

## Midterm Exam SOLUTIONS – DSC 10, Fall 2021

### Q1 Welcome to Flights! 🌍

3 Points

King Triton 🧜‍♂️, UCSD's mascot, is quite the traveler! Throughout this exam, we will be working with the `flights` DataFrame, which details several facts about each of the flights that King Triton has been on over the past few years. The first few rows of `flights` are shown below.

	DATE	FLIGHT	FROM	TO	DIST	HOURS	SEAT
0	2021-09-07	UA2016	EWR	SAN	2423	5.53	WINDOW
1	2021-09-04	DL4768	DTW	EWR	487	1.83	AISLE
2	2021-08-28	AA1355	DFW	DTW	987	2.73	MIDDLE
3	2021-08-28	AA648	SAN	DFW	1170	3.02	WINDOW
4	2021-08-17	AS3355	SJC	SAN	418	1.38	AISLE

Here's a description of the columns in `flights`:

- `'DATE'`: the date on which the flight occurred. Assume that there were no "redeye" flights that spanned multiple days.
- `'FLIGHT'`: the flight number. Note that this is not unique; airlines reuse flight numbers on a daily basis.
- `'FROM'` and `'TO'`: the 3-letter airport code for the departure and arrival airports, respectively. Note that it's not possible to have a flight from and to the same airport.
- `'DIST'`: the distance of the flight, in miles.
- `'HOURS'`: the length of the flight, in hours.
- `'SEAT'`: the kind of seat King Triton sat in on the flight; the only possible values are `'WINDOW'`, `'MIDDLE'`, and `'AISLE'`.

### Q1.1

2 Points

Which of these would it make sense to use as the index of `flights`?

- 'DATE'
- 'FLIGHT'
- 'FROM'
- 'TO'
- None of these are good choices for the index

### Q1.2

1 Point

What type of variable is `'FLIGHT'`?

- Categorical
- Numerical

## Q2 America's Finest City 🌞

10 Points

### Q2.1

2 Points

Which of these correctly evaluates to the number of flights King Triton took to San Diego (airport code 'SAN')?

- `flights.loc['SAN'].shape[0]`
- `flights[flights.get('TO') == 'SAN'].shape[0]`
- `flights[flights.get('TO') == 'SAN'].shape[1]`
- `len(flights.sort_values('TO', ascending=False).loc['SAN'])`

## Q2.2

5 Points

Fill in the blanks below so that the result also evaluates to the number of flights King Triton took to San Diego (airport code 'SAN').

```
flights.groupby(__(a)__).count().get('FLIGHT').__(b)__
```

**Q2.2.1 (1 Point)** What goes in blank (a)?

- 'DATE'
- 'FLIGHT'
- 'FROM'
- 'TO'

**Q2.2.2 (2 Points)** What goes in blank (b)?

- .index[0]
- .index[-1]
- .loc['SAN']
- .iloc['SAN']
- .iloc[0]

**Q2.2.3 (2 Points)** True or False: If we change `.get('FLIGHT')` to `.get('SEAT')`, the results of the above code block will not change. (You may assume you answered the previous two subparts correctly.)

- True
- False

## Q2.3

3 Points

For your convenience, we show the first few rows of `flights` again.

	DATE	FLIGHT	FROM	TO	DIST	HOURS	SEAT
0	2021-09-07	UA2016	EWR	SAN	2423	5.53	WINDOW
1	2021-09-04	DL4768	DTW	EWR	487	1.83	AISLE
2	2021-08-28	AA1355	DFW	DTW	987	2.73	MIDDLE
3	2021-08-28	AA648	SAN	DFW	1170	3.02	WINDOW
4	2021-08-17	AS3355	SJC	SAN	418	1.38	AISLE

Consider the DataFrame `san`, defined below.

```
san = flights[(flights.get('FROM') == 'SAN') & (flights.get('TO') == 'SAN')]
```

Which of these DataFrames **must** have the same number of rows as `san`?

- `flights[(flights.get('FROM') == 'SAN') and (flights.get('TO') == 'SAN')]`
- `flights[(flights.get('FROM') == 'SAN') | (flights.get('TO') == 'SAN')]`
- `flights[(flights.get('FROM') == 'LAX') & (flights.get('TO') == 'SAN')]`
- `flights[(flights.get('FROM') == 'LAX') & (flights.get('TO') == 'LAX')]`

### Q3 Speed

4 Points

For your convenience, we show the first few rows of `flights` again.

	DATE	FLIGHT	FROM	TO	DIST	HOURS	SEAT
0	2021-09-07	UA2016	EWR	SAN	2423	5.53	WINDOW
1	2021-09-04	DL4768	DTW	EWR	487	1.83	AISLE
2	2021-08-28	AA1355	DFW	DTW	987	2.73	MIDDLE
3	2021-08-28	AA648	SAN	DFW	1170	3.02	WINDOW
4	2021-08-17	AS3355	SJC	SAN	418	1.38	AISLE

Fill in the blanks below so that the result is a DataFrame with the same columns as `flights` plus a new column, `'SPEED'`, containing the average speed of each flight, in miles per hour.

```
flights.__(a)__(SPEED=__ (b) __)
```

#### Q3.1

1 Point

What goes in blank (a)?

- `groupby`
- `assign`
- `rename`
- `drop`
- `merge`

#### Q3.2

3 Points

What goes in blank (b)?

```
flights.get('DIST') / flights.get('HOURS')
```

## Q4 Seasons

8 Points

We define the seasons as follows:

<b>Season</b>	<b>Months</b>
Spring	March, April, May
Summer	June, July, August
Fall	September, October, November
Winter	December, January, February

## Q4.1

6 Points

We want to create a function `date_to_season` that takes in a `date` as formatted in the `'DATE'` column of `flights` and returns the season corresponding to that date. Which of the following implementations of `date_to_season` works correctly? **Select all that apply.**

Option 1:

```
def date_to_season(date):
    month_as_num = int(date.split('-')[1])
    if month_as_num >= 3 and month_as_num < 6:
        return 'Spring'
    elif month_as_num >= 6 and month_as_num < 9:
        return 'Summer'
    elif month_as_num >= 9 and month_as_num < 12:
        return 'Fall'
    else:
        return 'Winter'
```

Option 2:

```
def date_to_season(date):
    month_as_num = int(date.split('-')[1])
    if month_as_num >= 3 and month_as_num < 6:
        return 'Spring'
    if month_as_num >= 6 and month_as_num < 9:
        return 'Summer'
    if month_as_num >= 9 and month_as_num < 12:
        return 'Fall'
    else:
        return 'Winter'
```

Option 3:

```
def date_to_season(date):
    month_as_num = int(date.split('-')[1])
    if month_as_num < 3:
        return 'Winter'
    elif month_as_num < 6:
        return 'Spring'
    elif month_as_num < 9:
        return 'Summer'
    elif month_as_num < 12:
        return 'Fall'
    else:
        return 'Winter'
```



- Option 1
- Option 2
- Option 3
- None of these implementations of `date_to_season` work correctly

## Q4.2

2 Points

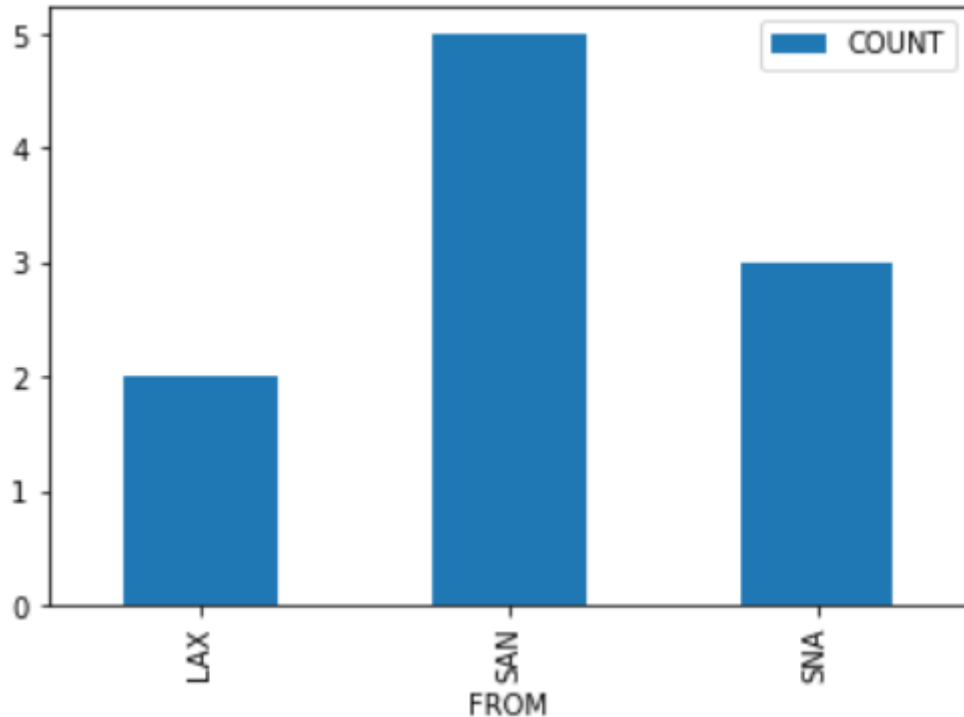
Assuming we've defined `date_to_season` correctly in the previous part, which of the following lines of code correctly computes the season for each flight in `flights`?

- `date_to_season(flights.get('DATE'))`
- `date_to_season.apply(flights).get('DATE')`
- `flights.apply(date_to_season).get('DATE')`
- `flights.get('DATE').apply(date_to_season)`

## Q5 West Coast Best Coast 🌴

2 Points

Suppose we create a DataFrame called `socal` containing only King Triton's flights departing from SAN, LAX, or SNA (John Wayne Airport in Orange County). `socal` has 10 rows; the bar chart below shows how many of these 10 flights departed from each airport.



Consider the DataFrame that results from merging `socal` with itself, as follows:

```
double_merge = socal.merge(socal, left_on='FROM', right_on='FROM')
```

How many rows does `double_merge` have?

=38+0

## Q6 Routes

4 Points

For your convenience, we show the first few rows of `flights` again.

	DATE	FLIGHT	FROM	TO	DIST	HOURS	SEAT
0	2021-09-07	UA2016	EWR	SAN	2423	5.53	WINDOW
1	2021-09-04	DL4768	DTW	EWR	487	1.83	AISLE
2	2021-08-28	AA1355	DFW	DTW	987	2.73	MIDDLE
3	2021-08-28	AA648	SAN	DFW	1170	3.02	WINDOW
4	2021-08-17	AS3355	SJC	SAN	418	1.38	AISLE

We define a "route" to be a departure and arrival airport pair. For example, all flights from `'SFO'` to `'SAN'` make up the "SFO to SAN route". This is different from the "SAN to SFO route".

Fill in the blanks below so that `most_frequent.get('FROM').iloc[0]` and `most_frequent.get('TO').iloc[0]` correspond to the departure and destination airports of the route that King Triton has spent the **most time flying on**.

```
most_frequent = flights.groupby(__(a)__).__(b)____
most_frequent = most_frequent.reset_index().sort_values(__(c)____)
```

### Q6.1

2 Points

What goes in blank (a)?

```
['FROM', 'TO']
```

## Q6.2

1 Point

What goes in blank (b)?

- `count()`
- `mean()`
- `sum()`
- `max()`

## Q6.3

1 Point

What goes in blank (c)?

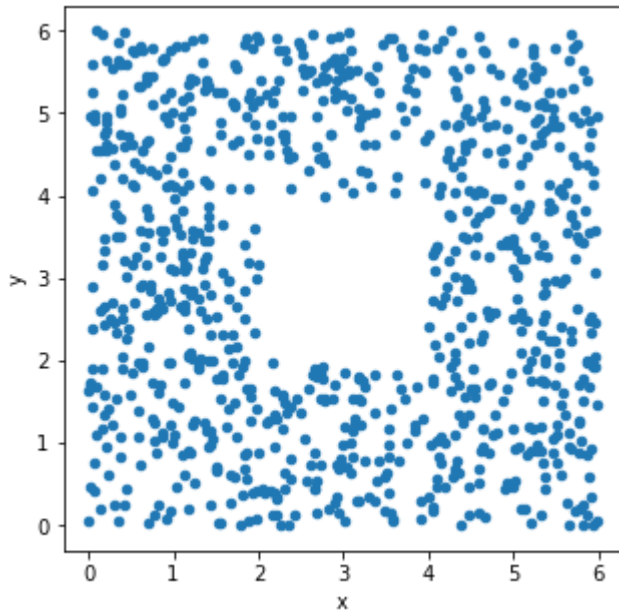
- `by='HOURS', ascending=True`
- `by='HOURS', ascending=False`
- `by='HOURS', descending=True`
- `by='DIST', ascending=False`

## Q7 Fingerprints 🙌

6 Points

The seat-back TV on one of King Triton's more recent flights was very dirty and was full of fingerprints. The fingerprints made an interesting pattern. We've stored the x and y positions of each fingerprint in the DataFrame `fingerprints`, and created the following scatterplot using

```
fingerprints.plot(kind='scatter', x='x', y='y')
```



## Q7.1

2 Points

True or False: The histograms that result from the following two lines of code will look very similar.

```
fingerprints.plot(kind='hist',  
                  y='x',  
                  density=True,  
                  bins=np.arange(0, 8, 2))
```

and

```
fingerprints.plot(kind='hist',  
                  y='y',  
                  density=True,  
                  bins=np.arange(0, 8, 2))
```

- True
- False

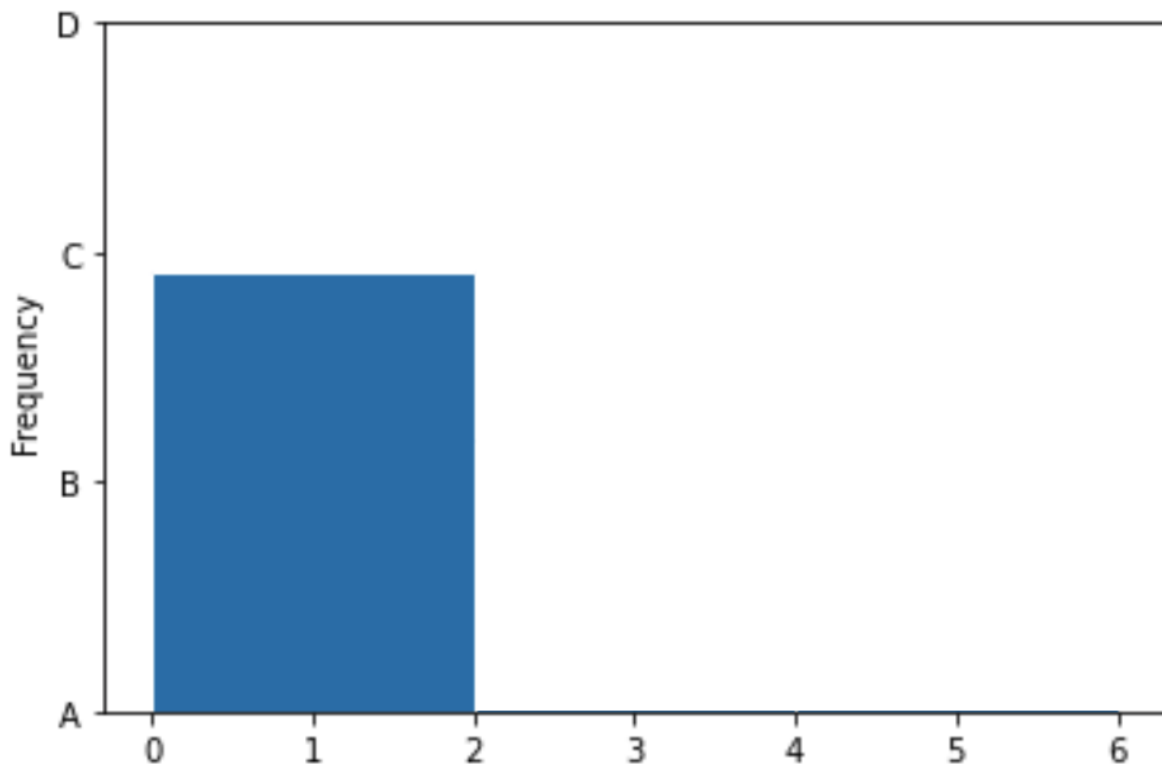
## Q7.2

4 Points

Below, we've drawn a histogram using the line of code

```
fingerprints.plot(kind='hist',  
                  y='x',  
                  density=True,  
                  bins=np.arange(0, 8, 2))
```

However, our Jupyter Notebook was corrupted, and so the resulting histogram doesn't quite look right. While the height of the first bar is correct, the histogram doesn't contain the second or third bars, and the y-axis is replaced with letters.



**Q7.2.1 (2 Points)** Which of the four options on the y-axis is closest to where the height of the middle bar should be?

- A
- B
- C
- D

**Q7.2.2 (2 Points)** Which of the four options on the y-axis is closest to where the height of the right-most bar should be?

A

B

C

D



## Q8 Airlines

6 Points

It turns out that King Triton is so busy that he doesn't even book his own flights – he has a travel agent who books his flights for him. He doesn't get to choose the airline that he flies on, but his travel agent gave him the following table, which describes the probability of each of his flights in 2022 being on Delta, United, American, or another airline:

Airline	Chance
Delta	0.4
United	0.3
American	0.2
All other airlines	0.1

The airline for one flight has no impact on the airline for another flight.

For this question, suppose that King Triton schedules 3 flights for January 2022.

### Q8.1

2 Points

What is the probability that all 3 flights are on United? Give your answer as an **exact** decimal between 0 and 1 (not a Python expression).

=0.027+-0

### Q8.2

2 Points

What is the probability that all 3 flights are on Delta, or all on United, or all on American? Give your answer as an **exact** decimal between 0 and 1 (not a Python expression).

=0.099+-0

### Q8.3

2 Points

True or False: The probability that all 3 flights are on the same airline is equal to the probability you computed in the previous subpart.

True

False

## Q9 Snack Time 🍪

6 Points

King Triton has boarded a Southwest flight. For in-flight refreshments, Southwest serves four types of cookies – chocolate chip, gingerbread, oatmeal, and peanut butter.

The flight attendant comes to King Triton with a box containing 10 cookies:

- 4 chocolate chip
- 3 gingerbread
- 2 oatmeal, and
- 1 peanut butter

The flight attendant tells King Triton to grab 2 cookies out of the box without looking.

Fill in the blanks below to implement a simulation that estimates the probability that both of King Triton's selected cookies are the same.

```
# 'cho' stands for chocolate chip, 'gin' stands for gingerbread,  
# 'oat' stands for oatmeal, and 'pea' stands for peanut butter.  
  
cookie_box = np.array(['cho', 'cho', 'cho', 'cho', 'gin',  
                       'gin', 'gin', 'oat', 'oat', 'pea'])  
  
repetitions = 10000  
prob_both_same = 0  
for i in np.arange(repetitions):  
    grab = np.random.choice(__(a)__)  
    if __(b)__:  
        prob_both_same = prob_both_same + 1  
prob_both_same = __(c)__
```

### Q9.1

2 Points

What goes in blank (a)?

- cookie\_box, repetitions, replace=False
- cookie\_box, 2, replace=True
- cookie\_box, 2, replace=False
- cookie\_box, 2

## Q9.2

2 Points

What goes in blank (b)?

```
grab[0] == grab[1]
```

## Q9.3

2 Points

What goes in blank (c)?

- `prob_both_same / repetitions`
- `prob_both_same / 2`
- `np.mean(prob_both_same)`
- `prob_both_same.mean()`

## Q10 Middle Seats

3 Points

In response to the pandemic, some airlines chose to leave middle seats empty, while others continued seating passengers in middle seats. Let's suppose Delta did not seat passengers in middle seats during the pandemic, and United did seat passengers in middle seats during the pandemic.

Delta wants to know whether customers were satisfied with them for making this decision not to use middle seats. Suppose they have access to a dataset of customer satisfaction surveys, taken annually for each airline. How can Delta determine whether its new seating policy caused an increase in customer satisfaction?

- Compare Delta's average customer satisfaction before and after this change went into effect.
- Compare Delta's average customer satisfaction after the change went into effect to United's average customer satisfaction at the same point in time.
- Compare the change in Delta's average customer satisfaction to the change in United's average customer satisfaction, throughout the same period of time, spanning the change.
- None of the above.