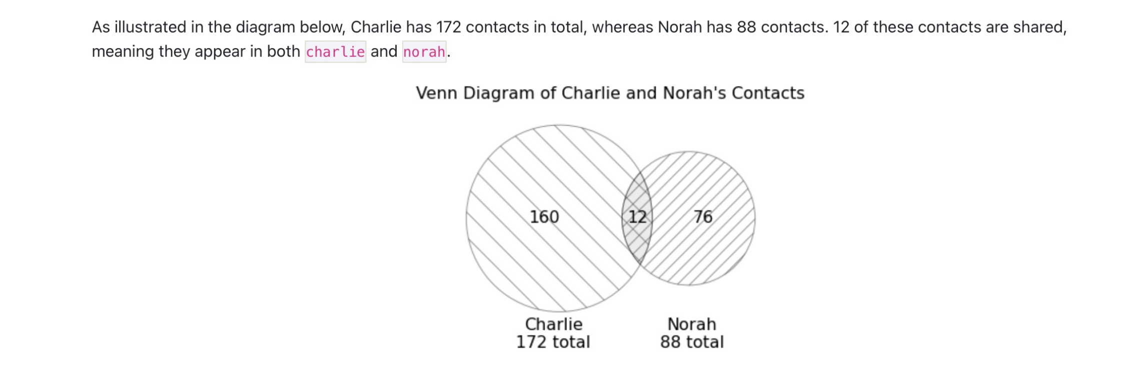
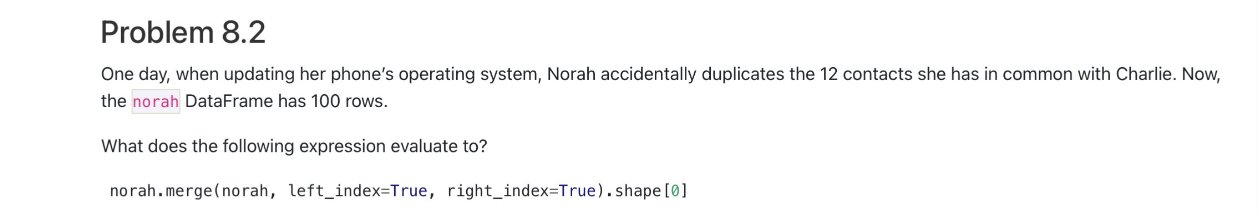
Disc 4, 8.2



As illustrated in the diagram below, Charlie has 172 contacts in total, whereas Norah has 88 contacts. 12 of these contacts are shared, meaning they appear in both `charlie` and `norah`.



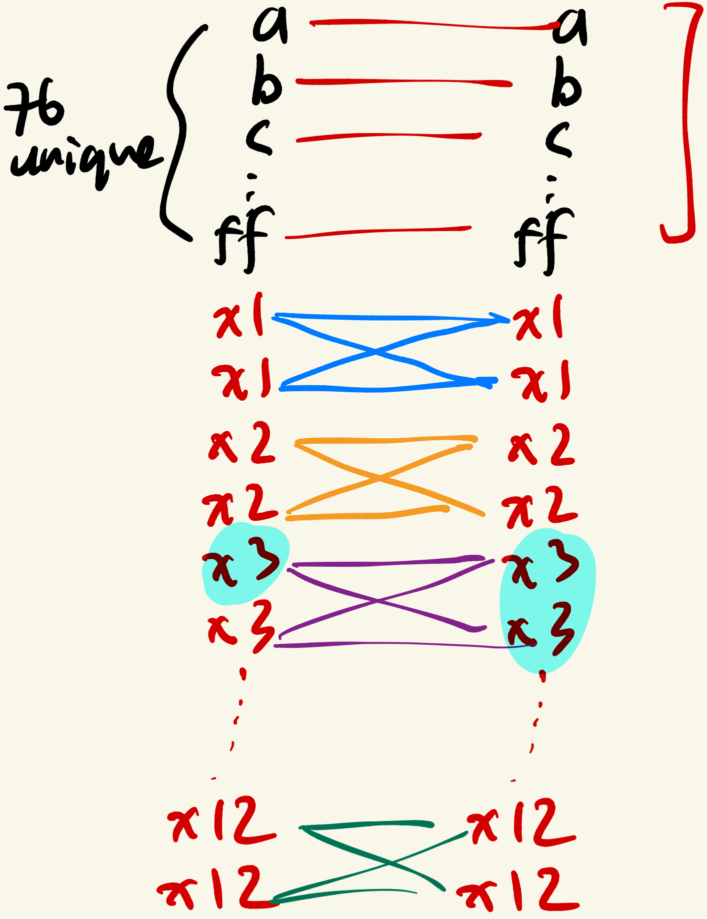
Problem 8.2

One day, when updating her phone's operating system, Norah accidentally duplicates the 12 contacts she has in common with Charlie. Now, the `norah` DataFrame has 100 rows.

What does the following expression evaluate to?

```
norah.merge(norah, left_index=True, right_index=True).shape[0]
```

$$76\ \text{unique}\left\{\begin{array}{l}a\\b\\c\\d\\\vdots\\xy\end{array}\right.$$



$(1\cdot 1)\,76 = 76$

4 x1s

4 x2s

4 x3s

⋮

4 x12s

$= 4\cdot 12$

$= 2\cdot 24$

$= 48$

Discussion 5, 4-2

⇒ What does .strip() do?

removes from beginning and end of string

```
"        hi        ".strip() → "hi"

"""

     hi        .strip()

"""
```

"+74++".strip("+")

→ "74"

Discussion 5, 3.5

r= requests.get (url)

⇒ r is a "Response" object

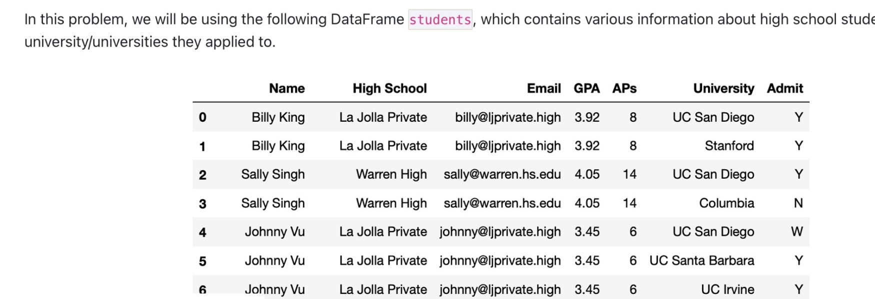
⇒ r.text is the content

- ⇒ always a string → BeautifulSoup(r.text)
- ⇒ could look like HTML → use BeautifulSoup
- ⇒ could look like JSON
  - ⇒ r.json()

↑ Python dictionary

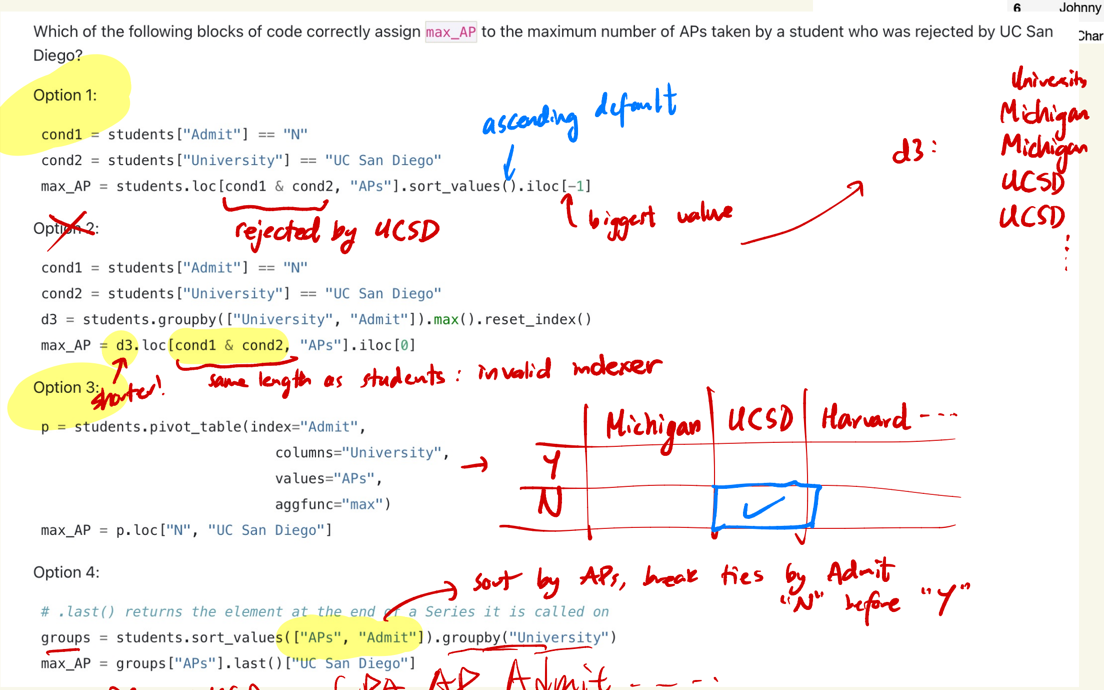
Monday Review, Problem 2

Answer: Option 1, Option 3



In this problem, we will be using the following DataFrame `students`, which contains various information about high school stude
university/universities they applied to.

	Name	High School	Email	GPA	APs	University	Admit
0	Billy King	La Jolla Private	billy@ljprivate.high	3.92	8	UC San Diego	Y
1	Billy King	La Jolla Private	billy@ljprivate.high	3.92	8	Stanford	Y
2	Sally Singh	Warren High	sally@warren.hs.edu	4.05	14	UC San Diego	Y
3	Sally Singh	Warren High	sally@warren.hs.edu	4.05	14	Columbia	N
4	Johnny Vu	La Jolla Private	johnny@ljprivate.high	3.45	6	UC San Diego	W
5	Johnny Vu	La Jolla Private	johnny@ljprivate.high	3.45	6	UC Santa Barbara	Y
6	Johnny Vu	La Jolla Private	johnny@ljprivate.high	3.45	6	UC Irvine	Y
	Charles	Triton Magnet High	cassie@triton.high	3.84	9	UC San Diego	N



Which of the following blocks of code correctly assign `max_AP` to the maximum number of APs taken by a student who was rejected by UC San Diego?

Option 1:

```
cond1 = students["Admit"] == "N"
cond2 = students["University"] == "UC San Diego"
max_AP = students.loc[cond1 & cond2, "APs"].sort_values().iloc[-1]
```

ascending default

rejected by UCSD

biggest value

~~Option 2~~:

```
cond1 = students["Admit"] == "N"
cond2 = students["University"] == "UC San Diego"
d3 = students.groupby(["University", "Admit"]).max().reset_index()
max_AP = d3.loc[cond1 & cond2, "APs"].iloc[0]
```

shorter!

same length as students: invalid indexer

d3:

University	Admit	GPA	APs	Name	High School
Michigan	Y				
Michigan	N				
UCSD	Y				
UCSD	N	✓			
⋮					

Option 3:

```
p = students.pivot_table(index="Admit",
                         columns="University",
                         values="APs",
                         aggfunc="max")
max_AP = p.loc["N", "UC San Diego"]
```

	Michigan	UCSD	Harvard - - -
Y			
N		✓	

Option 4:

```
# .last() returns the element at the end of a Series it is called on
groups = students.sort_values(["APs", "Admit"]).groupby("University")
max_AP = groups["APs"].last()["UC San Diego"]
```

sort by APs, break ties by Admit "N" before "Y"

groups: UCSD GPA AP Admit - - - -
Michigan
Harvard

```
(
  students
  .sort_values(--)
  .groupby("University")
  ["APs"]
  .last()
  .loc["UCSD"]
)
```

DataFrameGroupBy object

Series
DataFrame
.iloc[-1]

```
df.sort_values("AP").groupby("University").last()
```

equivalent

```
df.sort_values("AP").groupby("University").apply(lambda f:
                                                      f.iloc[-1])
```

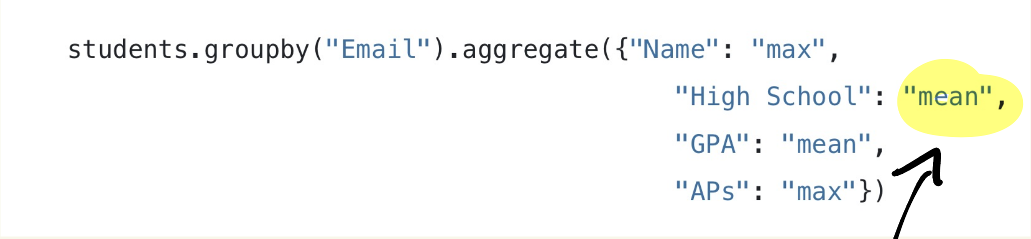
df

color	height	weight
red	2	4
blue	1	4
blue	4	2
red	7	3
green	3	1
blue	9	-1
green	1	0

```
df.groupby("color"). last()
```

usually, sort first

Monday Review, 2.3



```
students.groupby("Email").aggregate({"Name": "max",
                                     "High School": "mean",
                                     "GPA": "mean",
                                     "APs": "max"})
```

can't average strings!!!

$$f : S \rightarrow \text{"VMSS"}$$

$$S = \begin{bmatrix} \text{VMSS} \\ \text{VMSS} \\ \vdots \\ \text{VMSS} \end{bmatrix}$$

possible answers:

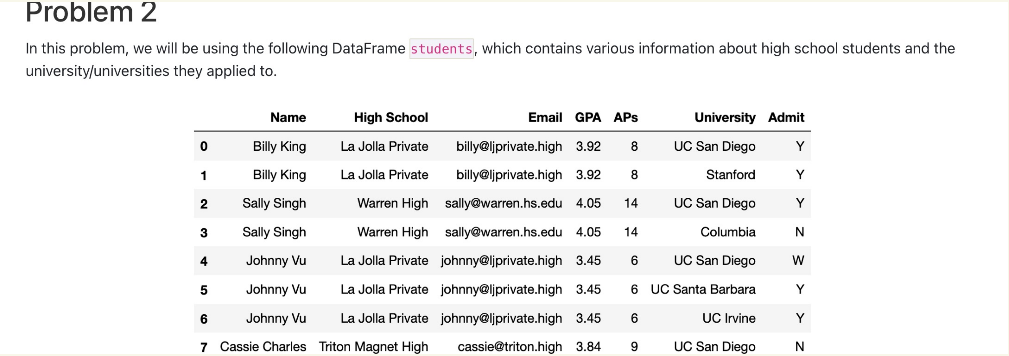
"min"

"max"

"first"

"last"

lambda hi: hi.iloc[0]



Problem 2

In this problem, we will be using the following DataFrame `students`, which contains various information about high school students and the university/universities they applied to.

	Name	High School	Email	GPA	APs	University	Admit
0	Billy King	La Jolla Private	billy@ljprivate.high	3.92	8	UC San Diego	Y
1	Billy King	La Jolla Private	billy@ljprivate.high	3.92	8	Stanford	Y
2	Sally Singh	Warren High	sally@warren.hs.edu	4.05	14	UC San Diego	Y
3	Sally Singh	Warren High	sally@warren.hs.edu	4.05	14	Columbia	N
4	Johnny Vu	La Jolla Private	johnny@ljprivate.high	3.45	6	UC San Diego	W
5	Johnny Vu	La Jolla Private	johnny@ljprivate.high	3.45	6	UC Santa Barbara	Y
6	Johnny Vu	La Jolla Private	johnny@ljprivate.high	3.45	6	UC Irvine	Y
7	Cassie Charles	Triton Magnet High	cassie@triton.high	3.84	9	UC San Diego	N

"17"

"9" → "17" < "9"

"17" vs. "EECS" : idk, but check ASCII lol

agg vs filter vs transform vs apply

see posted notebook and video under week 7!