# 446 Midterm 2 Solutions[1]

## 1. Question 1

Suppose we run the commands

```
import pandas as pd
data = {
    "state": ["Ohio", "Ohio", "Ohio", "Nevada", "Nevada", "Nevada"],
    "year": [2000, 2001, 2002, 2001, 2002, 2003],
    "pop": [1.5, 1.7, 3.6, 2.4, 2.9, 3.2]
}
frame = pd.DataFrame(data)
```

(a) What is the output of the following commands?

```
frame2 = frame.reindex(index = [3, 2, 5])
frame2
```

(b) What is the output of the following commands?

```
frame3 = frame2.set_index("year")
frame3
```

*Solution.*

```
    state  year  pop
3  Nevada  2001  2.4
2    Ohio  2002  3.6
5  Nevada  2003  3.2
```

```
          state    pop
year
2001  Nevada    2.4
2002    Ohio    3.6
2003  Nevada    3.2
```

## 2. Question 2

What is the output of the following program? Explain your reasoning.

```
import re
data = '''
"data-testid="bar-chart--results-bar" style="width:51%"
role="progressbar" aria-valuenow="51" class="jsx-4201391551
jsx-842384122 labeled-bar df white"><span data-testid=
"bar-chart--results-bar-percent" class="jsx-4201391551 jsx-842384122"
'''
search_string = r'jsx([\w-]{3})'

found_strings = re.findall(search_string, data)
found_strings
```

*Solution.*

[1] November 6, 2024, © 2024 Steven Heilman, All Rights Reserved.

```
['-42', '-84', '-42', '-84']
```

## 3. Question 3

Suppose we have a list of strings of the following form

```
strings = ['Blue Horse', 'Purple Cat', 'White Dog', 'Yellow Duck']
```

Write a Python function that removes the spaces from this list. That is, the output should be

```
['BlueHorse', 'PurpleCat', 'WhiteDog', 'YellowDuck']
```

*Solution.*

```
strings = ['Blue Horse', 'Purple Cat', 'White Dog', 'Yellow Duck']

def delete_space(strings):
    output = []
    for string in strings:
        index1 = string.find(' ')
        output.append(string[:index1] + string[1 + index1:])
    return output

delete_space(strings)
```

## 4. Question 4

Suppose we have a Pandas DataFrame named `df` with the following entries

```
            product_name    units_sold  unit_price  sale_date   region
product_id
4           widget_a        150         2.5         2023-01-10  east
3           widget_b        200         3.0         2023-01-12  east
2           widget_c        250         1.5         2023-01-14  west
1           widget_d        300         4.0         2023-01-10  south
0           widget_e        100         5.0         2023-01-15  east
```

Answer the following questions.

- What is the output of `df.drop(index = [4, 1, 0])`
- What is the output of `df.reindex(np.arange(6), method = "ffill")`
- What is the output of `df[2]["units_sold"]` ?
- Write a single line of Python code that returns a DataFrame containing only the rows of `df` where sales occurred in the `east` region.
- Write a single line of Python code to compute the total sales for each row of `df` (i.e. compute `units_sold` multiplied by `unit_price`) and create a new column of `df` called `total_sales` that contains the total sales of each row of `df`.

*Solution.*

- The output is `df` without the rows 4,1 and 0.
- The output is `df` with a reordered index of $0, 1, 2, 3, 4, 5$. The fifth row is filled with `NaN` values.

- This produces an error (exception) since we ordered the command incorrectly. In Pandas we have to specify the column then the row, but there is no column named 2, so an error occurs.
- `df[df["region"] == "east"]`
- `df["total_sales"] = df["units_sold"] * df["unit_price"]`.

We can test these commands with the following DataFrame instantiation.

```
import pandas as pd
diction = {
    "product_id" : [4, 3, 2, 1, 0],
    "product_name": ["widget_a", "widget_b", "widget_c", "widget_d", "widget_e"],
    "units_sold": [150, 200, 250, 300, 100],
    "unit_price": [2.5, 3.0, 1.5, 4.0, 5.0],
    "sale_date": ["2023-01-10", "2023-01-10", "2023-01-10", "2023-01-10", "2023-01-10"],
    "region": ["east", "east", "west", "south", "east"]
}
df = pd.DataFrame(diction)
df.set_index("product_id", inplace = True)
```