ChatGPT

Folder: `Day2/`

`contains_duplicate.py`

Problem: Contains Duplicate – LeetCode 217

📚 Concepts:

- Python `set()` for uniqueness
- Membership testing: `if num in set`
- Detecting duplicates efficiently

Logic:

```
1. Create an empty set `seen`.
2. For each number in the list:
   - If number is already in `seen`, return True.
   - Else add it to the set.
3. If loop completes, return False.
```

⏳ Time & Space:

- Time: `O(n)`
- Space: `O(n)`

`group_anagrams.py`

Problem: Group Anagrams – LeetCode 49

📚 Concepts:

- `defaultdict(list)` from `collections`
- Sorting strings and using them as tuple keys
- Grouping logic using dictionaries

Logic:

```
1. Initialize a defaultdict of lists.
2. For each word in the input:
   - Sort the word and convert to a tuple => use it as a key.
```

```
   - Append the original word to the list at that key.
3. Return list of all values from the dictionary.
```

⏳ **Time & Space:**

- Time: `O(N * K log K)` (N = words, K = avg length of word)
- Space: `O(NK)`

`merge_two_sorted_lists.py`

Problem: Merge Two Sorted Lists – LeetCode 21

📚 **Concepts:**

- Linked list traversal
- Dummy node technique
- Merging two sorted sequences

Logic:

```
1. Use a dummy node to simplify edge cases.
2. Compare nodes from both lists:
   - Append the smaller one to the merged list.
   - Move the pointer ahead in that list.
3. When one list ends, link the remaining nodes of the other.
4. Return dummy.next (head of the merged list).
```

⏳ **Time & Space:**

- Time: `O(n + m)` (n and m are the lengths of the two lists)
- Space: `O(1)` (in-place merge)

Folder Summary:

```
Day2/
├── contains_duplicate.py       # Set-based duplicate detection
├── group_anagrams.py           # Dictionary-based anagram grouping
└── merge_two_sorted_lists.py   # Linked list merge using dummy node
```
