Ishraq Kamal Adib - 22299325

Task 1

This code optimizes the Friend-Circle problem solution using Union-find. The find function implements path compression and the union function avoids calling "find" for the root node.

Task 2

We are using Kruskals algo. The edge are sorted by maintenance cost, and a Union-Find object "inf" is initialized with "n" cities. The code then iterates over the ~~sorted~~ sorted edges, checking if the endpoints at each edge belong to different sets in "inf". If they do, the edge weight is added to "mst_cost", a union operation is performed on

the endpoints of and the edge is added to mst_edges. After iterating over all edges, mst_cost will contain the minimum total maintenance cost to connect all cities.

Task 3

The code reads the number of stairs "n" from input file, uses count function to count distinct ways to climb using dynamic prog. where dp[i] represents the number of ways to climb "i" stairs. The base cases are dp[0] = dp[1] = 1 and dp[2] = 2. For each stair step, it calculates dp[i] as the sum of dp[i-1] and dp[i-2], representing the option to climb either one or two

rtain at a time.

Task 4

This code implements a dynamic prog. solution to find the minimum number of coins to make up a target amount. First initialize a table dp with size amount+1 and sets the dp[0] to 0. For each target amount from 1 to amount; it iterates over each coin denomination and updates dp[target] if the coin is less than or equal to target. After filling the dp table, it returns dp[amount] if its not infinity, indicating the target amount can be made up using the given coin denominations, otherwise, it returns -1