Problem 1 (10 pts)

Construct an example to show that the following Greedy algorithm for the interval partitioning problem can allocate more than depth many classrooms (so it is not optimum): Sort the lectures based on their **finishing time**. When considering the next job, allocate it to the available classroom with the smallest index. If no classroom is available, allocate a new classroom.
Note: An example for the interval partitioning problem includes starting and ending time for all lectures. You need to show that by applying the greedy algorithm mentioned in the problem over the example will allocate more classrooms than the depth.

Problem 2 (10 pts)

Suppose you are given a connected graph G, with edge costs that are all distinct. Prove that G has a unique minimum spanning tree.

Problem 3 (10 pts)

Prove or disprove the following: Given any undirected graph G with weighted edges, and a minimum spanning tree T for that G, there exists some sorting of the edge weights $w(e_1) \leq w(e_2) \leq \cdots \leq w(e_m)$, such that running Kruskal's algorithm with that sorting produces the tree T.

Problem 4 (10 pts)

Given a sequence of n real numbers $a_1, \cdots, a_n$ where n is even, design a polynomial time algorithm to partition these numbers into $n/2$ pairs in the following way: For each pair we compute the sum of its numbers. Denote $s_1, \cdots, s_{n/2}$ these $n/2$ sums. Your algorithm should find the partition which minimizes the maximum sum. For example, given numbers 3, -1, 2, 7 you should output the following partition: $\{3, 2\}, \{-1, 7\}$. In such a case the maximum sum is $7 + (-1) = 6$.
Note: Give the pseudo-code for your algorithm and analyze its time complexity to show that it has polynomial time complexity.

Problem 5 (10 pts)

Given two edge disjoint spanning trees T_1, T_2 (**NOT** necessarily minimum) on n vertices, prove that for every edge $e \in T_1$ there exists an edge $f \in T_2$ that satisfies both of the following criteria:

- $T_1 - e + f$ is a spanning tree (on n vertices).

- $T_2 - f + e$ is a spanning tree (on n vertices).

Note: Two edge-disjoint spanning trees in a graph G are two spanning trees T_1 and T_2 such that they share **NO** common edges.

Problem 6 (Bonus 10 pts)

Solve the following problem with a program which should use one of the greedy algorithms we discussed. Please submit your **source code** to Canvas and attach a **screenshot** of the running output of your code here.

There are n houses in a village. We want to supply water for all the houses by building wells and laying pipes.

For each house i, we can either build a well inside it directly with cost wells$[i-1]$ (note the -1 due to 0-indexing), or pipe in water from another well to it. The costs to lay pipes between houses are given by the array pipes where each pipes$[j] = [\text{house1}_j, \text{house2}_j, \text{cost}_j]$ represents the cost to connect house1_j and house2_j together using a pipe. Connections are bidirectional, and there could be multiple valid connections between the same two houses with different costs.

Return the minimum total cost to supply water to all houses.

The function template:

minCostSupplyWater(n: int, wells: List[int], pipes: List[List[int]]) → int

Test your code with the following cases:

Testing Case 1

- **Input**: n = 3, wells = [1,2,2], pipes = [[1,2,1],[2,3,1]]
- **Output**: 3

Testing Case 2

- **Input**: n = 4, wells = [1, 2, 2, 1], pipes = [[1, 2, 1], [1, 2, 3], [2, 3, 2], [3, 4, 3], [1, 4, 2]]
- **Output**: 5

Testing Case 3

- **Input**: n = 5, wells = [10, 2, 2, 10, 2], pipes = [1, 2, 1], [2, 3, 1], [3, 4, 1], [1, 4, 2], [2, 5, 2]]
- **Output**: 7