Midterm Exam - DSC 10, Spring 2022

Instructions:

- This exam consists of 12 questions, worth a total of 60 points.
- Write your PID or name in the top right of each page in the space provided.
- Please write neatly in the provided answer boxes. We will not grade work that appears elsewhere.
- Completely fill in bubbles and square boxes.
  - ◯ A bubble means that you should only **select one choice**.
  - ☐ A square box means you should **select all that apply**.
- You may refer to the DSC 10 reference sheet only. No other resources or technology (including calculators) are permitted.

Full Name:

PID:

Time: ◯ 10-10:50am
◯ 11-11:50am

By signing below, you are agreeing that you will behave honestly and fairly during and after this exam. You should not discuss any part of this exam with anyone enrolled in the course who has not yet taken the exam (this includes posting questions about the exam on Campuswire!)

Signature:

Please do not open your exam until instructed to do so.

Welcome to Sun God!

After a two-year hiatus due to the pandemic, UCSD's annual music festival, the Sun God festival, is back this year! In this exam, we'll be looking at a DataFrame named `sungod` that contains information on the artists who have performed at Sun God in years past. **For each year that the festival was held, we have one row for each artist that performed that year.** The columns are:

- `"Year" (int)`: the year of the festival
- `"Artist" (str)`: the name of the artist
- `"Appearance_Order" (int)`: the order in which the artist appeared in that year's festival (1 means they came onstage first)

The rows of `sungod` are arranged in **no particular order**. The first few rows of `sungod` are shown below (though `sungod` has **many more rows** than pictured here).

	Year	Artist	Appearance_Order
0	1993	Blues Traveler	1
1	2007	Third Eye Blind	4
2	2019	Vince Staples	3
3	2007	Ben Kweller	1
4	2015	OK Go	3

Assume:

- Only one artist ever appeared at a time (for example, we can't have two separate artists with a `"Year"` of `2015` and an `"Appearance_Order"` of `3`).
- An artist may appear in multiple different Sun God festivals (they could be invited back).
- We have already run `import babypandas as bpd` and `import numpy as np`.

Throughout this exam, we will refer to `sungod` repeatedly.

Question 1 (4 points)

Which of the following is a valid reason **not** to set the index of `sungod` to `"Artist"`? **Select all correct answers.**

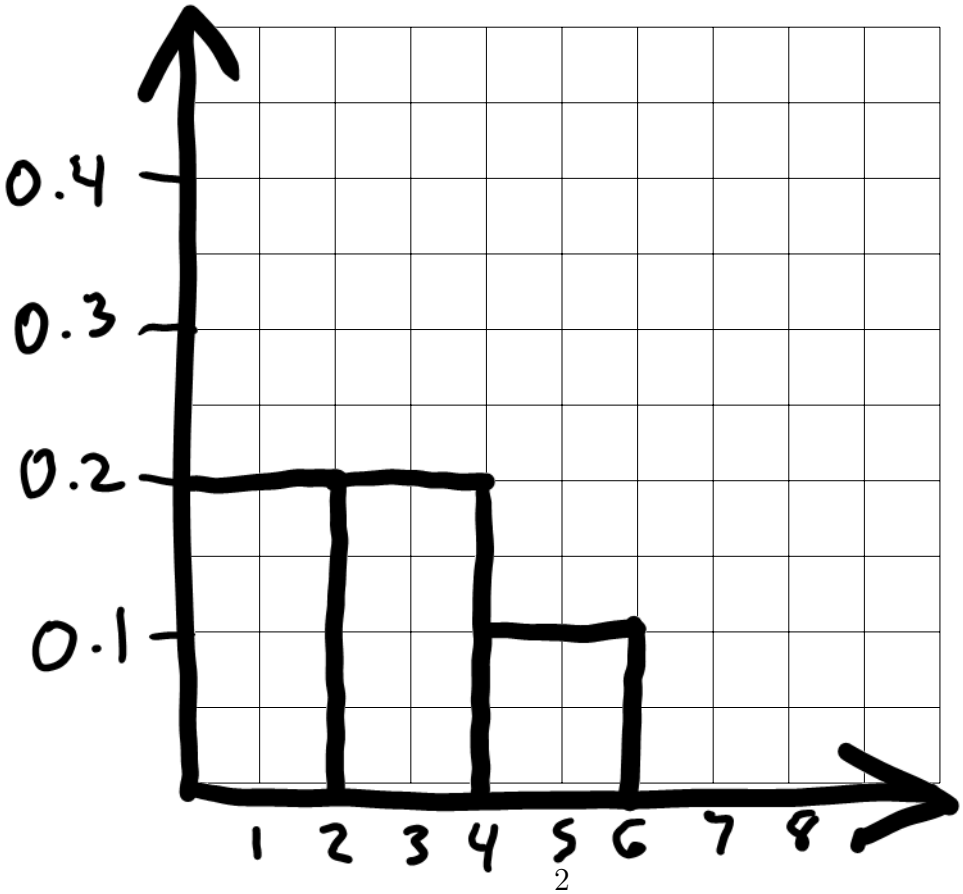
- [x] Two different artists have the same name.
- [x] An artist performed at Sun God in more than one year.
- [ ] Several different artists performed at Sun God in the same year.
- [ ] Many different artists share the same value of `"Appearance_Order"`.
- [ ] None of the above.

Question 2 (6 points)

On the graph paper below, draw the histogram that would be produced by this code.

```
(
sungod.take(np.arange(5))
      .plot(kind="hist", density=True,
      bins=np.arange(0, 7, 2), y="Appearance_Order");
)
```

In your drawing, make sure to label the height of each bar in the histogram on the vertical axis. You can scale the axes however you like, and the two axes don't need to be on the same scale.



Question 3 (4 points)

Suppose in a new cell, we type the following.

```
sungod.sort_values(by="Year")
```

After we run that cell, we type the following in a second cell.

```
sungod.get("Artist").iloc[0]
```

What is the output when we run the second cell? Note that the first Sun God festival was held in 1983.

● `"Blues Traveler"`

◯ The artist who appeared on stage first in 1983.

◯ An artist who appeared in 1983, but not necessarily the one who appeared first.

◯ Not enough information to tell.

Question 4 (4 points)

Write one line of code below to create a DataFrame called `openers` containing the artists that appeared first on stage at a past Sun God festival. The DataFrame `openers` should have all the same columns as `sungod`.

It's okay if you need to write on two lines to fit into the provided space, as long as your answer would be one line of Python code.

Solution: `openers = sungod[sungod.get("Appearance_Order")==1]`

Question 5 (4 points)

What was the largest number of artists that ever performed in a single Sun God festival? **Select all expressions that evaluate to the correct answer.**

☐ `sungod.groupby("Appearance_Order").count().get("Year").max()`

■ `sungod.groupby("Year").count().get("Artist").max()`

■ `sungod.get("Appearance_Order").max()`

☐ `sungod.groupby("Year").max().get("Year").max()`

☐ None of the above.

Question 6 (4 points)

Fill in the blank in the code below so that `chronological` is a DataFrame with the same rows as `sungod`, but ordered chronologically by appearance on stage. That is, earlier years should come before later years, and within a single year, artists should appear in the DataFrame in the order they appeared on stage at Sun God. Note that `groupby` automatically sorts the index in ascending order.

```
chronological = sungod.groupby(___________).max().reset_index()
```

◯ `["Year", "Artist", "Appearance_Order"]`

● `["Year", "Appearance_Order"]`

◯ `["Appearance_Order", "Year"]`

◯ None of the above.

Question 7 (4 points)

Another DataFrame called `music` contains a row for every music artist that has ever released a song. The columns are:

- `"Name" (str)`: the name of the music artist
- `"Genre" (str)`: the primary genre of the artist
- `"Top_Hit" (str)`: the most popular song by that artist, based on sales, radio play, and streaming
- `"Top_Hit_Year" (str)`: the year in which the top hit song was released

You want to know how many musical genres have been represented at Sun God since its inception in 1983. Which of the following expressions produces a DataFrame called `merged` that could help determine the answer?

◯ `merged = sungod.merge(music, left_on="Year", right_on="Top_Hit_Year")`

◯ `merged = music.merge(sungod, left_on="Year", right_on="Top_Hit_Year")`

● `merged = sungod.merge(music, left_on="Artist", right_on="Name")`

◯ `merged = music.merge(sungod, left_on="Artist", right_on="Name")`

Question 8 (6 points)

Consider an artist that has only appeared once at Sun God. At the time of their Sun God performance, we'll call the artist

- **outdated** if their top hit came out more than five years prior,
- **trending** if their top hit came out within the five years prior, and
- **up-and-coming** if their top hit came out after they appeared at Sun God.

Complete the function below so it outputs the appropriate description for any input artist who has appeared exactly once at Sun God.

```
def classify_artist(artist):
    filtered = merged[merged.get("Artist") == artist]
    year = filtered.get("Year").iloc[0]
    top_hit_year = filtered.get("Top_Hit_Year").iloc[0]
    if ___(a)___ > 0:
        return "up-and-coming"
    elif ___(b)___:
        return "outdated"
    else:
        return "trending"
```

a) What goes in blank (a)?

Solution: `top_hit_year-year`

b) What goes in blank (b)?

Solution: `year-top_hit_year>5`

Question 9 (4 points)

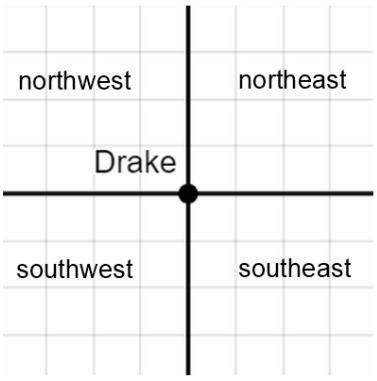
The expression below evaluates to `True`.

```
(
classify_artist("Michelle Branch")=="outdated"
and
classify_artist("Drake")=="trending"
)
```

Consider the scatterplot created by the code below.

```
merged.plot(kind="scatter", x="Year", y="Top_Hit_Year");
```

The point for Drake is somewhere in this scatterplot. Relative to the point for Drake, there are four quadrants of the scatterplot, as shown below.



There is one quadrant in which the point for Michelle Branch **cannot** appear. Which is it?

◯ northeast

● northwest

◯ southwest

◯ southeast

Question 10 (4 points)

In 2014, the UCSD administration made some important changes to Sun God policies, including:

1. eliminating guest tickets for non-students,
2. increasing security, and
3. introducing on-site medical care.

These changes were implemented because of incidents related to drug and alcohol abuse at the festival. At the 2013 Sun God festival, 48 students were hospitalized, and at the 2014 festival, only 8 students were hospitalized. Assuming there was no change in the total number of attendees from 2013 to 2014, which of the following statements is correct?

◯ We cannot be sure if there is an association between administrative changes and hospitalizations.

◯ There is an association between administrative changes and hospitalizations. One or more of the administrative changes is responsible for the decrease in hospitalizations, but since several administrative changes happened at the same time, we can't be sure of which one to credit with the reduction in hospitalizations.

● There is an association between administrative changes and hospitalizations. We can't be sure if any of the administrative changes are responsible for the reduction in hospitalizations.

◯ None of the above.

Question 11 (10 points)

The fine print of the Sun God festival website says "Ticket does not guarantee entry. Venue subject to capacity restrictions." RIMAC field, where the 2022 festival will be held, has a capacity of 20,000 people. Let's say that UCSD distributes 21,000 tickets to Sun God 2022 because prior data shows that 5% of tickets distributed are never actually redeemed. Let's suppose that each person with a ticket this year has a 5% chance of not attending (independently of all others). What is the probability that at least one student who has a ticket cannot get in due to the capacity restriction? Fill in the blanks in the code below so that `prob_angry_student` evaluates to an approximation of this probability.

```
num_angry = 0

for rep in np.arange(10000):
    # randomly choose 21000 elements from [True, False] such that
    # True has probability 0.95, False has probability 0.05
    attending = np.random.choice([True, False], 21000, p=[0.95, 0.05])
    if __(a)__:
        __(b)__

prob_angry_student = __(c)__
```

a) What goes in the **first** blank?

- ◯ `np.count_nonzero(attending) == 20001`
- ◯ `attending[20000] == False`
- ● `attending.sum() > 20000`
- ◯ `np.count_nonzero(attending) > num_angry`

b) What goes in the **second** blank?

Solution: `num_angry = num_angry+1`

c) What goes in the **third** blank?

Solution: `num_angry/10000`

Question 12 (6 points)

a) You're definitely going to Sun God 2022, but you don't want to go alone! Fortunately, you have n friends who promise to go with you. Unfortunately, your friends are somewhat flaky, and each has a probability p of actually going (independent of all others). What is the probability that you wind up going alone? Give your answer in terms of p and n.

Solution: $(1-p)^n$

b) In past Sun God festivals, sometimes artists that were part of the lineup have failed to show up! Let's say there are n artists scheduled for Sun God 2022, and each artist has a probability p of showing up (independent of all others). What is the probability that the number of artists that show up is less than n, meaning somebody no-shows? Give your answer in terms of p and n.

Solution: $1-p^n$