✅ Important Points and summary – Day 01

🔷 Key Logic Takeaways

- Always calculate `length` manually before looping.
- Use `float('-inf')` and `float('inf')` as starting points for min/max comparisons.
- Avoid using:
  - `in`
  - `len()`
  - `set()`
  - Slicing or list comprehensions

    (👉 Simulate all using loops)
- Copying into a new array avoids in-place modification side-effects.
- Use `(i + 1) % length` for **cyclic right shift** logic.
- Avoid breaking loops unless you're sure you've completed all checks.

🔷 Sorting Tips

- **Bubble Sort**: Makes the largest element bubble to the end of the array in each pass.
- **Selection Sort**: Finds the minimum and places it at the current index.

🧠 Manual Habits Practiced Today

- Writing loops from scratch
- Simulating common array behaviors without built-ins

- Understanding brute force time complexity using nested loops

🔁 Useful Code Snippets (Python)

```
python
CopyEdit
# Calculate length manually
length = 0
for _ in arr:
    length += 1

# Right Rotate (cyclic)
right_rot_arr[(i + 1) % length] = arr[i]

# Swap two elements
arr[i], arr[j] = arr[j], arr[i]
```

🧠 Summary – Day 01

✅ What I Did

- Practiced **array operations** from scratch without Python shortcuts
- Built core logic for:
    - Insertion and deletion at index
    - Min / Max / Second-Min / Second-Max
    - Rotation (left/right), Merge, Reverse
    - Frequency counting without dictionary
    - Removing duplicates manually
- Implemented:

    - ✅ **Bubble Sort**
    - ✅ **Selection Sort**

🚀 LeetCode Problems Solved (Brute Force Only)

Problem	Status
Two Sum	✅ Done
Maximum Subarray	✅ Done
Merge Sorted Array	✅ Done
Remove Duplicates from Sorted Array	✅ Done
Contains Duplicate	✅ Done

🔁 For Revision Later

- Sorting with swap counters
- Edge cases in `remove duplicates`
- Understand differences between brute-force and optimized logic