# LAB TEST-2

## AI ASSIST CODING

Name:D.Prasanna Latha

Roll no:2503A51L37

Batch no:24BTCAICSB20

## TASK-1:

- Implement a fare function: fare = km * base_per_km * surgeMultiplier, where surge applies strictly after 18:00 local time

**Prompt Given:**

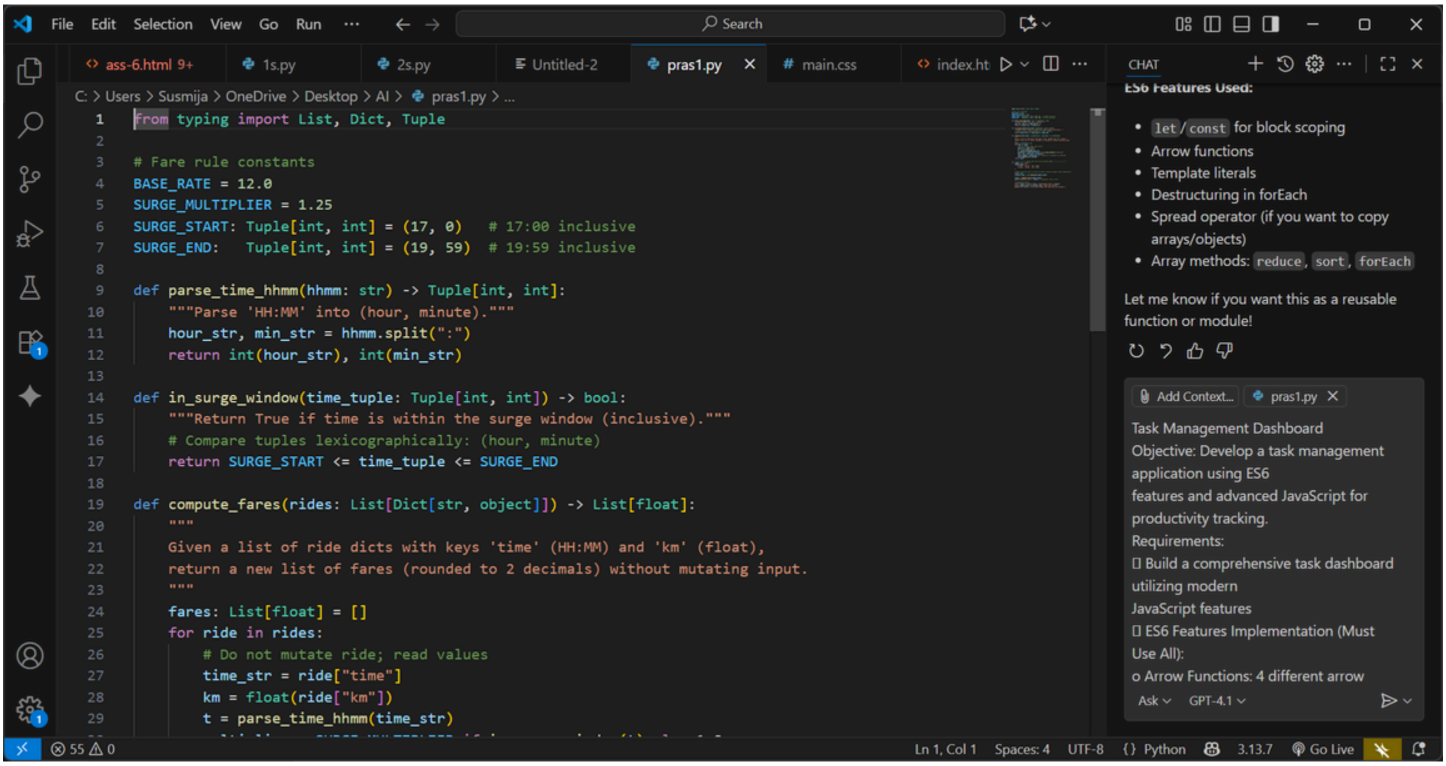
prompt AI to outline parsing HH:MM, applying conditionals, and rounding to 2 decimals; then implement and write a quick test.

Constraints & Notes: No external libraries; round each fare to 2 decimals; do not mutate input.

Sample Input

[{'time': '07:45', 'km': 2.8}, {'time': '18:45', 'km': 6.2}]

code Generated:



File Edit Selection View Go Run ··· Search
ass-6.html 9+
1s.py
2s.py
Untitled-2
pras1.py
main.css
index.ht
C: > Users > Susmija > OneDrive > Desktop > AI > pras1.py > ...
CHAT
Ln 1, Col 1 Spaces: 4 UTF-8 {} Python 3.13.7 Go Live
55 0

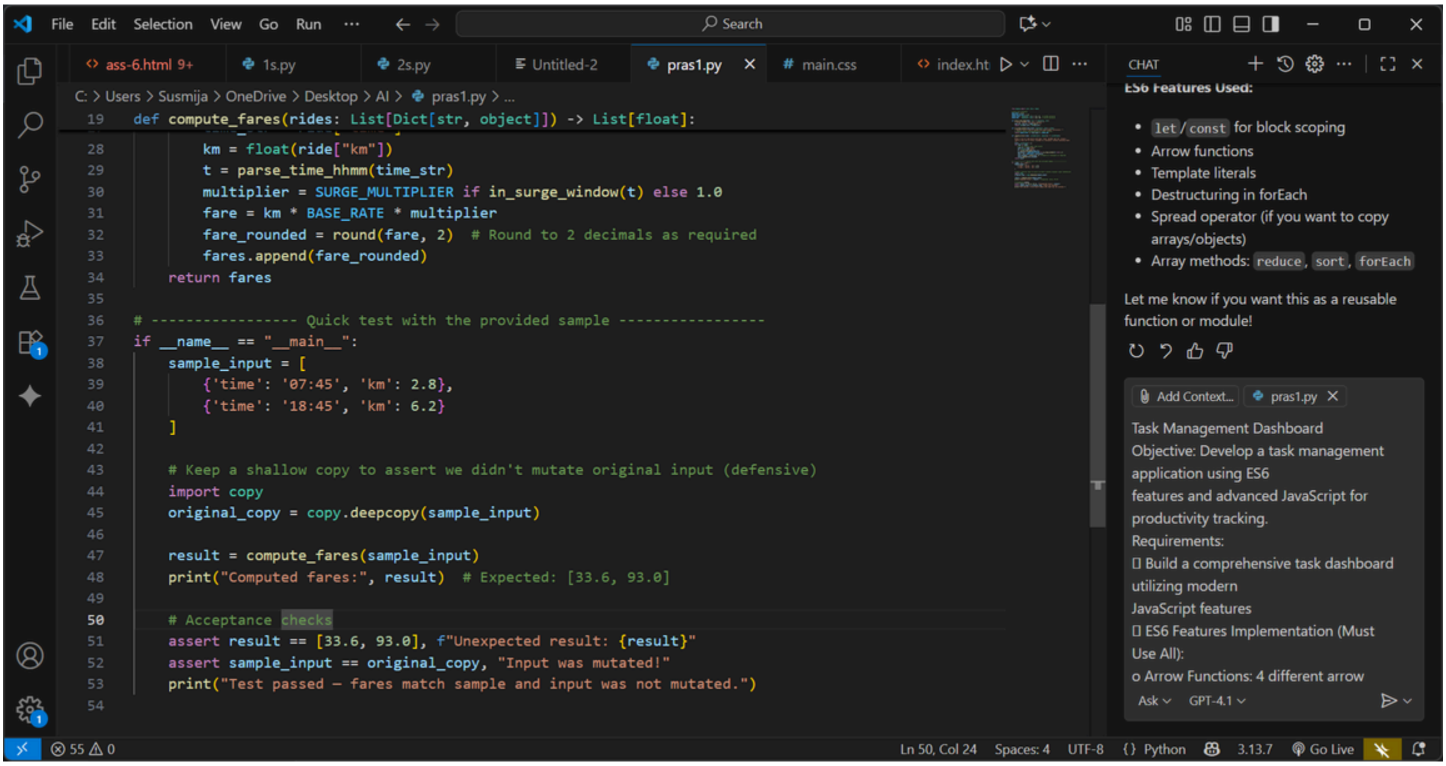
```python
from typing import List, Dict, Tuple

# Fare rule constants
BASE_RATE = 12.0
SURGE_MULTIPLIER = 1.25
SURGE_START: Tuple[int, int] = (17, 0)   # 17:00 inclusive
SURGE_END:   Tuple[int, int] = (19, 59)  # 19:59 inclusive

def parse_time_hhmm(hhmm: str) -> Tuple[int, int]:
    """Parse 'HH:MM' into (hour, minute)."""
    hour_str, min_str = hhmm.split(":")
    return int(hour_str), int(min_str)

def in_surge_window(time_tuple: Tuple[int, int]) -> bool:
    """Return True if time is within the surge window (inclusive)."""
    # Compare tuples lexicographically: (hour, minute)
    return SURGE_START <= time_tuple <= SURGE_END

def compute_fares(rides: List[Dict[str, object]]) -> List[float]:
    """
    Given a list of ride dicts with keys 'time' (HH:MM) and 'km' (float),
    return a new list of fares (rounded to 2 decimals) without mutating input.
    """
    fares: List[float] = []
    for ride in rides:
        # Do not mutate ride; read values
        time_str = ride["time"]
        km = float(ride["km"])
        t = parse_time_hhmm(time_str)
```

**ES6 Features Used:**

- `let`/`const` for block scoping
- Arrow functions
- Template literals
- Destructuring in forEach
- Spread operator (if you want to copy arrays/objects)
- Array methods: `reduce`, `sort`, `forEach`

Let me know if you want this as a reusable function or module!

Task Management Dashboard
Objective: Develop a task management application using ES6
features and advanced JavaScript for productivity tracking.
Requirements:
 Build a comprehensive task dashboard utilizing modern
JavaScript features
 ES6 Features Implementation (Must Use All):
o Arrow Functions: 4 different arrow



File Edit Selection View Go Run ··· Search
ass-6.html 9+
1s.py
2s.py
Untitled-2
pras1.py
main.css
index.ht
C: > Users > Susmija > OneDrive > Desktop > AI > pras1.py > ...
CHAT
Ln 50, Col 24 Spaces: 4 UTF-8 {} Python 3.13.7 Go Live
55 0

```python
def compute_fares(rides: List[Dict[str, object]]) -> List[float]:
        km = float(ride["km"])
        t = parse_time_hhmm(time_str)
        multiplier = SURGE_MULTIPLIER if in_surge_window(t) else 1.0
        fare = km * BASE_RATE * multiplier
        fare_rounded = round(fare, 2)  # Round to 2 decimals as required
        fares.append(fare_rounded)
    return fares

# ---------------- Quick test with the provided sample ----------------
if __name__ == "__main__":
    sample_input = [
        {'time': '07:45', 'km': 2.8},
        {'time': '18:45', 'km': 6.2}
    ]

    # Keep a shallow copy to assert we didn't mutate original input (defensive)
    import copy
    original_copy = copy.deepcopy(sample_input)

    result = compute_fares(sample_input)
    print("Computed fares:", result)  # Expected: [33.6, 93.0]

    # Acceptance checks
    assert result == [33.6, 93.0], f"Unexpected result: {result}"
    assert sample_input == original_copy, "Input was mutated!"
    print("Test passed — fares match sample and input was not mutated.")
```

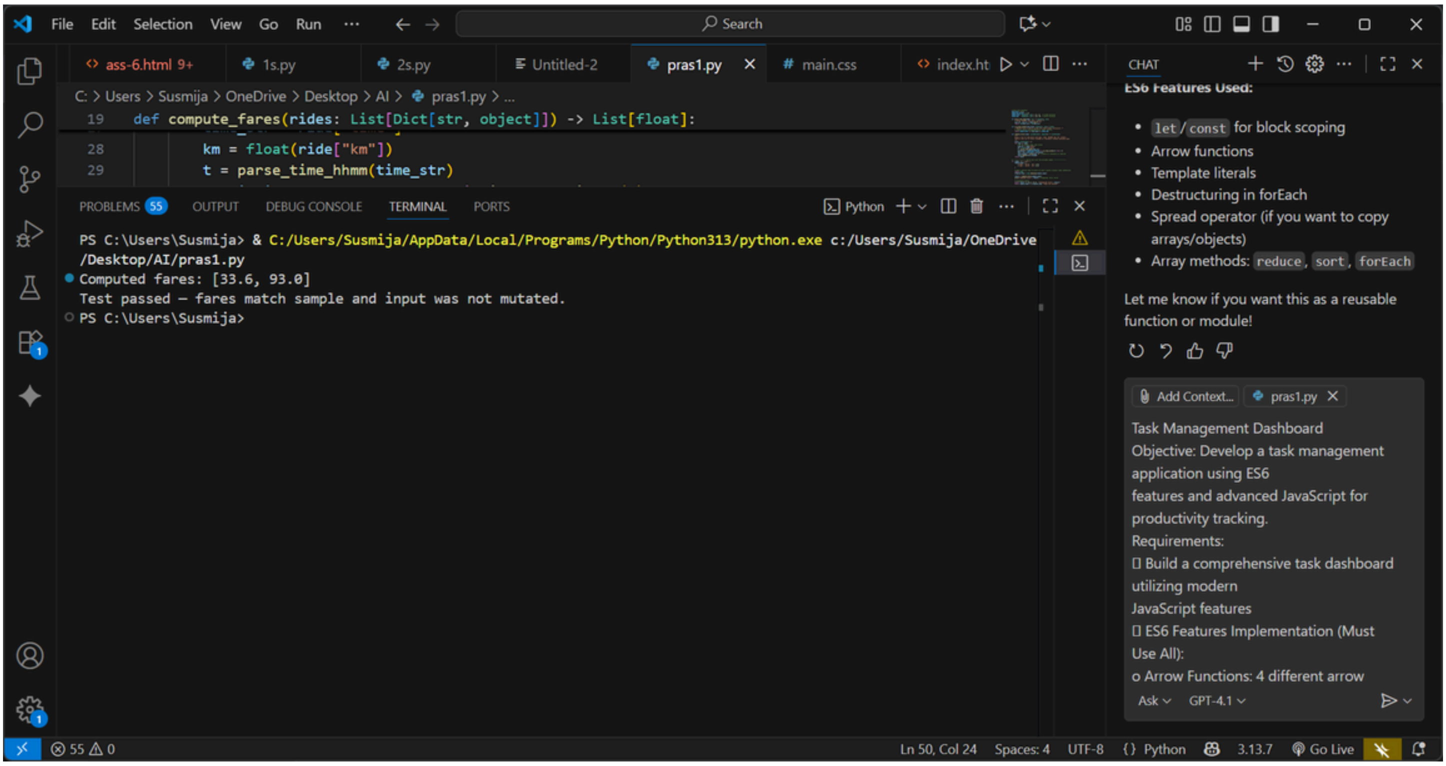
**ES6 Features Used:**

- `let`/`const` for block scoping
- Arrow functions
- Template literals
- Destructuring in forEach
- Spread operator (if you want to copy arrays/objects)
- Array methods: `reduce`, `sort`, `forEach`

Let me know if you want this as a reusable function or module!

Task Management Dashboard
Objective: Develop a task management application using ES6
features and advanced JavaScript for productivity tracking.
Requirements:
 Build a comprehensive task dashboard utilizing modern
JavaScript features
 ES6 Features Implementation (Must Use All):
o Arrow Functions: 4 different arrow

Output:



```
def compute_fares(rides: List[Dict[str, object]]) -> List[float]:
        km = float(ride["km"])
        t = parse_time_hhmm(time_str)
```

```
PS C:\Users\Susmija> & C:/Users/Susmija/AppData/Local/Programs/Python/Python313/python.exe c:/Users/Susmija/OneDrive
/Desktop/AI/pras1.py
Computed fares: [33.6, 93.0]
Test passed — fares match sample and input was not mutated.
PS C:\Users\Susmija>
```

**ES6 Features Used:**

- `let`/`const` for block scoping
- Arrow functions
- Template literals
- Destructuring in forEach
- Spread operator (if you want to copy arrays/objects)
- Array methods: `reduce`, `sort`, `forEach`

Let me know if you want this as a reusable function or module!

Task Management Dashboard
Objective: Develop a task management application using ES6
features and advanced JavaScript for productivity tracking.
Requirements:
 Build a comprehensive task dashboard utilizing modern
JavaScript features
 ES6 Features Implementation (Must Use All):
o Arrow Functions: 4 different arrow

## TASK-2:

Use AI to identify the bug and fix the window iteration so all valid windows are included.

**Prompt Given:**

Ask AI to add a failing test first, propose the minimal fix, and verify with the sample.

Constraints & Notes:

Guard invalid w (<=0 or >len(xs)); preserve O(n*w) simple solution.

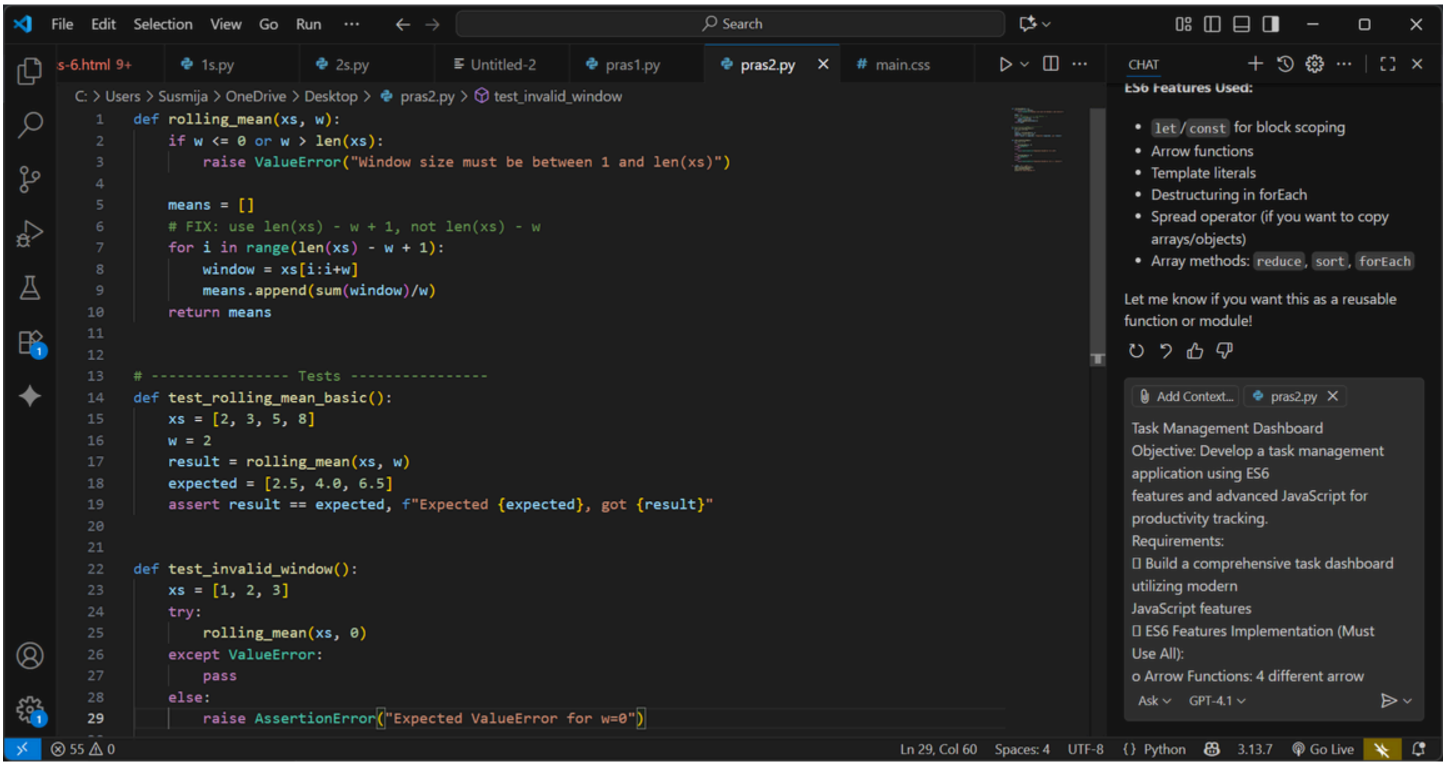
Sample Input

xs=[2, 3, 5, 8], w=2

Buggy code:

```
def rolling_mean(xs, w):

sums = []

for i in range(len(xs)-w):

window = xs[i:i+w]

sums.append(sum(window)/w)

return sums
```

## Code Generated:


File Edit Selection View Go Run ···
Search
s-6.html 9+
1s.py
2s.py
Untitled-2
pras1.py
pras2.py
main.css
C: > Users > Susmija > OneDrive > Desktop > pras2.py > test_invalid_window
CHAT
Ln 29, Col 60 Spaces: 4 UTF-8 {} Python 3.13.7 Go Live
⊗ 55 ⚠ 0

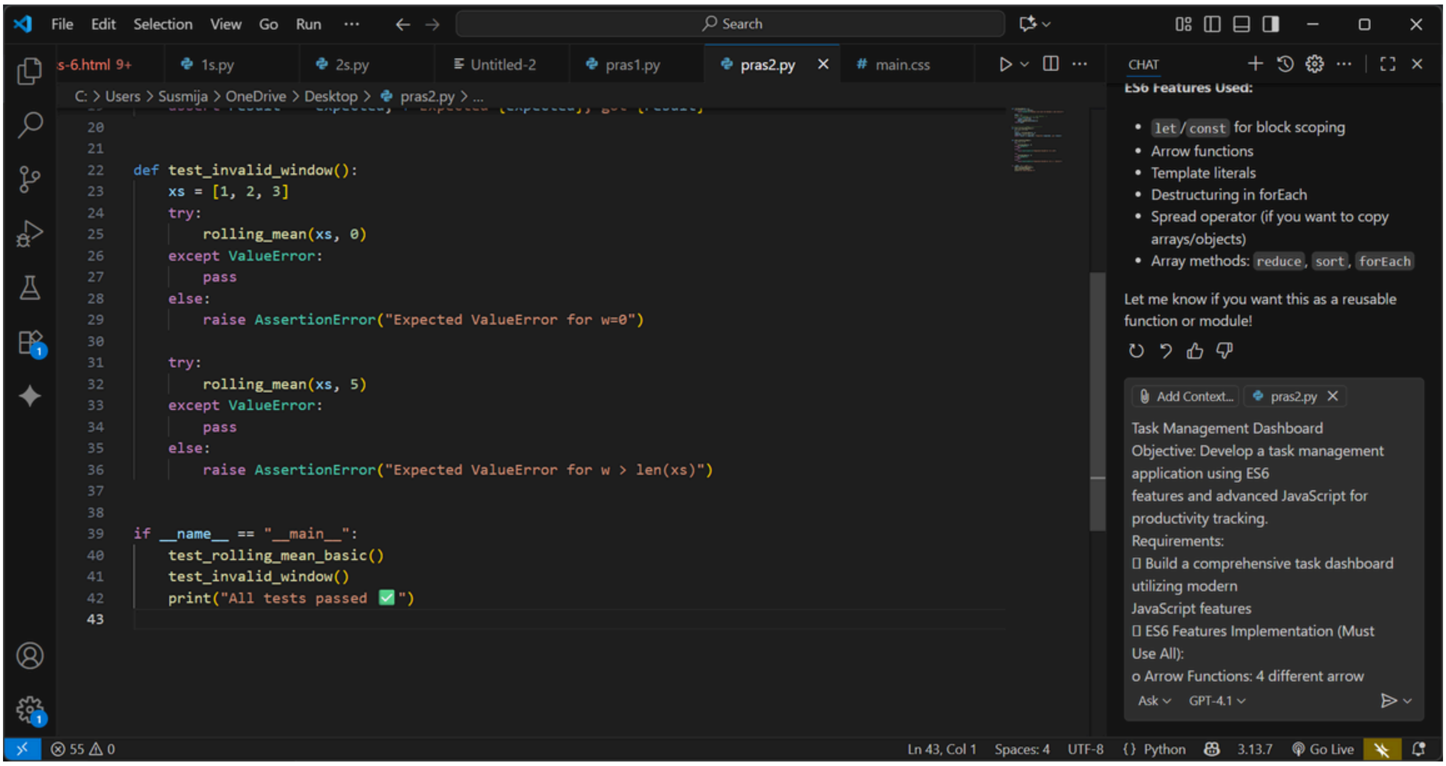
```python
def rolling_mean(xs, w):
    if w <= 0 or w > len(xs):
        raise ValueError("Window size must be between 1 and len(xs)")

    means = []
    # FIX: use len(xs) - w + 1, not len(xs) - w
    for i in range(len(xs) - w + 1):
        window = xs[i:i+w]
        means.append(sum(window)/w)
    return means


# --------------- Tests ---------------
def test_rolling_mean_basic():
    xs = [2, 3, 5, 8]
    w = 2
    result = rolling_mean(xs, w)
    expected = [2.5, 4.0, 6.5]
    assert result == expected, f"Expected {expected}, got {result}"


def test_invalid_window():
    xs = [1, 2, 3]
    try:
        rolling_mean(xs, 0)
    except ValueError:
        pass
    else:
        raise AssertionError("Expected ValueError for w=0")
```

**ES6 Features Used:**

- `let`/`const` for block scoping
- Arrow functions
- Template literals
- Destructuring in forEach
- Spread operator (if you want to copy arrays/objects)
- Array methods: `reduce`, `sort`, `forEach`

Let me know if you want this as a reusable function or module!

Task Management Dashboard
Objective: Develop a task management application using ES6
features and advanced JavaScript for productivity tracking.
Requirements:
 Build a comprehensive task dashboard utilizing modern
JavaScript features
 ES6 Features Implementation (Must Use All):
o Arrow Functions: 4 different arrow


File Edit Selection View Go Run ···
Search
s-6.html 9+
1s.py
2s.py
Untitled-2
pras1.py
pras2.py
main.css
C: > Users > Susmija > OneDrive > Desktop > pras2.py > ...
CHAT
Ln 43, Col 1 Spaces: 4 UTF-8 {} Python 3.13.7 Go Live
⊗ 55 ⚠ 0

```python


def test_invalid_window():
    xs = [1, 2, 3]
    try:
        rolling_mean(xs, 0)
    except ValueError:
        pass
    else:
        raise AssertionError("Expected ValueError for w=0")

    try:
        rolling_mean(xs, 5)
    except ValueError:
        pass
    else:
        raise AssertionError("Expected ValueError for w > len(xs)")


if __name__ == "__main__":
    test_rolling_mean_basic()
    test_invalid_window()
    print("All tests passed ✅")
```

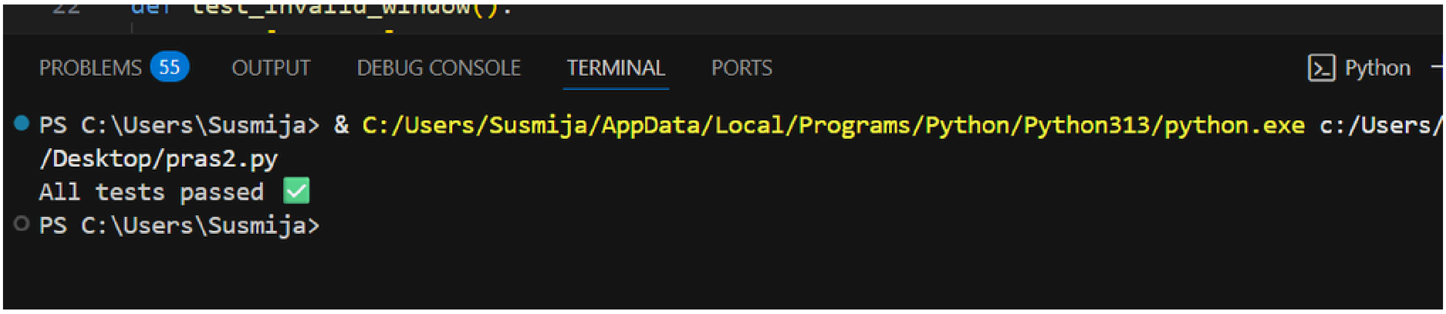
**ES6 Features Used:**

- `let`/`const` for block scoping
- Arrow functions
- Template literals
- Destructuring in forEach
- Spread operator (if you want to copy arrays/objects)
- Array methods: `reduce`, `sort`, `forEach`

Let me know if you want this as a reusable function or module!

Task Management Dashboard
Objective: Develop a task management application using ES6
features and advanced JavaScript for productivity tracking.
Requirements:
 Build a comprehensive task dashboard utilizing modern
JavaScript features
 ES6 Features Implementation (Must Use All):
o Arrow Functions: 4 different arrow

Output:



```
● PS C:\Users\Susmija> & C:/Users/Susmija/AppData/Local/Programs/Python/Python313/python.exe c:/Users/
  /Desktop/pras2.py
  All tests passed ✅
○ PS C:\Users\Susmija>
```