**Phase-3 training problems**

**Day-1 (Basic Recursion Problems):**

1. Discussed about Time and Space complexity
2. Started off with Recursion concept (Recursive call stack diagram)
3. Printing from 1 to N (Increasing sequence)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_printing_from_1_to_n.py)

1. Printing from N to 1 (Decreasing sequence)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_printing_from_n_to_1.py)

1. Printing even numbers from 1 to N

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_printing_even_numbers_1_to_n.py)

1. Printing Odd numbers from 1 to N

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_printing_odd_numbers_1_to_n.py)

1. Finding sum of list of integers using recursion

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_finding_sum_of_list_of_integers.py)

1. Finding Maximum element from a list of integers using recursion

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_finding_max_ele_in_a_list.py)

1. Finding Minimum element from a list of integers using recursion

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_finding_min_element_from_list.py)

**Practice problems on recursion:**

1. Find whether given string is a palindrome or not using recursion

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_check_for_palindrome.py)

1. Check whether 2 given strings are the same or not.

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_check_two_strings_same_or_not.py)

1. Check whether one string is substring of the other or not using recursion.

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_check_for_substring.py)

1. Check whether one string is a subsequence of the other or not using recursion.

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_check_for_subsequence.py)

1. Starting from (n-1, n-1), reach till (0, 0) in a n \* n square matrix, and print all the elements using recursion (similar to sudoku solver)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_print_2d_matrix.py)

1. Find all the digits sum within a given number using recursion.

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_digit_sum.py)

1. Finding pow(x, n) using recursion

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_find_pow.py)

1. Display an array (both from left to right as well as right to left)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_printing_array_in_both_directions.py)

1. Find the first occurrence of a target element within a given array.

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_find_first_occurrence.py)

\

1. Find the last occurrence of a target element within a given array.

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_find_last_occurrence.py)

1. Find the total frequency of a target element within a given array.

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_find_all_occurrences_of_target.py)

1. Find whether there is a sum of few elements matching with a given target value or not. Print “Found” or “Not Found”.

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/0_check_for_target_sum_in_a_list.py)

**Day-2 (Intermediate Recursion and Backtracking Problems):**

1. [Generating valid parenthesis problem](https://leetcode.com/problems/generate-parentheses/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/GeneratingValidParenthesis.py)

1. [Mobile Keypad Problem](https://leetcode.com/problems/letter-combinations-of-a-phone-number/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/letter_combinations_of_a_phone_number.py)

1. [Rat in a maze problem](https://www.geeksforgeeks.org/problems/rat-in-a-maze-problem/1)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/rat_in_a_maze.py)

1. [Palindromic partitioning](https://leetcode.com/problems/palindrome-partitioning/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/palindrome_partitioning.py)

1. [Word break](https://leetcode.com/problems/word-break/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/word_break.py)

**Day-3 (Advanced Recursion and Backtracking Problems):**

1. [Tower of Hanoi problem](https://www.geeksforgeeks.org/problems/tower-of-hanoi-1587115621/1)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/tower_of_hanoi.py)

1. [Sudoku solver problem](https://leetcode.com/problems/sudoku-solver/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/sudoku_solver.py)

1. [N Queens problem](https://leetcode.com/problems/n-queens/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/n_queens.py)

1. [Kinghts Tour](https://www.geeksforgeeks.org/problems/knight-walk4521/1)

* Solution link (can submit this problem, while discussing Graphs concept)

**Day-4**:

1. Rajeev sir discussed some problems on trees and graphs

**Day-5 (1- D Dynamic Programming Problems):**

1. Started off with Dynamic programming, initially took [fibonacci series problem](https://www.geeksforgeeks.org/problems/introduction-to-dp/1?utm_source=youtube&utm_medium=collab_striver_ytdescription&utm_campaign=introduction-to-dp)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/introduction_to_dp.py)

1. [Climbing stairs](https://leetcode.com/problems/climbing-stairs/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/climbing_stairs.py)

1. [Min Cost Climbing stairs](https://leetcode.com/problems/min-cost-climbing-stairs/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/min_cost_climbing_stairs.py)

1. [House Robber](https://leetcode.com/problems/house-robber/description/?envType=daily-question&envId=2024-01-21)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/house_robber_I.py)

1. [House Robber - II](https://leetcode.com/problems/house-robber-ii/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/house_robber_II.py)

1. [Minimal Cost](https://www.geeksforgeeks.org/problems/minimal-cost/1?utm_source=youtube&utm_medium=collab_striver_ytdescription&utm_campaign=minimal-cost)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/minimal_cost.py)

**Day-6 (2 - D Dynamic Programming Problems)**:

1. [Geek’s Training](https://www.geeksforgeeks.org/problems/geeks-training/1?utm_source=youtube&utm_medium=collab_striver_ytdescription&utm_campaign=geeks-training)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/15_geeks_training.py)

1. [Unique paths - I](https://leetcode.com/problems/unique-paths/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/16_unique_paths_I.py)

1. [Unique paths - II](https://leetcode.com/problems/unique-paths-ii/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/17_unique_paths_II.py)

1. [Minimum path sum](https://leetcode.com/problems/minimum-path-sum/description/)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/18_minimum_path_sum.py)

1. [0-1 Knapsack](https://www.geeksforgeeks.org/problems/0-1-knapsack-problem0945/1)

* [Solution link](https://github.com/ramsai-5A1/Super_100_Batch_Codes/blob/main/19_0_1_knapsack.py)

**Day-7 (3 - D Dynamic Programming Problems)**:

**Day-8 ( ):**

**Day-9 ( ):**

**Day-10 ( ):**

**Day-11 ( ):**

**Day-12 ( ):**

**Day-13 ( ):**

**Day-14 ( ):**

**Day-15 ( ):**

**Day-16 ( ):**

**Day-17 ( ):**

**Day-18 ( ):**

**The End**