Status:
Treat this as a single-shot optimization task. Minimize stress while ensuring the generated closed (all nodes connected) 2D Truss remains within the constraints.

Inputs:
- Node_dict: Given node position: ***{node_dict}***
- Loads: Forces acting on node (node_id: (magnitude, direction)): ***{load}***
- Supports: Support information ***{supports}***
- area_id: Area ID values to choose cross section from : ***{area_id}***

Objectives:
1. Generate optimal closed truss structure that minimizes stress, while satisfying all given constraints.

Constraints:
1. Max stress (compressive(-) and tensile(+)) <= ±***{max_stress}***
2. Total mass: Mass (sum of member length × area from ***{area_id}***) <= ***{max_weight}***

Instructions:
1. Treat this as a single-shot optimization task. No prior designs exist.
2. Design a fully connected (closed) 2D truss — all nodes must be part of a single closed connected structure.
3. Your goal is to minimize the stress while strictly satisfying all constraints.
4. Use only the given nodes, loads, supports, and area IDs. Do not modify load and support node positions.
5. Generate new nodes (if needed) with precise float values, Do not modify load and support node positions.
6. Choose area_id values only from ***{area_id}*** using string keys.
7. Avoid duplicate nodes or member connections. All members must reference existing, unique nodes.
8. Use your understanding of structural mechanics — infer effective strategies (e.g., triangulation, symmetry, direct load paths) to create most optimal structure.
9. Estimate stress and mass using basic structural principles (e.g., static equilibrium, trigonometry). Clearly state any assumptions made.
10. Output must be a complete, valid Python dictionary. All entries must be internally consistent and ready for parsing.

Output format (python dictionary):
Return a single valid Python dictionary in the format below. Keep all string fields concise. Use floats for coordinates.

```
{{
    "preamble" :  "Briefly describe the problem, current design status, and the optimization objective.",

    "scratch_pad_1": "Initial interpretation of given structure objectives and constraints. This is the first iteration",

    "analysis": "Detailed analysis of the structural problem based on inputs. Include interpretation of load paths, boundary conditions, possible failure modes, and efficiency of current design",

    "scratch_pad_2": "Reflect on the analysis. Note any patterns, areas of concern, or insights that may guide the design.",

    "reasoning": "Use design principles (e.g., symmetry, triangulation, load distribution) to justify the design direction. Propose structural changes or node repositions to improve performance.",

    "scratch_pad_3": "Outline key assumptions, set-up equations (e.g., force balance, mass calculation, trigonometric equations etc.), and calculations to be performed. Include considerations for member selection or node movement.",

    "calculation": ""Perform required calculations such as updated member forces, stress, and structure mass. Quantify constraint violations if any, and iterate on modifications if needed",
    "scratch_pad_final": "Final scratch pad before you generate the structure verify adherence to objective, constraints and Instructions. Perform modifications to ensure the design meets all requirements.",

    "node_dict": {{
        # Node coordinates must be precise floats based on structural calculations.
        # Each node entry should include design reason,  that will help future optimization.

        "node_1": (x, y), # thought, reason, design choice, calculation
        "node_2": (x, y), # thought, reason, design choice, calculation
    }},
    "member_dict": {example_member},
}}
```