DATA 6
Fall 2025

Lisa Yan
Final

Your name: _____

Your student ID: _____

Your Berkeley email: _____

Your room location: _____

Student ID of the person to your left: _____

Student ID of the person to your right: _____

You have 170 minutes. There are 8 questions of varying credit. (100 points total)

Question:	HC	1	2	3	4	5	6	7	8	Total
Points:	1	12	20	18	13	13	15	8	0	100

For questions with **circular bubbles**, you may select only one choice.

- ○ Unselected option (Completely unfilled)
- ◌ Don't do this (it will be graded as incorrect)
- ● Only one selected option (completely filled)

For questions with **square checkboxes**, you may select one or more choices.

- ■ You can select
- ■ multiple squares
- ☑ (Don't do this)

Anything written outside the answer boxes or ~~crossed out~~ will not be graded. If you write multiple answers, your answer is ambiguous, or the bubble/checkbox is not entirely filled in, we will grade the worst interpretation. For coding questions with blanks, you may write at most one statement per blank, unless otherwise stated.

As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others. I will follow the rules of this exam.

Honor Code (HC): I have read and agree to the honor code above.

(1 point) Sign your name: _____

Q1 *What Would Python Do? (WWPD)* **(12 points)**

Q1.1 (2 points) Recall that `print` displays arguments to the screen but returns `None`. See the reference sheet for `bool` typecasting.

```
1 x = 3
2 x *= x
3 y = bool(print(x))
4 x += int(y)
```

Q1.1.1 After the code is run, what is printed? If Python will error, select "Error".

◯ `True` ◯ `0` ◯ `9` ◯ Error

◯ `False` ◯ `3` ◯ `None`

Q1.1.2 After the code is run, what is the value of `x`? If Python will error, select "Error".

◯ `3` ◯ `9` ◯ `12` ◯ `18` ◯ `None` ◯ Error

Q1.2 (2 points) Recall that functions can have names, too. Consider the below code:

```
1 len = sum
2 sum = min
3 arr1 = make_array(1, 2, 3, 4)
4 arr2 = make_array(3, 1)
5 print(sum(len(arr1), len(arr2)))
```

Q1.2.1 In Line 5, what does `len(arr1)` evaluate to? If Python will error, select "Error".

◯ `1` ◯ `2` ◯ `4` ◯ `10` ◯ Error

Q1.2.2 After the code is run, what is printed? If Python will error, select "Error".

◯ `1` ◯ `2` ◯ `4` ◯ `10` ◯ Error

Q1.3 - Q1.7 (8 points) Implement the function `reverse_join`, which returns the elements of the list `iterable` **in reverse order** as a single string, with each element separated by the string `sep`. For example, `reverse_join("!", ["D", "A", "T", "A", "6"])` returns `"6!A!T!A!D"`.

Hints: (1) `np.arange` can take negative steps. For example, `np.arange(10, 4, -2)` returns a size-3 array with elements 10, 8, and 6, in order.
(2) You may not need all blanks. If you do not use a blank, cross it out.

```
def reverse_join(sep, iterable):
    ret = ""     # the empty string

    for i in np.arange(_____Q1.3_____, _____Q1.4_____, -1):

        ret += _____Q1.5_____

        _____Q1.6_____

    _____Q1.7_____
    return ret
```

SID: _____

Q2 *The National Park Service* **(20 points)**

The National Park Service (NPS) is a United States government agency that manages **park units** like historic sites, wilderness and wildlife preserves, etc. Among the park units, the most well-known are **National Parks** (image right: Yosemite NP in California).

The `parks` table (Table 1) reports the number of visits, per year, to each of the current 63 NPS National Parks, from 1979 to 2024.

park_name	unit_code	region	state	year	visits
Acadia NP	ACAD	Northeast	ME	1979	2787366
Acadia NP	ACAD	Northeast	ME	1980	2779666
...	...	...	...	...	...
Yosemite NP	YOSE	Pacific West	CA	1979	2350782
...	...	...	...	...	...
Yosemite NP	YOSE	Pacific West	CA	2024	4121807
...	...	...	...	...	...

Table 1: Some rows from the `parks` table (2843 rows total).

Variables:

- `park_name`: The name of the park unit.
- `unit_code`: The four-letter park unit code.
- `region`: The park unit's designated NPS geographic region: Alaska, Intermountain, Midwest, Northeast, Pacific West, Southeast, and National Capital. National Capital has zero National Parks; all other regions have multiple National Parks.
- `state`: The two-letter state code that the park unit is located in.
- `year`: The year of the reported number of visits to the park unit.
- `visits`: The reported number of visits to the park unit for fun, vacation, exercise, school trips, etc.

Q2.1 (1 point) What is the variable type of `unit_code`?

- ◯ Discrete Numerical
- ◯ Ordinal Categorical
- ◯ Continuous Numerical
- ◯ Nominal Categorical

Q2.2 (1 point) What is the variable type of `year`?

- ◯ Discrete Numerical
- ◯ Ordinal Categorical
- ◯ Continuous Numerical
- ◯ Nominal Categorical

Q2.3 (2 points) Based on the description above, does the `parks` table report visits for every current National Park for every year from 1979 to 2024, inclusive?

- ◯ Yes
- ◯ No
- ◯ Insufficient data to make conclusion

Q2.4 (2 points) Which of the below would be best for visualizing the visits to Yosemite National Park by year?

- ○ Bar chart
- ○ Scatter plot
- ○ Box plot
- ○ Histogram
- ○ Line plot
- ○ None of the above

Q2.5 (2 points) Which of the below would be best for visualizing the 2024 visits in each state by the number of parks in that state?

- ○ Bar chart
- ○ Scatter plot
- ○ Box plot
- ○ Histogram
- ○ Line plot
- ○ None of the above

Consider Table 2 and Figure 2 below. Both report total visits over all time, disaggregated by region.

Q2.6 (3 points) Which line of code correctly uses the `parks` table to create Table 2? Select all that apply.

- ☐ `parks.group("region", sum)`
- ☐ `parks.group("region", sum).select("region", "visits sum")`
- ☐ `parks.group("region", sum).drop("year sum")`
- ○ None of the above

region	visits sum
Alaska	49977455
Intermountain	1154481399
Midwest	476006528
Northeast	251445176
Pacific West	766292058
Southeast	606267860

Table 2: Output when code is run.

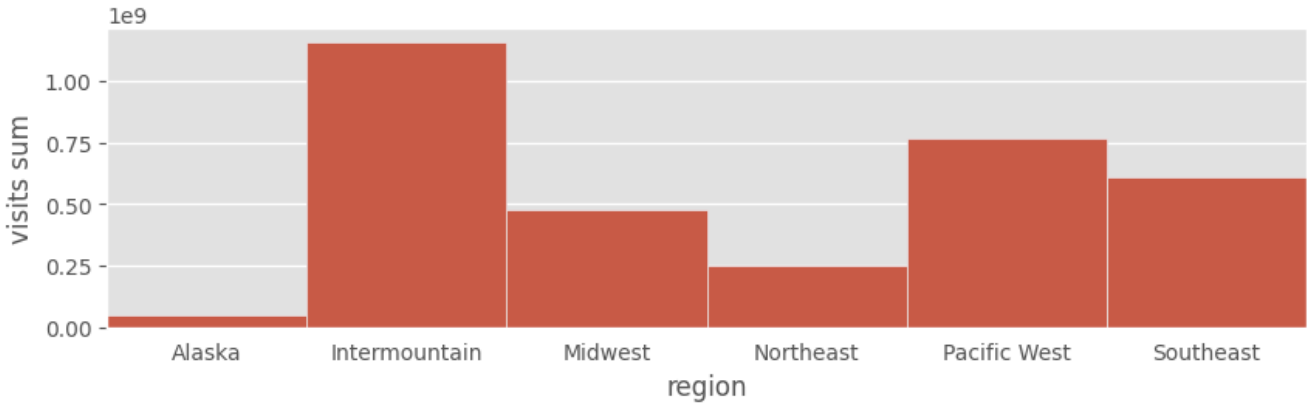


Figure 2: Visualization generated from Table 2

Q2.7 (1 point) What type of visualization is shown in Figure 2?

- ○ Bar chart
- ○ Histogram
- ○ Neither of these

Q2.8 (2 points) Consider the following statement:

On Figure 2's y-axis, the value 1e9 scales [A] by [B].

Q2.8.1 Fill in Blank [A].

- ○ values on the y-axis
- ○ visits per region

Q2.8.2 Fill in Blank [B].

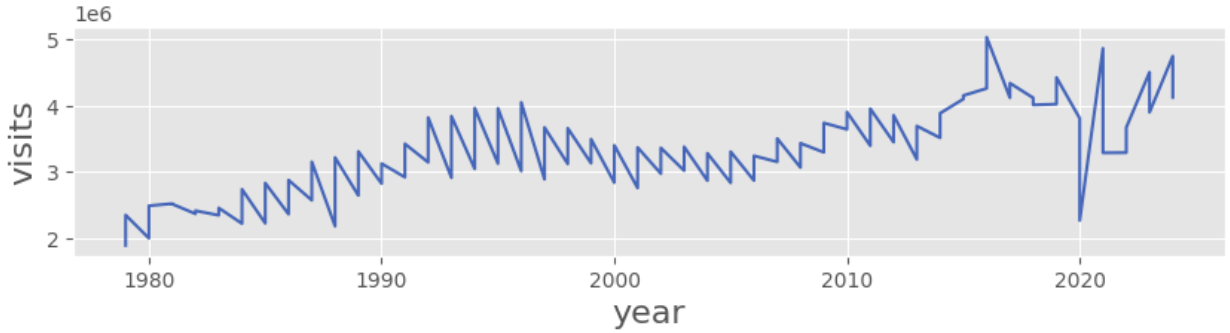
- ○ e^9, where e is the natural number
- ○ 10^9

Yellowstone NP and Yosemite NP are the only two National Parks whose park names start with "Y".

Q2.9 - Q2.10 (3 points) Complete the code below to create the table `yell_yose`, which includes a row of `parks` only if it corresponds to one of the two parks: Yellowstone NP or Yosemite NP.

```
yell_yose = parks.where(_____Q2.9_____,
    _____Q2.10_____)
```

Q2.11 (3 points) Your friend (who hasn't taken Data 6) knows that line plots are used to study chronological trends and patterns. They call `yell_yose.plot("year", "visits")`, which produces the following visualization.



Below, list one reason why your friend's plot cannot be used to compare visits to Yosemite and Yellowstone. Respond in 1-2 sentences.

Q3 *Don't Table Funding for Our Popular National Parks!* **(18 points)**

We continue analyzing NPS data to study popular times visiting National Park vists. We define the **popular year** for a given park as the year corresponding to the largest reported number of visits for that park, across all years. Assume each park has one unique popular year.

We highlight White Sands NP, a National Park recently established in 2019. The **popular year** of White Sands NP is 2021, where the park had 782469 visits. Table 3 below has relevant rows from the `parks` table, which, like before, reports the number of visits per year to each of the current 63 NPS National Parks, from 1979 to 2024.

park_name	unit_code	region	state	year	visits
...	...	...	...	...	...
White Sands NP	WHSA	Intermountain	NM	2019	608785
White Sands NP	WHSA	Intermountain	NM	2020	415383
White Sands NP	WHSA	Intermountain	NM	2021	782469
White Sands NP	WHSA	Intermountain	NM	2022	705127
White Sands NP	WHSA	Intermountain	NM	2023	729096
White Sands NP	WHSA	Intermountain	NM	2024	702236
...	...	...	...	...	...

Table 3: The relevant White Sands NP rows from the `parks` table (like before, 2843 rows total).

Q3.1 (4 points) Implement the `get_max_visits` function, which takes in a park name and returns the `visit` value for that park's popular year. For example, calling `get_max_visits("White Sands NP")` returns 782469.

```
def get_max_visits(park_name):
    visits_for_this_park = parks.where( [A] ). [B] ("visits")
    return [C]
```

Q3.1.1 Fill in Blank [A].

- ◯ `park_name, park_name`
- ◯ `"park_name", park_name`
- ◯ `park_name, "park_name"`
- ◯ `"park_name", "park_name"`

Q3.1.2 Fill in Blank [B].

- ◯ `column`
- ◯ `select`
- ◯ `columns`
- ◯ None of the above

Q3.1.3 Fill in Blank [C].

Q3.2 (3 points) Assume a correct implementation of `get_max_visits` from the previous part.
- Q3.3 Implement the `is_popular` function, which takes in a park name and an integer value and returns `True` if the value is **at least as big** as the number of visits from the park's popular year, and `False` otherwise.

For example, `is_popular("White Sands NP", 608785)` evaluates to `False`; `is_popular("White Sands NP", 782469)` evaluates to `True`; and `is_popular("White Sands NP", 999999)` evaluates to `True`.

Hint: Your code should not use any conditionals (e.g., if or if-else statements).

```
def is_popular(park_name, num_visits):

    max_visit = get_max_visits(_____Q3.2_____)

    return _____Q3.3_____
```

Q3.4 (7 points) Assume a correct implementation of `is_popular` from the previous part. Complete
- Q3.8 the code below to filter `parks` and output only the rows of corresponding to the popular years of the 63 National Parks. The resulting table (Table 4) should have the same columns as `parks`.

park_name	unit_code	region	state	year	visits
Acadia NP	ACAD	Northeast	ME	1989	5440952
...	...	...	...	...	...
White Sands NP	WHSA	Intermountain	NM	2021	782469
...	...	...	...	...	...
Yosemite NP	YOSE	Pacific West	CA	2016	5028868
...	...	...	...	...	...

Table 4: Some rows of the table output when your code is run (63 rows total).

Hint: The first blank will include commas, e.g., `arg1, arg2` passes in two arguments to `apply`. Other blanks assume no commas.

```
(
  parks
    .with_column("is_popular",

          parks.apply(_____Q3.4_____))

    ._____Q3.5_____(_____Q3.6_____, _____Q3.7_____)

    ._____Q3.8_____("is_popular")
)
```

Your friend (who hasn't taken Data 6) knows that the `group` method takes an optional argument `function` for aggregating column values. Further, they astutely (and correctly) observe that aggregating a single-value column with `max` produces that single value—even with string types.

Your friend writes the below code, which outputs a table like Table 5:

```
parks.group("park_name", max)
```

park_name	unit_code max	region max	state max	year max	visits max
Acadia NP	ACAD	Northeast	ME	…	…
…	…	…	…	…	…
White Sands NP	WHSA	Intermountain	NM	…	…
…	…	…	…	…	…
Yosemite NP	YOSE	Pacific West	CA	…	…
…	…	…	…	…	…

Table 5: Some rows of the table output when your friend's code is run (63 rows total). Certain values are intentionally omitted.

Q3.9 (4 points) Your friend believes that your approach to finding popular years is too complicated, and they argue that their approach also works. In other words, they believe that, barring column renaming and possible row reordering, Table 4 and Table 5 are identical.

Q3.9.1 Is your friend correct?

○ Yes, correct ○ No, incorrect ○ Insufficient data to make conclusion

Q3.9.2 Justify your choice above in most three sentences.

SID: _____

Q4 *NPS Tourism Boosts State Economies* **(13 points)**

The NPS preserves the ecological and historical integrity of a range of **park units**, which include National Parks, National Memorials (right: Lincoln Memorial in Washington, D.C.), National Historic Sites, and more.

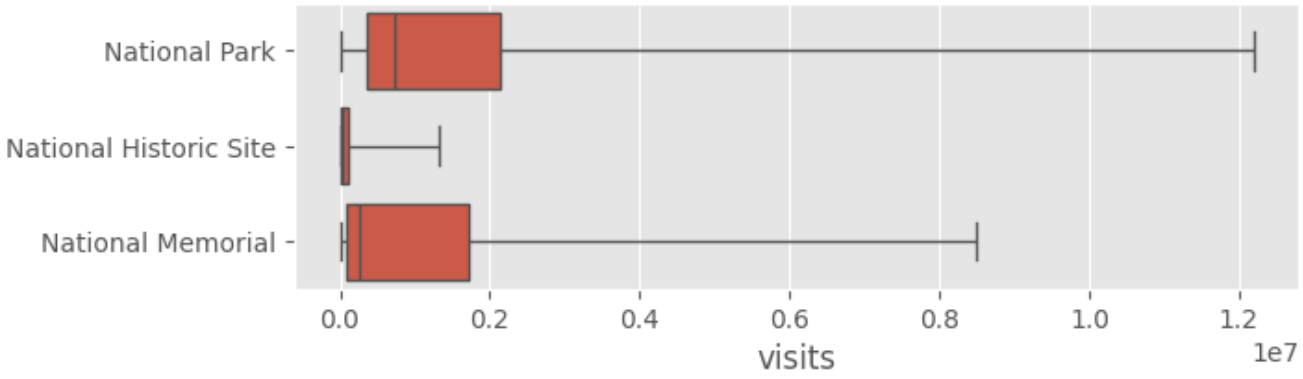
The `nps_2024` table (Table 6 below) reports the number of visits in 2024 to each of the current 398 NPS park units. It has the columns of `parks` (from before) with one additional column:

- `park_type`: The type of park unit, e.g., National Park, National Memorial, etc.

park_name	unit_code	park_type	region	state	year	visits
...	...	...	...	...	...	...
Lincoln Memorial	LINC	National Memorial	National Capital	DC	2024	8479349
...	...	...	...	...	...	...
Tuskegee Airmen NHS	TUAI	National Historic Site	Southeast	AL	2024	28474
...	...	...	...	...	...	...
Yosemite NP	YOSE	National Park	Pacific West	CA	2024	4121807
...	...	...	...	...	...	...

Table 6: Some rows from the `nps_2024` table (398 rows total).

The `nps_2024` table was used to generate the following boxplots of number of visits to park units, disaggregated by three park types:



Q4.1 (3 points) Using the boxplots above, which of the following are true statements about the park units in the `nps_2024` table? A park unit's **popularity** is its number of visits in 2024.

☐ There are fewer National Historic Sites than National Memorials.

☐ For National Memorials, the mean number of visits is greater than the median number of visits.

☐ Any park unit in the bottom 50% of National Parks is more popular than any park unit in the bottom 50% of National Memorials.

○ None of the above

We would like to analyze the **2024 tourism rate** for a park unit, which we define as the number of visits to that park unit in 2024 per 1,000 residents in that park unit's state. For example, the 2024 estimated number of residents of Alabama is 5157699; therefore, the Tuskegee Airmen NHS has a 2024 tourism rate of $28474/5157699 \times 1000 \approx 5.52$.

The below visualization (Figure 6) plots the distribution of 2024 tourism rates across all park units, separated into five bins: [0, 100), [100,200), [200, 300), [300, 400), and [400, 1200]. Use Figure 6 to answer the questions below.

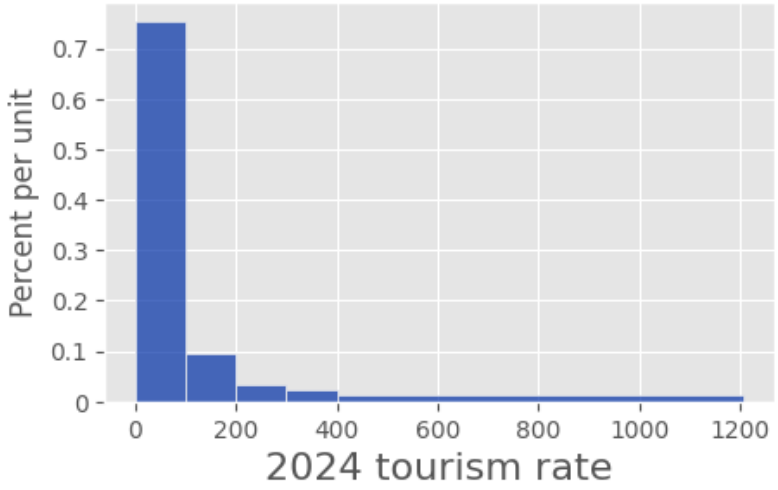


Figure 6: Distribution of 2024 tourism rates across all park units.

Q4.2 (1 point) Consider the y-axis in Figure 6. What is the "unit" in "Percent per unit"?

- ◯ 1,000 state residents
- ◯ state resident
- ◯ park unit
- ◯ 10 state residents
- ◯ visitor
- ◯ None of the above

Q4.3 (1 point) Which bin has the most park units in it?

- ◯ [0, 100)
- ◯ [200, 300)
- ◯ [400, 1200]
- ◯ [100, 200)
- ◯ [300, 400)
- ◯ None of the above

Q4.4 (1 point) Roughly how many park units have a 2024 tourism rate between 100 and 200 park unit visits per 1,000 state residents?

- ◯ 1
- ◯ 40 (fixed post-quiz)
- ◯ 400 (fixed post-quiz)
- ◯ 10
- ◯ 100
- ◯ None of the above

The table `census` (Table 7) contains 2024 estimates of the number of residents for each of the fifty states, the District of Columbia, Puerto Rico, and other U.S. Territories.

Name	Population	Abbreviation
Alabama	5157699	AL
Alaska	740133	AK
Arizona	7582384	AZ

Table 7: First 3 rows of `census`.

Q4.5 (7 points) Using `nps_2024` (Table 6) and `census` (Table 7), plot Figure 6 (from the previous part), which visualizes the distribution of 2024 tourism rates across all park units.

```
my_bins = make_array(0, 100, 200, 300, 400, 1200)

nps_tourism = nps_2024.[A]([B])

nps_tourism = nps_tourism.with_column("2024 tourism rate", [C])

nps_tourism.[D]([E], [F])
```

Q4.5.1 Fill in Blank [A].

Q4.5.2 Fill in Blank [B]. To pass in multiple arguments, include commas, e.g., `arg1, arg2`

Q4.5.3 Fill in Blank [C]. Your answer should not include commas.

Q4.5.4 Fill in Blank [D].

Q4.5.5 Fill in Blank [E]. Your answer should not include commas.

Q4.5.6 Fill in Blank [F]. Your answer should not include commas.

SID: _____

Q5 *The "Bear" Necessities* **(13 points)**

Figure 7 shows a black bear and a grizzly bear—the two species of bears native to the continential United States.

Figure 7: Left: black bear; right: grizzly bear.

We analyze an article about bears from a webpage on the Glacier National Park website.

Q5.1 (1 point) Which file format below is most approriate for storing the webpage data?

- ◯ CSV file
- ◯ JSON file
- ◯ None of these
- ◯ HTML file
- ◯ Jupyter Notebook

Q5.2 (1 point) Which Python package below is most appropriate for extracting the article content from the webpage into a Jupyter Notebook for further analysis?

- ◯ BeautifulSoup
- ◯ numpy
- ◯ None of these
- ◯ datascience
- ◯ scikit-learn

For the next part, consider **the HTML element in Lines 3-5** from the HTML snippet below:

```
1 <picture>
2   <source type="image/webp" srcset="...">
3   <img alt="Black Vs. Grizzly Bear comparison chart"
4        src="/.../BlackVsGrizz_2.jpg"
5        title="">
6 </picture>
```

Q5.3 (1 point) What is this HTML element's tag?

- ◯ `img`
- ◯ `alt`
- ◯ `src`
- ◯ `title`
- ◯ None of these

Q5.4 (1 point) How many attributes does this HTML element have?

- ◯ 0
- ◯ 1
- ◯ 2
- ◯ 3
- ◯ 4

Q5.5 (2 points) The BeautifulSoup paradigm uses parents, siblings, and children to describe the structural relationships between HTML elements.

Q5.5.1 What is this HTML element's parent? If it doesn't have one, select "None of these."

- ◯ `img`
- ◯ `picture`
- ◯ `source`
- ◯ None of these

Q5.5.2 What is this HTML element's sibling? If it doesn't have one, select "None of these."

- ◯ `img`
- ◯ `picture`
- ◯ `source`
- ◯ None of these

Accurate bear identification is key to safety of both humans and bears. The black bear and grizzly bear differ in how they react to encountering humans and other unknown situations.

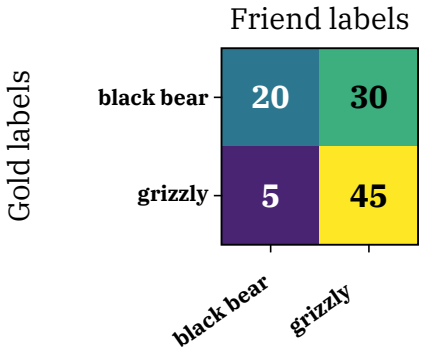
Classification Task: Given a photo of a bear, identify if the bear is a black bear or a grizzly bear.

Q5.6 (3 points) After reading over the webpage, your friend decides to try out the task above. They identify a dataset of 100 bear photos with **gold labels**, which were generated and validated by bear wildlife experts.[1]

Your friend comes back to you with 100 labels (call these "friend labels"). You would like to compare your friend's classification results to these gold labels. In this case, why might accuracy be a better performance metric than Cohen's kappa (κ)? Justify your answer in 1-2 sentences.

Q5.7 (4 points) You create a confusion matrix compares your friend's labels to the gold labels.

Use this visualization to answer the questions below. Round to the nearest percent.



Q5.7.1 What is your friend's overall accuracy (%)?

%

Q5.7.2 What is your friend's class accuracy (%) for grizzly bears?

%

Q5.7.3 What percentage of the dataset did your friend classify as grizzly bears?

%

Q5.7.4 What percentage of black bears did your friend **mis**-classify as grizzly bears?

%

[1]Wildlife researchers may tag wild bears for tracking purposes, to proactively minimize human-bear interaction. Physical data like DNA, if collected, is done so in a minimally invasive manner.

Q6 *"Bear" is the representative word of 2025 (by popular vote)* **(15 points)**

We return to our webpage article about bears. The text content on the webpage has been extracted out from the HTML and stored as a **bag of words** model via the dictionary `bear_of_words`.

Some key-value pairs in `bear_of_words` are shown on the right.

- Each key is a word that appears on the webpage.
- Each value is the number of times that word appears on the webpage.

Assume that all words are alphabetic lowercase (i.e., no numbers nor punctuation).

```
{
    'a': 16,
    'bear': 21,
    'glacier': 9,
    'honeycombs': 1,
    'national': 6,
    'the': 80,
    'ursus': 2,
    ...
}
```

Q6.1 (1 point) Which file format below is most useful for storing the dictionary `bear_of_words`?

- ◯ CSV file
- ◯ JSON file
- ◯ None of these
- ◯ HTML file
- ◯ Jupyter Notebook

Q6.2 (2 points) Which of the following is most likely an English stop word? Select all that apply.

- ☐ `'a'`
- ☐ `'are'`
- ☐ `'bear'`
- ☐ `'dna'`
- ☐ `'the'`
- ◯ None of the above

Q6.3 (3 points) This code computes `avg_freq`, the average word frequency in `bear_of_words`.

```
1 freq_sum = 0
2 for word in bear_of_words:
3     freq_sum += bear_of_words[word]
4 avg_freq = freq_sum / len(bear_of_words)
```

Consider substituting Lines 2 and 3 above for each of the for loops below. Which substitution will still correctly compute `avg_freq`? Select all that apply.

Hint: See the last page of the reference sheet for dictionary iteration techniques.

- ☐
```
for k in bear_of_words.keys():
    freq_sum += bear_of_words[k]
```
- ☐
```
for k in bear_of_words.keys():
    freq_sum += k
```
- ☐
```
for v in bear_of_words.values():
    freq_sum += bear_of_words[v]
```
- ☐
```
for v in bear_of_words.values():
    freq_sum += v
```
- ☐
```
for k, v in bear_of_words.items():
    freq_sum += bear_of_words[k, v]
```
- ◯ None of the above
- ☐
```
for k, v in bear_of_words.items():
    freq_sum += v
```

Assume that `avg_freq` is correctly assigned above, `stop_words` is assigned to a list of stop words in the English language, and all words are alphabetic lowercase.

Q6.4 (9 points) Construct the list `highlight_words` from `bear_of_words`, where each word in `highlight_words` should fulfill at least one of the following criteria:

- The word appears at least as frequently as `avg_freq` but is *not* a stop word in `stop_words`.
- The word is at least 8 letters long.

If a word satisfies both criteria, it should appear in `highlight_words` exactly once, not twice.

Instructions: Fill in the blanks below. If the blank is not needed, either select "N/A" or write "N/A".

```
highlight_words = []
for word in bear_of_words:
    [A] [A-cond]
        [A-body]
    [B] [B-cond]
        [B-body]
    [C] [C-cond]
        [C-body]
```

Q6.4.1 Blank [A) ○ `if` ○ `elif` ○ `else:` ○ N/A

Q6.4.2 Blank [A-cond]

Q6.4.3 Blank [A-body]

Q6.4.4 Blank [B] ○ `if` ○ `elif` ○ `else:` ○ N/A

Q6.4.5 Blank [B-cond]

Q6.4.6 Blank [B-body]

Q6.4.7 Blank [C] ○ `if` ○ `elif` ○ `else:` ○ N/A

Q6.4.8 Blank [C-cond]

Q6.4.9 Blank [C-body]

Q7 *Potpourri (Multiple-Choice Question Grab Bag)* **(8 points)**

Q7.1 (2 points) Which of the scenarios below describe "joining"? Select all that apply.

- ☐ Cross-classifying individuals across two or more categorical variables in the same table
- ☐ Extending one table to include the rows from another table
- ☐ Extending one table to include the columns from another table
- ☐ Stitching together a list of strings into one string
- ○ None of the above

Q7.2 (1 point) True or False: To analyze a table of data on individuals at a larger unit of analysis, group rows together.

○ True ○ False

Q7.3 (1 point) True or False: A confounding factor describes a variable that has a very high standard deviation, i.e., two individuals can report very different values for that variable.

○ True ○ False

Q7.4 (4 points) Consider the case studies across this course. Which of the following statements about reliability are true? Select all that apply.

- ☐ [Project] In the 1973 UC Berkeley Admissions study, women exhibited lower reliability of getting admitted into more competitive graduate programs.
- ☐ [Homework] One risk to reliability of the Body Mass Index (BMI) is that the BMI consistently classifies athletes on the San Francisco 49ers football team as obese, even though by many standards these athletes are considered healthy.
- ☐ [Project] In the computational social science task study, LLM-assisted AI coders can exhibit lower reliability compared to human coders.
- ☐ [Reading] If human study subjects do not consent to participate in a research study, then the research data will be less reliable.
- ○ None of the above

Q8 *Just for fun!* **(0 points)**

Q8.1 Draw something fun, or write a message for the staff! Or leave this blank!

Thanks for a great semester!

—*The Data 6 Fall 2025 Course Staff*