CS 220 - Spring 2023
Instructors: Mike Doescher and Gurmail Singh

Final Exam — 12%

(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 08:50 AM (Mike)
   - 002 - MWF 11:00 AM (Mike)
   - 003 - MWF 01:20 PM (Gurmail)
   - 004 - MWF 03:30 PM (Gurmail)
4. Under *F* of SPECIAL CODES, write **D** and fill in bubble **9**

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!

Many of the problems in this exam are related to the course projects, but some questions assume the availability of slightly different functions (e.g., for accessing the data). We won't have any trick questions where we call a function that doesn't exist, and you need to notice. Thus, if you see a call to a function we haven't explicitly defined in the problem, assume the function was properly implemented (perhaps immediately before the code snippet we DO show) and is available to you.

You may only reference your note sheet. You may not use books, your neighbors, calculators, or other electronic devices on 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 this exam and your note sheet in addition to your filled-in scantron.

Web

1. Suppose a GET request to the URL `https://example.com/api/v1/users` produces a HTTP response with the status code 500. Status code 500 is a ConnectionError. What is printed after executing the below code snippet?

```python
import requests
def get_users(url):
    try:
        r = requests.get(url)
        status_code = r.status_code
    except requests.HTTPError:
        print("An error occurred: ", e)
        return "Error"
    if status_code == 200:
        print("Successfully retrieved user data.")
        return r.json()
    else:
        print("Failed to retrieve user data.")
        return None

user_data = get_users("https://example.com/api/v1/users")
```

    A. `An error occurred:  500`

    B. `Successfully retrieved user data.`

    C. `Failed to retrieve user data.`

    D. `An error occurred:  ConnectionError`

    E. `Program does not print anything.`

For the following questions, assume that we have downloaded an HTML file named `bday_list.html` that contains the following:

```
<h1>My Birthday List</h1>
<ul>
    <li>
        <a href="https://www.amazon.com/dp/B08J68VKR6">Apple AirPods Pro</a>
    </li>
    <li>
        <a href="https://www.bestbuy.com/6426149">Sony PlayStation 5 Console</a>
    </li>
    <li>
        <a href="https://www.amazon.com/dp/B07CXG6C9W">Amazon Kindle Paperwhite</a>
    </li>
</ul>
```

2. Consider the following HTML?

```
<h1>helloworld.html</h1> <a href="dragonworld.com">See exciting
world!</a>  <ul><li>Item1</li><li>Item2</li></ul>
```

Which component of the anchor element can be used to access "dragonworld.com"? Assume variable `anchor stores the associated element.`

    A. `anchor.attrs`

    B. `anchor.children`

    C. `anchor.get_text()`

    D. `anchor.find("href")`

    E. `anchor.get_link()`

3. Continuing from the code snippet we created earlier, what will be printed out on the screen after this code snippet?

```
for child in ul.children:
  print(child.get_text())
```

    A. `Apple AirPods Pro`
       `Sony PlayStation 5 Console`
       `Amazon Kindle Paperwhite`

    B. `li`
       `li`
       `li`

    C. `https://www.amazon.com/dp/B08J68VKR6`
       `https://www.bestbuy.com/6426149`
       `https://www.amazon.com/dp/B07CXG6C9W`

    D. `TypeError`

TV Shows (Database)

Consider the following code, and the corresponding output:

```
import sqlite3
import pandas as pd

conn = sqlite3.connect("tv_shows.db")
tv_shows = pd.read_sql("SELECT * FROM tv_shows", conn)
tv_shows
```

	name	type	language	premiered
0	Hellsing Ultimate	Animation	Japanese	2006
1	Californication	Scripted	English	2007
3	Long Shadow	Documentary	English	2014
2	Transporter: The Series	Scripted	English	2013
4	Berserk	Animation	Japanese	1997
5	The Colbert Report	Talk Show	English	2005
6	Town of the Living Dead	Reality	English	2014

4. What should be added to the following query to return TV shows that premiered in 2013?

```
where_clause = ...
query = "SELECT * FROM tv_shows WHERE " + where_clause
premiered_year_2013 = pd.read_sql(query, conn)
```

   A. `"year = 2013"`

   B. `"premiered = 2013"`

   C. `"premiered(year) = 2013"`

   D. `"premiered EQUALS 2013"`

   E. `None of the above`

5. Which query will produce a DataFrame containing all rows sorted in alphabetical order by name?

   A. `SELECT * FROM tv_shows WHERE type = 'Scripted' ORDER BY name ASC`

   B. `SELECT name FROM tv_shows WHERE type = 'Scripted' SORT BY name`

   C. `SELECT * FROM tv_shows WHERE type = 'Scripted' ORDER BY name DESC`

   D. `SELECT ROWS FROM tv_shows WHERE type = 'Scripted' SORT BY name`

   E. `A and D`

6. Given the following query, how many rows will appear in the output?

```
query = """
SELECT language, COUNT(*) AS num_shows
FROM tv_shows
GROUP BY language
HAVING num_shows > 2
ORDER BY 'num_shows' DESC
"""
output = pd.read_sql(query, conn)
```

   A. 0   B. 1   C. 2   D. 5   E. 7

7. What is the type of the variable `reality_shows` in the following code?

```
query = """
SELECT *
FROM tv_shows
WHERE type = 'Reality'
"""
reality_shows = pd.read_sql(query, conn)["name"]
```

   A. `DataFrame`   B. `list`   C. `Series`   D. `set`   E. `string`

8. Which SQL query can be used to identify the English language scripted show that had the earliest premiere?

   A. `SELECT * FROM tv_shows WHERE language = 'English' AND type = 'Scripted' ORDER BY premiered ASC LIMIT 1`

   B. `SELECT * FROM tv_shows WHERE language = 'English' & type = 'Scripted' ORDER BY premiered ASC`

   C. `SELECT * FROM tv_shows WHERE language = 'English' AND type = 'Scripted' ORDER BY premiered DESC LIMIT 1`

   D. `SELECT * FROM tv_shows WHERE language = 'English' & type = 'Scripted' ORDER BY premiered ASC LIMIT 1`

9. Which of the following statements best describes the result of the code below?

```
mystery_db = pd.read_sql("""
SELECT premiered, COUNT(*) AS shows
FROM tv_shows
GROUP BY premiered
HAVING shows > 3
""", conn)

len(mystery_db)
```

      A. The number of shows that premiered in each year.

      B. The number of years which saw the premier of more than three shows.

      C. The length of a database containing the most popular show from each year.

      D. The number of years in the dataset.

      E. The number of show types in the dataset.

10. What should replace the ... in the code below to produce a DataFrame containing only non-English shows?

```
non_english = pd.read_sql("""
SELECT *
FROM tv_shows
WHERE ...
""", conn)
```

      A. language = "English"

      B. "English" NOT IN language

      C. language NOT "English"

      D. language != "English"

      E. none of the above

Plotting

11. Which of the following set the size of a figure to the tuple `figsize`

    A. `df.plot.line().set_figsize(figsize)`

    B. `df.plot.line(figsize=figsize)`

    C. `df.plot.line(set_size=figsize)`

    D. `df.plot.line().set_size(figsize)`

12. Using the DataFrame `df`, how can we graph a blue colored line plot with column A as the x-axis and B as the y-axis?

```
df = pd.DataFrame({"A":[1,2,3,4], "B":[2,7,5,8], "C":[3,6,9,12]})
```

    A. `df.plot(x=df.A, y=df.B, color="blue", kind="line")`

    B. `df.plot.line(x="A", y="B", color="blue")`

    C. `df.line_plot(x="A", y="B", color="blue")`

    D. `df[["A","B"]].plot(color = "blue")`
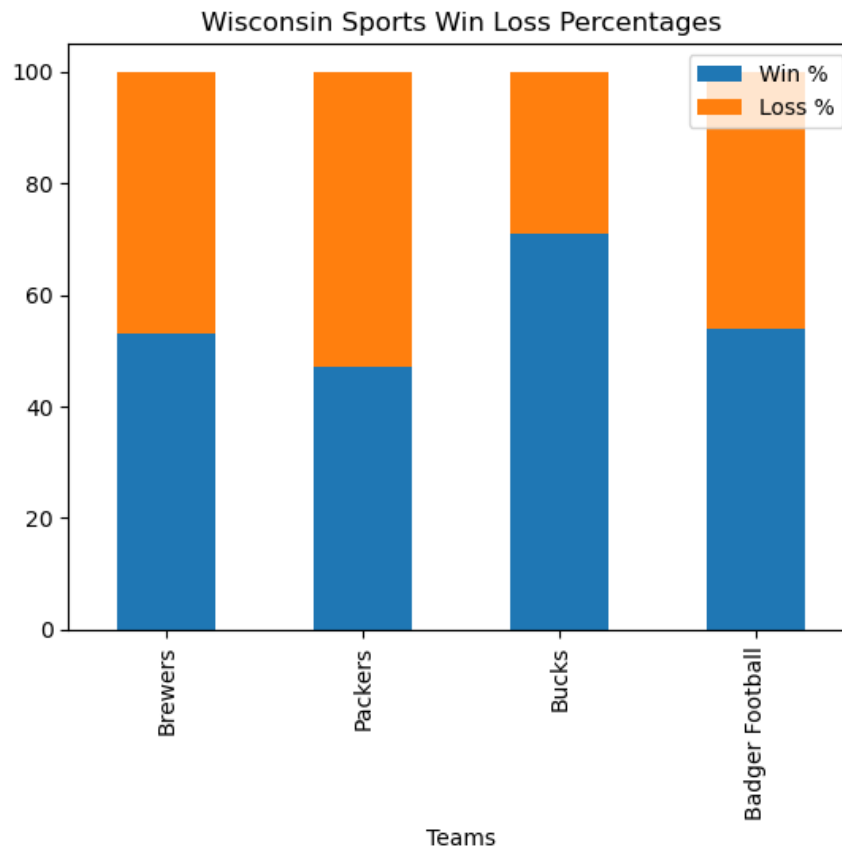
    E. `plt.plot_line(df["A"], df["B"], color = "blue")`

13. What type of graph will be displayed after executing the code below?

```
df = pd.DataFrame({"A":[1,2,3,4], "B":[2,7,5,8], "C":[3,6,9,12]})
plot = df.plot(x="B", y="C", kind="scatter")
```

    A. A scatter plot with blue dots

    B. A line plot with blue dots

    C. A scatter plot with C as the x-axis

    D. A scatter plot with B as the y-axis

14. Given the following DataFrame, which line of code will produce the following graph?

```
df = pd.DataFrame({
        "Teams": ["Brewers", "Packers", "Bucks", "Badger Football"],
        "Win %": [53, 47, 71, 54],
        "Loss %": [47, 53, 29, 46],})
```


A. `df.plot(x = "Teams", type = "bar", title= "Wisconsin Sports Win Loss Percentages")`

B. `df.plot.bar(x = "Teams", stacked = True, title= "Wisconsin Sports Win Loss Percentages")`

C. `plt.bar(df["Teams"], df["Win %", "Loss %"], color = ["blue", "orange"])`

D. `plt.bar(df["Teams"], df["Loss %"], color = "blue") + plt.bar(df["Teams"], df["Win %"], color = "orange")`

15. How can you create a scatter plot that uses Avg GPA as the x-axis and students as the y-axis?

```
df = pd.DataFrame({
        "Majors": ["Comp Sci", "Data Science", "Journalism"],
        "Students": [34, 61, 43],
        "Double Major Students": [9, 18, 13],
        "Avg GPA": [3.79, 3.60, 3.88],})
```

   A. `df.plot.scatter(x="Avg GPA", y = "Students")`

   B. `df.plot.scatter(x="Avg GPA", y= "Double Major Students")`

   C. `plt.scatter(x="Avg GPA", y= "Students")`

   D. `plt.scatter(x=df["Avg GPA"], y=df["Students"])`

16. Which of the following is true regarding the code below?

```
df.plot.scatter(x = "weight", y = "price", s = df["height"],
                color = "r", marker = "2")
```

   A. `The argument marker defines the shape of each point in the plot`

   B. `Points along the x-axis correspond to values from the weight column of the DataFrame df`

   C. `Points along the y-axis correspond to values from the price column of the DataFrame df`

   D. `The argument color defines what color we want to set for the scatter points in the plot`

   E. `All of the above`

Review

17. What will be the output of the following code?

```
str_var = "CS220 is so cool!"
ind_list = [str_var.find("2"), str_var.find("C"), str_var.lower().find("s")]
print(sorted(ind_list)[-1])
```

A. 0   B. 1   C. 2   D. 7   E. `TypeError`

18. Fill in the following three ... respectively to print the name along with the sum of values in each list.

```
sample_list = [["Alex", 2, 6], ["Bob", 1, 4, 5], ["Cat", 3, 2, 2, 1]]

for element in sample_list:
    total = 0
    for x in range(len(...)):
        if x == 0:

            ...
        else:
            total += ...
    print(element[0], total)
```

Note that the expected output is the following:

```
Alex 8
Bob 10
Cat 8
```

A. `element, break, x`

B. `element, break, element[x]`

C. `element, continue, element[x]`

D. `sample_list[element], continue, x`

E. `sample_list[element], continue, element[x]`

19. What will be the output of the following code segment?

```
def func(x=3):
    if x >= 5:
        return 1
    return x + func(x+1)

print(func())
```

A. 1   B. 3   C. 5   D. 8   E. 12

20. Which of the following will sort the following list of dictionaries `player_list` in the descending order of the score of each player?

```
player_list = [{"player": "Alex", "Score": 100},
               {"player": "Bob", "Score": 80},
               {"player": "Dylan", "Score": 90}]
```

    A. `sorted(player_list, key = lambda x:x[1])`

    B. `sorted(player_list, key = lambda x:x[1], reverse=True)`

    C. `sorted(player_list, key = lambda x:x["Score"])`

    D. `sorted(player_list, key = lambda x:x["Score"], reverse=True)`

    E. `sorted(player_list.items(), key = lambda x:x["Score"], reverse=True)`

21. Given the following function, which of the following function calls will return the largest value?

```
def func(x):
    total = x
    if x < 2:
        total += 2
    if x < 5:
        total += 3
    elif x % 2 == 0:
        total /= 2
    else:
        total += 1
    return total
```

A. `func(1)`   B. `func(4)`   C. `func(5)`   D. `func(8)`

22. Which of the following code block will cause the final value of `a` to be `[2, 2, 0]`?

```python
import copy
a=[]
a.append([1,0])
a.append(2)
b=copy.copy(a)
#Add one of the code blocks here
```

      A. ```python
b[0]=2
a.pop(0)
a.append(b[0])
```

      B. ```python
b[0]=2
a.pop(0)
a.extend(b[0])
```

      C. ```python
b[0][0]=2
a.pop(0)
a.append(b[0])
```

      D. ```python
b[0][0]=2
a.pop(0)
a.extend(b[0])
```

23. What value is printed after the following code is run?

```python
forecast_a=["Clear", "Windy", "Rain", "Thunderstorm", "Windy"]
weather_options=list(set(forecast_a))
actual=["Clear", "Clear", "Rain"]
correct_forecast=[]
for idx in range(len(weather_options)):
    try:
        if actual[idx]==forecast_a[idx]:
            correct_forecast.append(actual[idx])
    except:
        correct_forecast.append("Unknown")
print(correct_forecast)
```

      A. `["Clear", "Rain"]`

      B. `["Clear", "Unknown", "Rain"]`

      C. `["Clear", "Rain", "Unknown"]`

      D. `["Clear", "Rain", "Unknown", "Unknown"]`

      E. `["Clear", "Unknown", "Rain", "Unknown"]`

24. Which of the following list comprehensions will update `our_list` to the same value as the code below?

```
player_list = [{"player": "Alex", "Score": 100},
               {"player": "Bob", "Score": 80},
               {"player": "Dylan", "Score": 90}]
our_list=[]
i=0
while(i<=2):
    our_list.append(player_list[i]["player"])
    i+=1
```

    A. our_list = [x for x in player_list]

    B. our_list = [x for x in player_list if x!="Dylan"]

    C. our_list = [x["player"] for x in player_list]

    D. our_list = [x["player"] for x in player_list if
       x["player"]!="Dylan"]

25. What is the output of the following lines of code?

```
encouragement="We believe in you!"
def positivity(x):
    encouragement=x*2
    return encouragement
positivity("You got this! ")
print(encouragement)
```

    A. We believe in you!

    B. You got this!

    C. You got this! You got this!

    D. We believe in you!You got this! You got this!

26. Consider the lines of code below one at a time. How many of these lines have the potential for throwing an error, assuming all other lines of code run correctly?

```python
from collections import namedtuple
Course=namedtuple("Course", ["sections", "credits"])
def update (x, y=1):
    x.sections=y
csCourse=Course(4, 3)
update(y=3, csCourse)
```

      A. 0

      B. 1

      C. 2

      D. 3

      E. None of the above

27. Consider the following lines of code. Each line in the code is represented by a line number

```python
# Line 1 def predict(glucose = 120, bmi = 50, age = 30):
# Line 2     if glucose <= 127.5:
# Line 3         if age <= 28.5:
# Line 4             if bmi <= 45.4:
# Line 5                 return False
# Line 6             elif bmi <= 55.3:
# Line 7                 return True
# Line 8             else:
# Line 9                 return False
# Line 10        else:
# Line 11            return True
```

Select the option that best describes the correct order of execution for the following function call:

```python
predict (glucose = 120, bmi = 60, age = 25)
```

      A. 1, 2, 10, 11

      B. 1, 2, 3, 4, 6, 7

      C. 1, 2, 3, 4, 8, 10, 11

      D. 1, 2, 3, 4, 5, 6, 7

      E. 1, 2, 3, 4, 6, 8, 9

28. Consider the following code snippet

```
def predict(glucose = 120, bmi = 50, age = 30):
    #this function has some lines of code here!
```

Which of the following function calls have the correct syntax?

        A. `answer = predict(glucose = 120)`

        B. `answer = predict(50, 30, 48, glucose = 120)`

        C. `answer = predict(70, 30)`

        D. `answer = predict(glucose = 120, 50, 30)`

        E. Both `A` and `C`

29. What is printed by the following code segment?

```
import copy
x = ["D", "S", [2, 2, 5]]
y = x
y[0] = "C"
z = copy.copy(x)
z[2][2] = 0
z.append("is awesome!")
print(x,z)
```

        A. `["C", "S", [2, 2, 0]] ["C", "S", [2, 2, 0], "is awesome!"]`

        B. `["D", "S", [2, 2, 0]] ["D", "S", [2, 2, 0], "is awesome!"]`

        C. `["C", "S", [2, 2, 0]] ["C", "S", [2, 2, 0]]`

        D. `["C", "S", [2, 2, 5]] ["C", "S", [2, 2, 0]]`

        E. `["D", "S", [2, 2, 5]] ["C", "S", [2, 2, 0], "is awesome!"]`

30. Which lines are executed in the following code? Use the marked line number (comment) to answer this question.

Hint: Note the difference between "which lines are executed" and "which lines are successfully executed." This question is asking, "which lines are executed."

```python
#line 1 def f(a, b):
#line 2     return a+b

#line 3 def g(a):
#line 4     return len(a)

#line 5 def h():
#line 6     try:
#line 7         c = 3
#line 8         d = 4
#line 9         return g(uninitialized_variable) * f(c,d)
#line 10     except:
#line 11         return "An error occurred!"

#line 12 try:
#line 13    c = 6
#line 14    d = -100
#line 15    print(f(c,d) + h())
#line 16 except:
#line 17    print("An error occurred!")
```

    A. All lines except line 4

    B. All lines except line 11

    C. All lines except line 17

    D. All lines except lines 4, 16, and 17

    E. All lines except lines 9, 16, and 17

Buildings (Pandas)

Consider the following code:

```
capital = {"Name": "State Capitol", "Height": 284, "Floors": 6, "Year": 1917}
van_hise = {"Name": "Van Hise Hall", "Height": 243, "Floors": 19, "Year": 1967}
sterling = {"Name": "Sterling", "Height": 184, "Floors": 14, "Year": 1968}
humanities = {"Name": "Humanities", "Height": 180, "Floors": 15, "Year": 1966}
office = {"Name": "State Offices", "Height": 177, "Floors": 13, "Year": 1939}

data = pd.DataFrame([capital, van_hise, sterling, humanities, office])
```

31. Which line of code gets a series of the heights in `data`? Note that the order of the series does NOT matter.

    A. `data.sort_values("Name")["Height"]`

    B. `data["Height"].iloc[:5]`

    C. `data["Height"].sort_values()`

    D. `data["Height"]`

    E. `All of the Above`

32. How can we generate a DataFrame of the buildings with at least 15 floors?

    A. `data[data["Floors"] >= 15].sort_values("Height")`

    B. `data["Floors"] >= 15`

    C. `data["Floors" >= 15]`

    D. `data.sort_values("Floors").loc[0:2]`

    E. `None of the Above`

33. Which of the following commands will output the `Name` of the shortest building in the DataFrame?

    A. `data["Height"].min()["Name"]`

    B. `data["Height"]["Name"].min()`

    C. `data["Height"].min().iloc[0]`

    D. `data[data["Height"] == data["Height"].min()].Name`

    E. `None of the Above`

34. What is the output of the following command?

    ```
    data["Year"].iloc[:-2]
    ```

    A. `Series(0:1917, 1:1967, 2:1968)`

    B. `Series(0:1917, 1:1967, 2:1968, 3:1966)`

    C. `Series(2:1968)`

    D. `[1917, 1967, 1968]`

    E. `None of the Above`

35. How can we generate a DataFrame of all the buildings taller than 180 feet with more than 13 floors?

    A. `data[data["Height"] > 180 & data["Floors"] > 13]`

    B. `(data["Height"] > 180) and (data["Floors"] > 13)`

    C. `data[(data["Height"] > 180) & (data["Floors"] > 13)]`

    D. `data[(data["Height"] > 180) and (data["Floors"] > 13)]`

    E. `data[(data["Height"] > 180) | (data["Floors"] > 13)]`

(Blank Page)