Competitive Programming (SoC'25)

Project id - 22
Mentor - Himanshu Shete(23B0770)

Week 3 : Week 3: Linked Lists, OOP (Classes, Inheritance), Pointers and Memory Concepts

Theory:

(these are just resources you can always learn from youtube or other sources)

1. OOP
   a. https://www.geeksforgeeks.org/object-oriented-programming-in-cpp/
   b. Inheritance- https://www.geeksforgeeks.org/inheritance-in-c/
   c. Polymorphism- https://www.geeksforgeeks.org/cpp-polymorphism/
   d. Constructors and destructors:
      i. https://www.geeksforgeeks.org/constructors-c/
      ii. https://www.geeksforgeeks.org/difference-between-constructor-and-destructor-in-c/.
      iii. Uses of constructors can be custom structs, objects in segment trees, tries, graphs
      iv. destructors aren't needed, most memory management is handled by STL containers (vector, map, etc.), which have built-in destructors
2. Pointers & Memory
   a. https://www.geeksforgeeks.org/c-pointers/
   b. https://cplusplus.com/doc/tutorial/pointers/
   c. https://www.geeksforgeeks.org/new-and-delete-operators-in-cpp-for-dynamic-memory/
3. Linked Lists
   a. https://www.programiz.com/dsa/linked-list
   b. (singly, doubly, circular, operations) https://www.geeksforgeeks.org/linked-list-data-structure/
   c. https://www.geeksforgeeks.org/program-to-implement-singly-linked-list-in-c-using-class/

Problems:

(increasing difficulty, maintain a git repo)
(this week isn’t problem heavy, continue doing number theory practice)

1. Linked lists
   a. https://leetcode.com/problems/reverse-linked-list/description/
   b. https://leetcode.com/problems/linked-list-cycle/description/
   c. https://leetcode.com/problems/middle-of-the-linked-list/description/
   d. https://leetcode.com/problems/merge-two-sorted-lists/description/
   e. https://leetcode.com/problems/add-two-numbers/description/