Question 1

You have a DataFrame, `books`, where the rows represent individual books and the columns are `"title"`, `"author"`, `"genre"`, and `"price"` as a float in dollars.

a) Suppose both of the following DataFrames have the same number of rows.

```
books.groupby(["author", "genre"]).count()
books.groupby("author").count()
```

In **one sentence**, interpret what this means about authors and genres.

b) Write **one line of code** that creates a DataFrame with the same contents as `books`, except omit books where the mean price of all books in that genre is less than 15 dollars.

c) Write **one line of code** that evaluates to a Series indexed by `"genre"`, containing a count of the number of books in that genre that are priced above the average price for that genre.

Question 2

a) Suppose DataFrames `a` and `b` do not have any missing values. We merge them as follows:

```
merged = a.merge(b, on="key", how="left")
```

In **one sentence**, justify whether it is possible for `merged` to have any missing values.

b) Suppose we merge DataFrames `c` and `d` as follows:

```
result = c.merge(d, on="key", how="right")
```

You are told that `result` and `d` have the same number of rows. Which of the following statements are **necessarily true**? **Select all that apply.**

1. `c` and `d` have the same number of rows.
2. Every key that appears in `c` also appears in `d`.
3. Every key that appears in `d` also appears in `c`.
4. No key that appears in `c` appears more than once in `d`.
5. No key that appears in `d` appears more than once in `c`.
6. All the keys in `c` are distinct (no duplication).
7. All the keys in `d` are distinct (no duplication).

Question 3

a) Which of these questions require a **permutation test**? **Select all that apply.**

1. Do more people own their home or rent their home?
2. Do one quarter of people have blue eyes and three quarters of people have eyes that are not blue?
3. Are pink clothes more expensive than clothes that are not pink?

b) You want to test whether left-handed and right-handed people have the same distribution of intelligence quotient (IQ) or whether right-handed people have **lower** IQs, on average.

You plan to simulate values of a test statistic (in the array `stats`), calculate an observed value of the statistic (`obs`), and compute the p-value using `(stats >= obs).mean()`. Which test statistic(s) could you use? **Select all that apply.**

1. Difference in mean IQs (right minus left).
2. Difference in mean IQs (left minus right).
3. Absolute difference in mean IQs.
4. Total variation distance.
5. Kolmogorov-Smirnov statistic.

c) In the DataFrame `cookbook`, each row represents a recipe, and the `"ingredients"` columns records the number of ingredients in the recipe. The `"category"` column contains either `"savory"` or `"sweet"`. Define `means` as follows:

```
means = cookbook.groupby("category")["ingredients"].mean()
```

Which of the following computes the average number of `"ingredients"` for `"savory"` recipes minus the average number of `"ingredients"` for `"sweet"` recipes? **Select all that apply.**

1. `means.diff().iloc[-1]`
2. `means.diff().iloc[-1] * -1`
3. `means.diff().sum() * -1`
4. `means.diff().iloc[0]`
5. `means.iloc[0] - means.iloc[1]`
6. `means.iloc[1] - means.iloc[0]`

Question 4

In the DataFrame `fitness`, the `"steps"` column records how many steps Noah took each day, and the `"day"` column records the day of the week (e.g. `"Monday"`). Some values in `"steps"` are missing at random (MAR) dependent on `"day"`.

a) Fill in the blanks to perform **mean imputation conditional on day of the week**. The resulting DataFrame, `filled`, should have no missing values in `"steps"`.

```
def cond_mean_impute(steps):
    steps = steps.copy()
    steps[steps.isna()] = ___(i)___
    return steps

filled = fitness.copy()
filled["steps"] = (fitness.___(ii)___.___(iii)___)
```

b) We now want to perform **probabilistic imputation conditional on day of the week**. Fill in blank `(i)` above in a different way, such that keeping the rest of the code unchanged correctly accomplishes this.

c) Suppose that Noah's **actual** daily step counts, including those that were missing from `fitness`, form a bell curve centered at 10,000 with a standard deviation of 2,000.

- Which imputation strategy above gives a distribution that better represents this true data?
- For both strategies, state whether the standard deviation of the imputed data is less than, greater than, or approximately equal to 2,000.

Question 5

At a university, an end-of-term course survey asks students for their `"instructor rating"`, `"grade"`, and `"graduation year"`. This table shows the distribution of `"grade"` for survey responses where the `"instructor rating"` was missing, and the corresponding distribution for where it was not missing (observed). Note that both columns sum to 1.

`"grade"`	`"instructor rating"` missing	`"instructor rating"` observed
A	0.11	0.25
B	0.19	0.20
C	0.15	0.18
D	0.25	0.22
F	0.30	0.15

We want to do a permutation test to determine whether the missingness of `"instructor rating"` might be dependent on `"grade"`.

a) Which test statistic should be used for this permutation test? Calculate the observed value of this statistic, **showing your work**.

b) Suppose the p-value is 0.71 and we have previously determined that `"instructor rating"` is neither missing by design (MD) nor missing not at random (MNAR). Can we conclude that `"instructor rating"` is missing completely at random (MCAR)? **Justify your answer**.

c) Justify in **one sentence** why the `"instructor rating"` could actually be missing not at random (MNAR).

Question 6

The DataFrame `grades` is indexed by unique `"PID"` numbers. It has one student `"name"` column and many columns for student scores on various course assignments. Missing values (`NaN`) occur if and only if a student did not submit a particular assignment.

For each part, write **one line of code** to answer the question. Use `numpy` and `pandas` operations, and avoid loops.

a) Create a Series indexed by `"PID"` with the number of assignments each student submitted.

b) **Without sorting**, find the name of the student with the highest score on `"Project 2"`, assuming there were no ties.

c) **Without grouping**, find the most common score on `"Homework 4"`, if we consider students who did not submit the assignment as earning a score of 0.