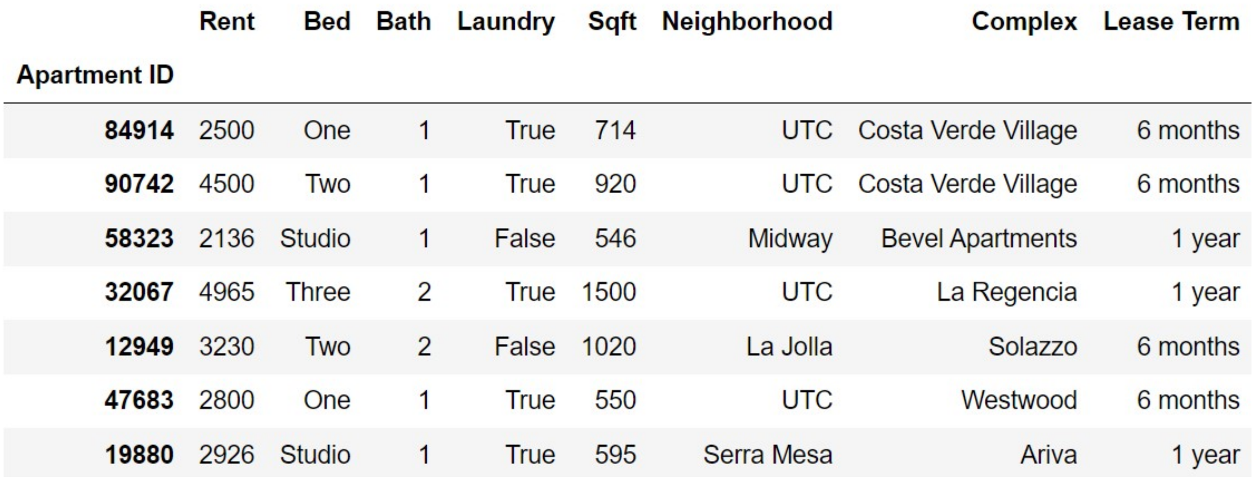
San Diego Apartments

Today, we'll explore data on 800 different apartments available for rent in San Diego. Each row in the DataFrame `apts` corresponds to an individual apartment. The DataFrame is indexed by `"Apartment ID" (int)`, which is a unique identifier for the apartment. The columns of `apts` are as follows:

- `"Rent" (int)`: The monthly rent for the apartment, in dollars.
- `"Bed" (str)`: The number of bedrooms in the apartment. Values are `"Studio"`, `"One"`, `"Two"`, and `"Three"`.
- `"Bath" (str)`: The number of bathrooms in the apartment. Values are `"One"`, `"One and a half"`, `"Two"`, `"Two and a half"`, and `"Three"`.
- `"Laundry" (bool)`: If the apartment comes with an in-unit washer and dryer.
- `"Sqft" (int)`: The area of the apartment, in square feet.
- `"Neighborhood" (str)`: The neighborhood in which the apartment is located.
- `"Complex" (str)`: The complex the apartment is a part of.
- `"Lease Term" (str)`: The duration of the apartment's lease. Values are `"1 month"`, `"6 months"`, and `"1 year"`.

The first few rows of `apts` are shown below, though `apts` has many more rows than pictured, 800 in total. The data in `apts` is only a sample from the much larger population of **all** San Diego apartments.



	Rent	Bed	Bath	Laundry	Sqft	Neighborhood	Complex	Lease Term
Apartment ID								
84914	2500	One	1	True	714	UTC	Costa Verde Village	6 months
90742	4500	Two	1	True	920	UTC	Costa Verde Village	6 months
58323	2136	Studio	1	False	546	Midway	Bevel Apartments	1 year
32067	4965	Three	2	True	1500	UTC	La Regencia	1 year
12949	3230	Two	2	False	1020	La Jolla	Solazzo	6 months
47683	2800	One	1	True	550	UTC	Westwood	6 months
19880	2926	Studio	1	True	595	Serra Mesa	Ariva	1 year

Throughout this exam, assume that we have already run `import babypandas as bpd` and `import numpy as np`.

Question 15 (8 pts)

Imagine a DataFrame constructed from `apts` called `bedrooms`, which has one row for each bedroom in an apartment in `apts`. More specifically, a one bedroom apartment in `apts` will appear as one row in `bedrooms`, a two bedroom apartment in `apts` will appear as two rows in `bedrooms`, and a three bedroom apartment in `apts` will appear as three rows in `bedrooms`. Studio apartments will not appear in `bedrooms` at all.

The `"Apartment ID"` column of `bedrooms` contains the `"Apartment ID"` of the apartment in `apts`. Notice that this is not the index of `bedrooms` since these values are no longer unique. The `"Cost"` column of `bedrooms` contains the rent of the apartment divided by the number of bedrooms. All rows of `bedrooms` with a shared `"Apartment ID"` should therefore have the same value in the `"Cost"` column.

a) (3 pts) Recall that `apts` has 800 rows. How many rows does `bedrooms` have?

○ 800
○ More than 800.
○ Less than 800.
● Not enough information.

Apt ID bed

2

1

0 ×

b) (5 pts) Suppose `no_studio` is defined as follows. (Remember, we previously converted the `"Beds"` column to integers.)

```
no_studio = apts[apts.get("Bed") != 0]
```

Which of the following statements evaluates to the same value as the expression below?

```
bedrooms.get("Cost").mean()
```

Select all that apply.

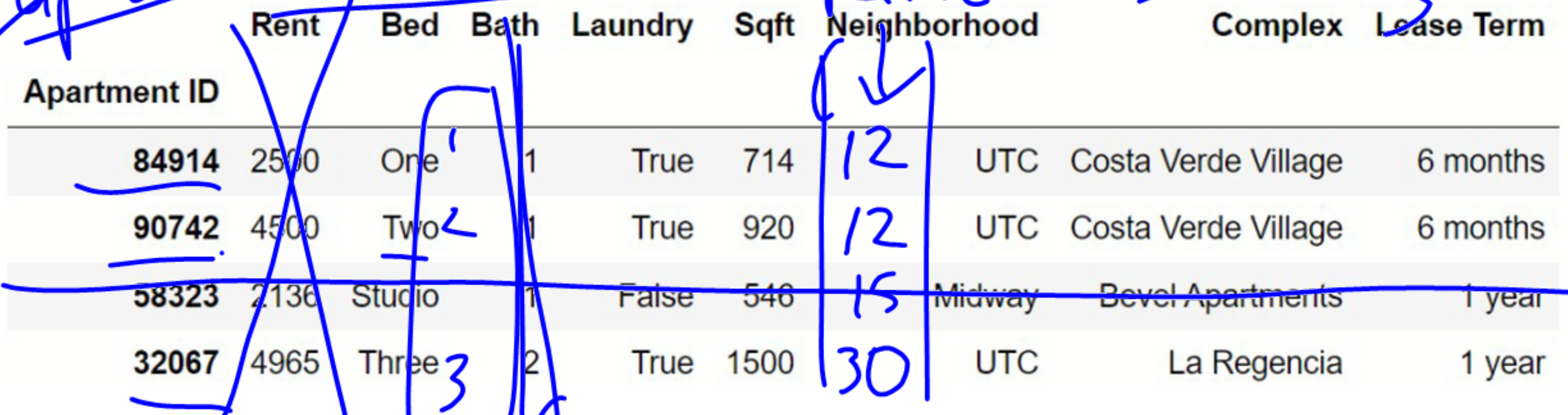
□ `no_studio.get("Rent").mean()`
☒ `no_studio.get("Rent").sum() / apts.get("Bed").sum()`
□ `(no_studio.get("Rent") / no_studio.get("Bed")).mean()`
□ `(no_studio.get("Rent") / no_studio.get("Bed").sum()).sum()`
☒ `no_studio.get("Rent").mean() / no_studio.get("Bed").mean()`
□ None of these.

$\begin{bmatrix} 12 \\ 12 \\ 30 \end{bmatrix} \div 6 = \begin{bmatrix} 2 \\ 2 \\ 5 \end{bmatrix}$

2+2+5

$\frac{12+12+30}{3}$

apts no_studio Rent



Apartment ID	Rent	Bed	Bath	Laundry	Sqft	Neighborhood	Complex	Lease Term
84914	2500	One	1	True	714	UTC	Costa Verde Village	6 months
90742	4500	Two	1	True	920	UTC	Costa Verde Village	6 months
58323	2136	Studio	1	False	546	Midway	Bevel Apartments	1 year
32067	4965	Three	2	True	1500	UTC	La Regencia	1 year

1 2 3

6

12 12 15 30

bedrooms →

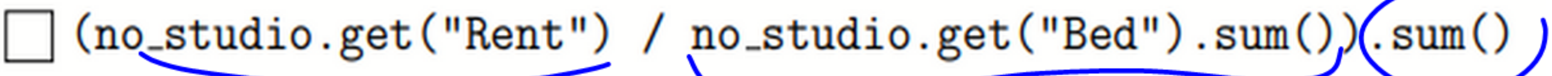
	Cost
84914	12
90742	6
90742	6
32067	10
32067	10
32067	10

> mean of all six #s

$$\frac{\underbrace{12+\underbrace{6+6}}+\underbrace{10+10+10}}{6} = \frac{12+12+30}{6}$$

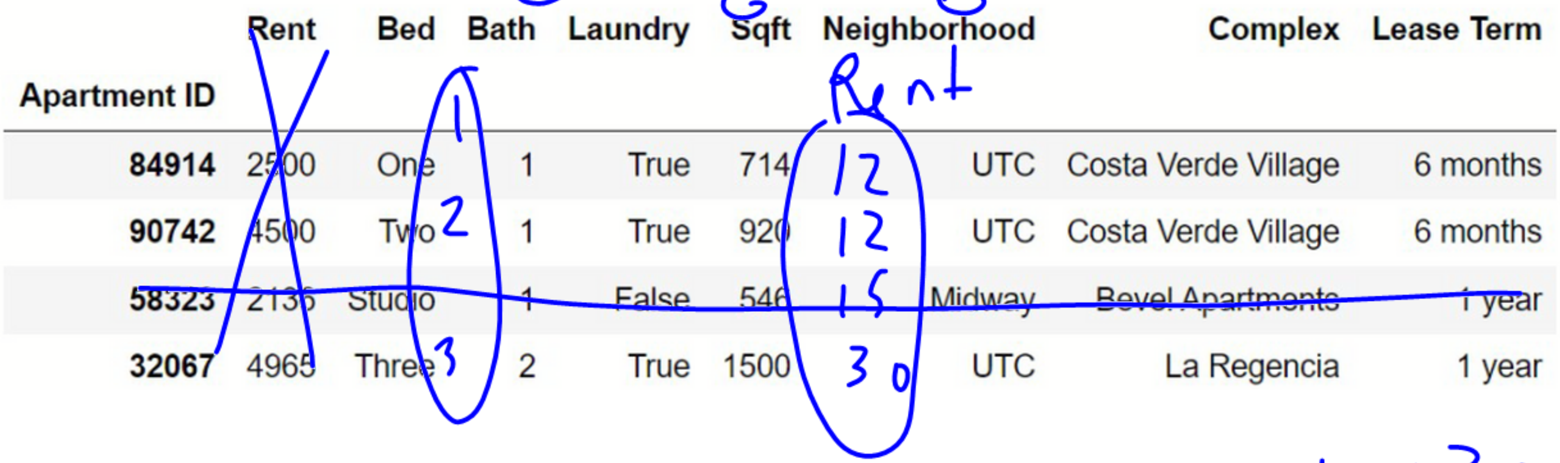
$$2+2+5$$

$$\begin{bmatrix}12\\12\\30\end{bmatrix} \Big/ 6 = \begin{bmatrix}2\\2\\5\end{bmatrix}$$

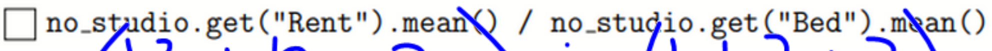


```
□ (no_studio.get("Rent") / no_studio.get("Bed").sum()).sum()
```

$$\frac{12}{6}+\frac{12}{6}+\frac{30}{6}$$



Apartment ID	Rent	Bed	Bath	Laundry	Sqft	Neighborhood	Complex	Lease Term
84914	2500	One	1	True	714	UTC	Costa Verde Village	6 months
90742	4500	Two	1	True	920	UTC	Costa Verde Village	6 months
~~58323~~	~~2136~~	~~Studio~~	~~1~~	~~False~~	~~546~~	~~Midway~~	~~Bevel Apartments~~	~~1 year~~
32067	4965	Three	2	True	1500	UTC	La Regencia	1 year



```
□ no_studio.get("Rent").mean() / no_studio.get("Bed").mean()
```

$$\left(\frac{12+12+30}{3}\right) \div \left(\frac{1+2+3}{3}\right) = \frac{12+12+30}{1+2+3}$$

Question 9 (20 pts)

For each expression below, determine the data type of the output and the value of the expression, if possible. Write the data type in the left box and the value in the right box. If there is not enough information to determine the expression's value, write "Unknown" in the right box.

a) (4 pts) `apts.get("Rent").iloc[43] * 4 / 2`

type: □ value: □

b) (4 pts) `apts.get("Neighborhood").iloc[2][-3]`

type: □ value: □

c) (4 pts) `(apts.get("Laundry") + 5).max()`

type: □ value: □

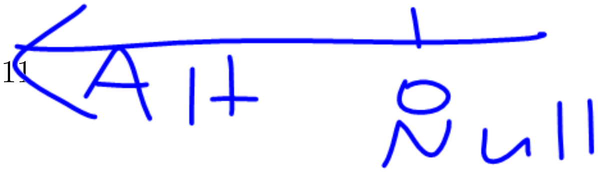
d) (4 pts) `apts.get("Complex").str.contains("Verde")`

type: □ value: □

e) (4 pts) `apts.get("Sqft").median() > 1000`

type: □ value: □

if we did non-UTC



UTC - non

Null | Alt

Question 10 (28 pts)

We want to use the data in `apts` to test the following hypotheses:

UTC = non

- **Null Hypothesis**: The rents of the apartments in UTC and the rents of the apartments in other neighborhoods come from the same distribution.

UTC > non

- **Alternative Hypothesis**: The rents of the apartments in UTC are **higher** than the rents of the apartments in other neighborhoods on average.

nbhd	re
UTC	15
non	10
UTC	12
UTC	9

While we could answer this question with a permutation test, in this problem we will explore another way to test these hypotheses. Since this is a question of whether two samples come from the same unknown population distribution, we need to construct a "population" to sample from. We will construct our "population" in the same way as we would for a permutation test, except we will draw our sample differently. Instead of shuffling, we will draw our two samples **with replacement** from the constructed "population." We will use as our test statistic the difference in means between the two samples (in the order **UTC minus elsewhere**).

put both samples together (in apts)

15

15

9

12

a) (13 pts) Suppose the data in `apts`, which has 800 rows, includes 85 apartments in UTC. Fill in the blanks below so that `p_val` evaluates to the p-value for this hypothesis test, which we test according to the strategy outlined above.

```
diffs = np.array([])
for i in np.arange(10000):
    utc_sample_mean = __(a)__
    elsewhere_sample_mean = __(b)__
    diffs = np.append(diffs, utc_sample_mean - elsewhere_sample_mean)
observed_utc_mean = __(c)__
observed_elsewhere_mean = __(d)__
observed_diff = observed_utc_mean - observed_elsewhere_mean
p_val = np.count_nonzero(diffs __(e)__ observed_diff) / 10000
```

a: apts.sample(85, replace=True).get("Rent").mean()

b: apts.sample(715, replace=True).get("Rent").mean()

c: apts[apts.get("Neighborhood") == "UTC"].get("Rent").mean()

d: apts[apts.get("Neighborhood") != "UTC"].get("Rent").mean()

e: ○ > ● >= ○ < ○ <= ○ == ○ !=