Question 1:
Write a function that returns the second-largest number in a given list of integers.
(Provide your code and a short explanation of your approach.)

```
def find_second_largest(data):
"""
Finds and returns the second-largest number in a list of integers.
data (list): A list of integers.
"""
if len(data) < 2:
return None # Cannot find a second largest in a list with < 2 elements

# Remove duplicates and sort the unique elements
unique_sorted = sorted(list(set(data)))

# Check if there is a second unique element
if len(unique_sorted) < 2:
return None # All elements are the same, [5, 5, 5]

# The second-largest is the second-to-last element in the sorted unique list
return unique_sorted[-2]
```

Explanation.

Unique Elements (set()) - Convert the input list to a set to automatically remove any duplicate integers.

Convert Back and Sort (list() and sorted()) - Convert the set back into a list and sort it in ascending order.

Edge Case Handling - Check if the resulting list has at least two unique elements.

Indexing - The second-largest number will be the element at the second-to-last position, which is accessed using the index −2 in Python. unique_sorted[-1] is the largest, and unique_sorted[-2] is the second-largest.

Question 2:
Explain how you would optimize a page that loads too slowly. Mention at least **three causes** and how you'd fix each.

Excessive Javascript/CSS files - Minification of HTML, CSS, and JavaScript by removing whitespace and comments and also compressing text assets using Gzip on the server. Optimizing images using appropriate next-gen formats such as WebP, compressing them, and implementing lazy loading for the images.

Excessive HTTP Requests from a page - Bundle and concatenate multiple small CSS and JavaScript files into fewer, larger files to reduce the number of initial requests.

Server Response Delay – One can defer and async loading , optimize database queries, or upgrade hosting.

Question 3 (Front-end):
You are creating a simple profile page that fetches user data from an API

(https://jsonplaceholder.typicode.com/users/1).
Explain or show code for:

- Fetching and displaying the user's name and email.

- Handling the loading and error states.

```
import React, { useEffect, useState } from "react";

export default function UserProfile() {
const [user, setUser] = useState(null);
const [loading, setLoading] = useState(true);
const [error, setError] = useState(null);

useEffect(() => {
fetch("https://jsonplaceholder.typicode.com/users/1")
.then((res) => {
if (!res.ok) throw new Error("Failed to fetch data");
return res.json();
})
.then((data) => {
setUser(data);
setLoading(false);
})
.catch((err) => {
setError(err.message);
setLoading(false);
});
}, []);

if (loading) return <p>Loading...</p>;
if (error) return <p>Error: {error}</p>;

return (
<div>
<h2>{user.name}</h2>
<p>{user.email}</p>
</div>
);
}
```

Explanation

The useEffect() fetches user data when the component mounts. Loading shows a message while data loads. The error handles network or fetch failures and once fetched, the component displays the user's name and email.

Question 4 (Back-end / Logic):
A small store wants to calculate total sales from this dataset:

```
def total_sales(data):
return sum(item["price"] * item["quantity"] for item in data)

sales = [
{"item": "Pen", "price": 20, "quantity": 3},
{"item": "Book", "price": 200, "quantity": 2},
{"item": "Bag", "price": 800, "quantity": 1}
]

print(total_sales(sales))
```

Question 5:
You've been given this code snippet:

```
numbers = [1, 2, 3, 4, 5]
for i in range(len(numbers)):
    if i % 2 == 0:
        numbers.remove(i)
print(numbers)
```

1. What's wrong with the code?

The code calls numbers.remove(i), which removes the value equal to i, not the element at index (i) and Since 0 is not in the list, it raises a Value Error

2. What will it output?

```
ValueError: list.remove(x): x not in list
```

3.How would you fix it to remove even numbers correctly?

```
numbers = [1, 2, 3, 4, 5]
numbers = [n for n in numbers if n % 2 != 0]
print(numbers) # Output: [1, 3, 5]
```

Question 6:
Explain how you would use Git to collaborate on a team project with other developers.
Mention at least:

- One common Git command you use often.
- One problem you've faced while using Git and how you solved

Each developer clones the shared repository using git clone <repo_url>.

The work is done on separate branches to avoid conflicts.

After committing changes (git commit), developers push their branch (git push origin feature-login).

A pull request is created for code review before merging into the main branch.

`git pull` — used to fetch and merge the latest changes from the remote repository to stay up to date with the team.

Problem faced - Merge conflict when two people edited the same file.
Solution: Used git status to identify conflicted files, manually resolved differences in the file.