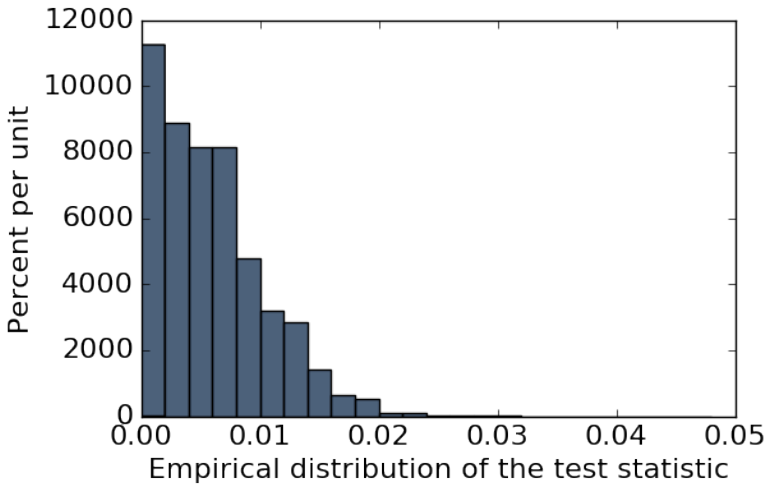
A recent paper called *Do animals bite more during a full moon?* analyzed 1,660 emergency room visits for animal bites over a two-year period. The table shows bite counts by the day of the lunar cycle. In the lunar cycle, the moon is full on days 29, 30, and 1. On all other days, the moon is not full.

Day	17, 18, 19	20, 21, 22	23, 24, 25	26, 27, 28	29, 30, 1	2, 3, 4	5, 6, 7	8, 9, 10	11, 12, 13	14, 15, 16
Bites	136	151	162	202	268	154	141	145	147	154

(a) (4 pt) Fill in the blanks of this null hypothesis so that it can be tested using the data above and so that it would help inform the question of whether there is a relation between bite frequency and the full moon.

The day of the lunar cycle on which an animal bites is chosen uniformly at random, so the chance that an animal will bite on days 29, 30, or 1 of the lunar cycle is $\frac{1}{10}$. Any deviation from this proportion is due to random chance.

(b) (4 pt) The histogram below shows the empirical distribution of a test statistic under the null hypothesis. It was generated by repeating the following process 1000 times: the lunar day for each of 1660 simulated animal bites was chosen uniformly at random from 1 to 30. A test statistic was computed for each resulting list of 1660 days: the absolute difference between $\frac{1}{10}$ and the proportion of bites on days 29, 30, and 1.



(*Note*: The proportion of observed bites on lunar days 29, 30, and 1 was $\frac{268}{1660} = 0.161$.)

Fill in each blank in the conclusion below:

We reject the null hypothesis because the observed value of the test statistic is 0.061, which is inconsistent with the null distribution.