

CS 220 - Fall 2022
Instructors: Meenakshi Syamkumar, Mike Doescher, and Gurmail Singh

Final — 10%

(Last) Surname: _____ (First) Given name: _____

NetID (email): _____ @wisc.edu

Fill in these fields (left to right) on the scantron form (use #2 pencil):

1. LAST NAME (surname) and FIRST NAME (given name), fill in bubbles
2. IDENTIFICATION NUMBER is your Campus ID number, fill in bubbles
3. Under *ABC* of SPECIAL CODES, write your lecture number, fill in bubbles:
 - 001 - MWF 11:00am (Meena)
 - 002 - MWF 1:20pm (Meena)
 - 003 - MWF 8:50am (Gurmail)
 - 004 - MWF 9:55am (Mike)
 - 005 - MWF 3:30pm (Mike)
 - 006 - MWF 1:20pm (Gurmail)
4. Under **F** of SPECIAL CODES, write *A* and fill in bubble **6**

If you miss step 4 above (or do it wrong), the system may not grade you against the correct answer key, and your grade will be no better than if you were to randomly guess on each question. So don't forget!

You may only reference your note sheet. You may not use books, your neighbors, calculators, or other electronic devices during this exam. Please place your student ID face up on your desk. Turn off and put away portable electronics (including smart watches) now.

Use a #2 pencil to mark all answers. When you're done, please hand in the exam and note sheet and your filled-in scantron form. The note sheet will not be returned.

Smartphones (Pandas)

Consider the following code:

```
import pandas as pd
smartphones_dict = {
    "Model": ["iPhone 13", "iPhone 13 Pro Max", "iPhone 12", \
    "Galaxy S22 Ultra", "Galaxy A13"],
    "Manufacturer": ["Apple", "Apple", "Apple", "Samsung", "Samsung"],
    "Release Year": [2021, 2021, 2020, 2022, 2022],
    "Cost ($)": [799, 1099, 599, 955, 149],
    "Market Share (in %)": [5.5, 3.4, 1.6, 1.4, 1.4]
}
smartphones = pd.DataFrame(smartphones_dict)
```

1. What data type does `smartphones["Manufacturer"].value_counts()` return?
A. int B. dict C. str **D. Series** E. DataFrame
2. What will the inner data structures in `smartphones_dict` represent in the `smartphones` DataFrame?
A. rows **B. columns** C. DataFrames D. lists
3. How can we get the name of the most expensive Model?
A. `smartphones["Cost ($)"].max()["Model"]`
B. `smartphones["Model"]["Cost ($)"].max()`
C. `smartphones[smartphones["Cost ($)"] == smartphones["Cost ($)"].max()]["Model"]`
D. `smartphones["Cost ($)"] == smartphones["Cost ($)"].max()["Model"]`
4. How can we get the name of the least expensive Manufacturer for each Release Year?
A. `smartphones["Cost ($)"].value_counts().min()[["Manufacturer", "Release Year"]]`
B. `smartphones[["Manufacturer", "Release Year", "Cost ($)"]]["Cost ($)"].value_counts().min()`
C. `smartphones[smartphones["Cost ($)"] == smartphones["Cost ($)"].min()][["Manufacturer", "Release Year"]]`
D. `smartphones["Release Year"].groupby("Release Year")[["Manufacturer", "Cost ($)"]].min()`
E. `smartphones[["Manufacturer", "Release Year", "Cost ($)"]].groupby("Release Year").min()`

-
5. Suppose the shop selling these smartphones wants to provide a Christmas season discount of 10% to all of the phones. Which of the following will allow us to add a new column called `Sale Cost ($)` (cost after the 10% discount) to the `smartphones` DataFrame?
- A. `smartphones["Sale Cost ($)"] = smartphones["Cost ($)"] * 10 / 100`
 - B. `smartphones["Sale Cost ($)"] = smartphones["Cost ($)"].apply(lambda c: c - (c * 10 / 100))`
 - C. `smartphones["Sale Cost ($)"] = smartphones["Cost ($)"].apply(c * 10 / 100)`
 - D. `smartphones["Cost ($)"] - (smartphones["Cost ($)"] * 10 / 100)`
6. Which of the following expressions will **NOT** help us extract iPhone 13's Market Share (in %)? Note: ignore hard coding assumptions to answer this question.
- A. `smartphones.iloc[0]["Market Share (in %)"]`
 - B. `smartphones.loc[0]["Market Share (in %)"]`
 - C. `smartphones.iloc[0].loc["Market Share (in %)"]`
 - D. `smartphones.loc[0].loc[4]`
 - E. `smartphones.loc[0].loc["Market Share (in %)"]`
7. Find the average price of phones sold by Apple.
- A. `mean(smartphones[smartphones["Manufacturer"] == "Apple"]["Cost ($)"])`
 - B. `smartphones[smartphones["Manufacturer"] == "Apple"]["Cost ($)"].mean()`
 - C. `(smartphones["Manufacturer"] == "Apple")["Cost ($)"].mean()`
 - D. `smartphones[smartphones["Manufacturer"] == "Apple"]["Cost ($)"].sum() / len(smartphones)`
8. How can we sort the `smartphones` DataFrame based on decreasing order of "Release Year"?
- A. `smartphones.sort_values(by="Release Year", ascending=False)`
 - B. `smartphones.sort_values(by="Release Year", descending=True)`
 - C. `smartphones.sort_values(by="Release Year", reverse=True)`
 - D. `smartphones.sort_index(by="Release Year", reverse=True)`

FIFA (Database)

Consider the following code, and the corresponding output:

```
import sqlite3
import pandas as pd

conn = sqlite3.connect("fifa.db")
fifa_scores = pd.read_sql("SELECT * from fifa_top_scorers", conn)
fifa_scores
```

	player_name	country	goals_scored	year
0	Alex Morgan	USA	6	2019
1	Harry Kane	England	6	2018
2	James Rodriguez	Colombia	6	2014
3	Thomas Muller	Germany	5	2010
4	David Villa	Spain	5	2010
5	Miroslav Klose	Germany	5	2006

9. Which SQL clause enables us to narrow down columns of interest?
- A. SELECT B. FROM C. WHERE D. GROUP BY E. LIMIT
10. How many rows will the following SQL query return?
- ```
SELECT COUNT(player_name) FROM fifa_top_scorers GROUP BY goals_scored LIMIT 4
```
- A. 0   B. 1   C. 2   D. 3   E. 4
11. Sort by descending order of **year**, the rows of players whose **goals\_scored** is more than 5.
- A. SELECT FROM fifa\_top\_scorers WHERE goals\_scored > 5 ORDER BY year  
DESC
- B. SELECT \* FROM fifa\_top\_scorers WHERE goals\_scored > 5 ORDER BY DESC  
year
- C. SELECT \* FROM fifa\_top\_scorers WHERE goals\_scored > 5 ORDER BY year  
DSC
- D. SELECT \* FROM fifa\_top\_scorers WHERE goals\_scored > 5 ORDER BY year  
DESC
12. Which **player\_name** will be **excluded** when we execute the below SQL query?
- ```
SELECT player_name FROM fifa_top_scorers WHERE year >= 2010 AND year < 2020
```
- A. Alex Morgan B. Thomas Muller C. David Villa D. Miroslav Klose

13. Which SQL query produces a similar output to the following pandas expression?

```
fifa_scores[fifa_scores["country"] == "Germany"]["goals_scored"].mean()
```

- A. `SELECT goals_scored FROM fifa_top_scorers WHERE country = "Germany" ORDER BY goals_scored DESC`
- B. `SELECT AVG(goals_scored) FROM fifa_top_scorers WHERE country = "Germany"`
- C. `SELECT SUM(goals_scored)/MAX(goals_scored) FROM fifa_top_scorers WHERE country = "Germany"`
- D. `SELECT AVG(goals_scored) FROM fifa_top_scorers GROUP BY country`

14. Which pandas expression will produce the closest results to the following SQL query?

```
SELECT country, MAX(goals_scored) FROM fifa_top_scorers GROUP BY country
```

- A. `fifa_scores["country"].value_counts().max()`
- B. `fifa_scores["goals_scored"].value_counts().max()`
- C. `fifa_scores[fifa_scores["goals_scored"] == fifa_scores["goals_scored"].max()]`
- D. `fifa_scores[["country", "goals_scored"]].groupby("country").max()`

15. Which SQL query answers the following question: Which year(s) had more than 6 goals_scored in total?

- A. `SELECT year, SUM(goals_scored) AS total_goals FROM fifa_top_scorers WHERE HAVING total_goals > 6 GROUP BY year`
- B. `SELECT year, SUM(goals_scored) AS total_goals FROM fifa_top_scorers GROUP BY year HAVING total_goals > 6`
- C. `SELECT year, SUM(goals_scored) AS total_goals FROM fifa_top_scorers GROUP BY year ORDER BY total_goals DESC`
- D. `SELECT year, SUM(goals_scored) AS total_goals FROM fifa_top_scorers GROUP BY total_goals LIMIT 1`

Grades (Plotting)

The top 5 rows within `majors_df` DataFrame are displayed below. Assume that `majors_df` DataFrame contains many rows.

	GPA	Time studied (hours)	Major
0	3.64	16.45	Data Science
1	3.20	13.95	Biology
2	2.95	5.97	Biology
3	3.16	9.22	Engineering
4	2.50	4.42	Economics
...

16. Which of the following options will generate a scatter plot having “Time studied (hours)” on the x-axis and “GPA” on the y-axis for all students pursuing the “Data Science” major?

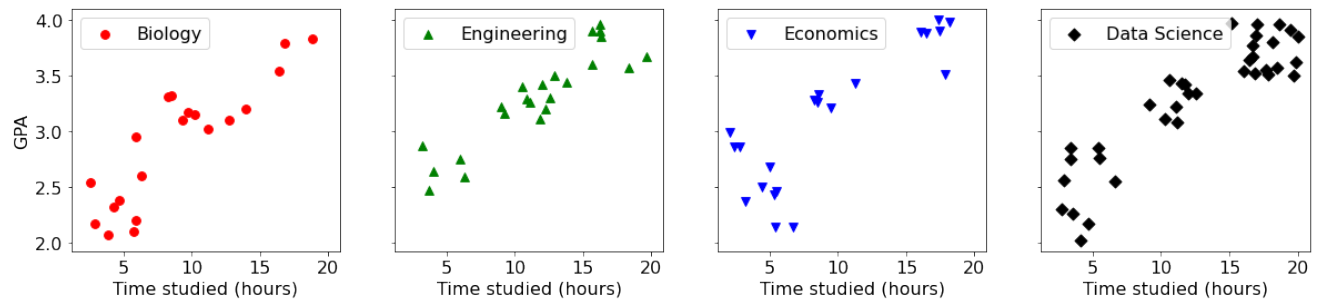
- A. `(majors_df["Major"] == "Data Science").set_index("GPA")["Time studied (hours)"].plot.scatter()`
- B. `majors_df[majors_df["Major"] == "Data Science"].plot.scatter(x="Time studied (hours)", y="GPA")`
- C. `majors_df[majors_df["Major"] == "Data Science"].plot.scatter(x="GPA", y="Time studied (hours)")`
- D. `(majors_df["Major"] == "Data Science").set_index("Time studied (hours)")["GPA"].plot.scatter()`

17. Refer to the provided sample rows in `majors_df` for answering this question. Which of the following options should replace Line 1 in the code snippet below to create the **best line plot** between “Time studied (hours)” (x-axis) and “GPA” (y-axis)?

```
ax = majors_df.plot.line(x = "Time studied (hours)", y = "GPA") # Line 1
ax.set_xlabel("Time studied (hours)")                          # Line 2
ax.set_ylabel("GPA")                                           # Line 3
```

- A. `ax = majors_df.sort_values(by="Time studied (hours)").plot.line(x="Time studied (hours)", y="GPA")`
- B. `ax = majors_df.sort_values(by="GPA").plot.line(x="Time studied (hours)", y="GPA")`
- C. `ax = majors_df.sort_index().plot.line(x="Time studied (hours)", y="GPA")`
- D. `ax = majors_df.set_index("Time studied (hours)").plot.line(y="GPA")`

18. Assume that `majors_df` DataFrame column `Major` contains 4 unique values. In the below code snippet, what should replace the `???` (argument passed to the `ax` parameter) to produce the below scatter plots?



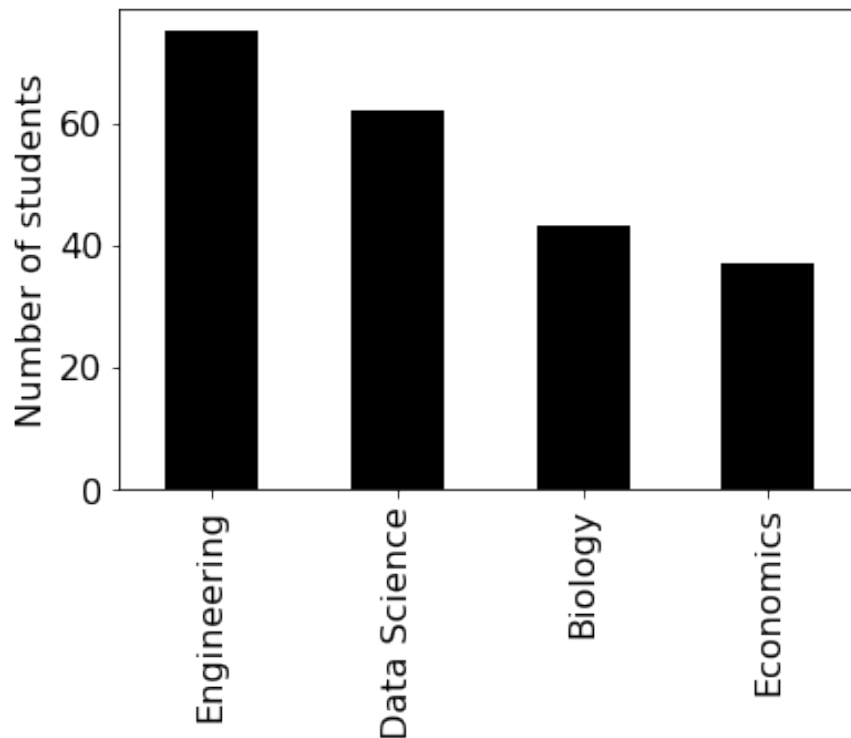
```
from matplotlib import pyplot as plt

majors = list(set(majors_df["Major"]))
fig, axes = plt.subplots(ncols=4, sharex=True, sharey=True)
plot_area = None

for i in range(len(majors)):
    major = majors[i]
    major_data = majors_df[majors_df["Major"] == major]
    # assume that appropriate arguments are passed to other relevant parameters
    plot_area = major_data.plot.scatter(x="Time studied (hours)", y="GPA", \
                                         ax=???, ...)
```

- A. `plot_area` B. `axes` C. `axes[i]` D. `range(i)` E. None
19. When we invoke `majors_df["GPA"].plot.bar()`, which of the following represents the y-axis?
- A. GPA values
 - B. row indices
 - C. column names
 - D. row integer positions
 - E. column integer positions

-
20. Which of the following options would create the below bar plot representing the number of students in each Major having a GPA above 3.8?



- A. `ax = majors_df[majors_df["GPA"] > 3.8].value_counts().plot.bar()`
B. `ax = majors_df[majors_df["GPA"] > 3.8].plot.bar()`
C. `ax = majors_df[majors_df["GPA"] > 3.8]["Major"].plot.bar()`
D. `ax = majors_df[majors_df["GPA"] > 3.8]["Major"].value_counts().plot.bar()`
21. Consider the bar plot from the previous question. If we want to change the angle of the x-axis tick labels to 45 degrees, what additional keyword argument should we pass to the below method call?

```
# assume the variable 'indices' stores a reference  
# to the list of x-axis tick labels  
ax.set_xticklabels(indices, ???)
```

- A. `flip=45` B. `move=45` C. `turn=45` D. `tilt=45` E. **`rotation=45`**

Web

22. Suppose a GET request to the URL `https://cs220.cs.wisc.edu/christmas_list.html` produces an HTTP response with the status code 404, what gets printed after executing the below code snippet?

```
import requests
from bs4 import BeautifulSoup

def download_christmas_list(url):
    try:
        r = requests.get(url)
        status = r.status_code
    except requests.HTTPError:
        print("Santa could not find your list!")
        return "Error"

download_christmas_list("https://cs220.cs.wisc.edu/christmas_list.html")
```

- A. Santa could not find your list!
 - B. Santa could not find your list!
Error
 - C. 404
 - D. Program crashes
 - E. Program doesn't print anything**
23. In the code snippet below, what should replace ??? so that the contents of the webpage are written to the file `christmas_list.html`? Assume that the HTTP GET request produces a response with the status code 200.

```
import requests

r = requests.get("https://santaclaus.com/christmas_list.html")
assert r.status_code == 200
with open("christmas_list.html", "w") as f:
    f.write(???)
```

- A. `r.text` B. `r.url` C. `r.string` D. `r.json`

Assume the following is the content of the file `christmas_list.html`:

```
<h1>A Student's Christmas List 2022</h1>
<ol>
  <li>Gingerbread cookies</li>
  <li>Action Figures</li>
  <li>A pony</li>
  <li>
    Good grade in:<a href="https://cs220.cs.wisc.edu/f22/schedule.html">CS 220</a>
  </li>
</ol>
```

For the next two questions, assume that we have executed the below code snippet:

```
from bs4 import BeautifulSoup

f = open("christmas_list.html")
html = f.read()
f.close()

doc = BeautifulSoup(html, "html.parser")
ol = doc.find("ol")
li = ol.find_all("li")
a = ol.find("a")
```

24. What is the output of `li[3].get_text()`?
- A. None
 - B. A pony
 - C. Good grade in:CS 220
 - D. Good grade in:CS 220
 - E. TypeError
25. Which of the following will enable us to retrieve the URL found in `christmas_list.html`?
- A. `a`
 - B. `a.attrs[0]`
 - C. `a.attrs["href"]`
 - D. `a.children`
 - E. `a.find_attributes()`

Assume that a GET request to `https://santaclaus.com/christmas_checklist.json` produces a successful response (200 status code) with the below content:

```
{"Snowmen":3, "Reindeer":9, "Pine Trees":20, "Snowflakes":2000}
```

Assume that the below code snippet is executed:

```
import requests
import json
```

```
r = requests.get("https://santaclaus.com/christmas_checklist.json")
r.raise_for_status()
```

26. Which of the following will enable us to read the JSON content into a Python data structure?
- A. `json.loads(r)`
 - B. `r.text`
 - C. `r.json()`
 - D. `r.text.json()`
27. What is the output of `r.text.split(", ")[1][-1]`?
- A. `'9'` B. `'"Reindeer":9'` C. `'"Snowflakes":2000}'` D. `'2000'`

Review

The `cell` and `cell_mult` functions can be used to format the population data in the data structures below. Consider the below code for the following 3 questions.

```
header = ["state", "year", "percent_urban", "pop_rural", "pop_urban"]
data = [
    ["Wisconsin", "2015", "60", "150", "200"],
    ["Idaho", "2016", "30", "200", "90"],
    ["Wisconsin", "2017", "67", "150", "300"]
]

def cell(row_idx, col_name):
    col_idx = header.index(col_name)
    val = data[row_idx][col_idx]
    if col_name ??? ["year", "percent_urban"]:
        return int(val)
    elif col_name ??? ["pop_rural", "pop_urban"]:
        return int(val) * 10**3
    else:
        return val

def cell_mult(row_idx, col_name, multiplier = 2):
    value = cell(row_idx, col_name)
    return value * multiplier
```

28. Which of the following operators should replace ??? in the `cell` function definition?
- A. `==` B. `!=` C. `for` D. `in` E. `len`
29. Which function call **will NOT** return 60?
- A. `cell_mult(1, "percent_urban", 2)`
B. `cell_mult(col_name="percent_urban", multiplier=2, row_idx=1)`
C. `cell_mult(row_idx=1, "percent_urban", 2)`
D. `cell_mult(1, "percent_urban")`
E. `cell_mult(1, col_name="percent_urban")`

30. Which function call will print 2015?

- A. `print(cell(0, "year"))`
- B. `print(cell(1, "year"))`
- C. `print(cell("Wisconsin", "year"))`
- D. `print(cell(0, 1))`
- E. Both A and C

31. What will be printed by the following code snippet?

```
subjects = ["Math", "History", "Chemistry", "Computer Science"]
print(sorted(subjects, key=lambda w: -len(w)))
```

- A. ["Chemistry", "Computer Science", "History", "Math"]
- B. ["Math", "History", "Chemistry", "Computer Science"]
- C. ["Math", "History", "Computer Science", "Chemistry"]
- D. ["Math", "Chemistry", "History", "Computer Science"]
- E. ["Computer Science", "Chemistry", "History", "Math"]

32. What will be printed after the code executes?

```
import copy

groceries = {
    "Apple": [2.5, 1],
    "Banana": 3,
    "Cherry": 2
}
groceries["Cherry"] = 4
groceries_2 = copy.copy(groceries)
groceries_2["Banana"] = 5
groceries_2["Apple"][0] = 0.5
print(groceries)
```

- A. {"Apple": [0.5, 1], "Banana": 3, "Cherry": 4}
- B. {"Apple": [0.5, 1], "Banana": 5, "Cherry": 4}
- C. {"Apple": [2.5, 1], "Banana": 3, "Cherry": 2}
- D. {"Apple": [2.5, 1], "Banana": 3, "Cherry": 4}
- E. {"Apple": [2.5, 1], "Banana": 5, "Cherry": 4}

-
33. Consider the below definition for the function `combo`. Which of the following function calls would print out `False`?

```
def combo(a, b, c=2):  
    return a == 2 or b == 2 or c == 0
```

- A. `print(combo(1, 2, 0))`
- B. `print(combo(2, 1, 0))`
- C. `print(combo(2, 2, 0))`
- D. `print(combo(3, 1, 0))`
- E. `print(combo(3, 1, 1))`

34. What is the output after the following code executes?

```
def mystery(n):  
    if n == 0:  
        print("a", end="")  
        n = 10  
        return None  
    elif n > 0:  
        print("b", end="")  
        mystery(n-1)  
        return None  
    if n == 1:  
        print("c", end="")  
        return None  
    else:  
        print("d", end="")  
        return None
```

`mystery(3)`

- A. `bbbca` B. `bbba` C. `bbbc` D. `bdbdbca` E. Infinite recursion

-
35. Which of the following options will enable us to generate a new dictionary mapping student names to their average quiz score based on the data from `quiz_scores`?

```
quiz_scores = {  
    "Rudolph": [59, 80, 70],  
    "Dasher": [59, 84, 88],  
    "Dancer": [67, 80, 79],  
    "Prancer": [71, 68, 70],  
    "Vixen": [86, 90, 87]  
}
```

- A. `[(s, sum(quiz_scores[s]) / len(quiz_scores[s])) for s in quiz_scores.keys()]`
- B. `{(s, sum(scores) / len(scores)) for scores in quiz_scores.values()}`
- C. `{s:sum(s) / len(s) for s in quiz_scores}`
- D. `{s:sum(scores) / len(scores) for (s, scores) in quiz_scores.items()}`

(Blank Page)

(Blank Page: you may tear this off for scratch work, but hand it in with your exam)

(Blank Page: you may tear this off for scratch work, but hand it in with your exam)