Task 5: Greedy approach

This task is about how to apply greed to the problem of generating a graph for bordering enclosures.

Part A: Chosen approach (5 marks)

Prepare a written discussion of the manner in which your intent to apply greed to this problem. Be sure to respond to the following in your discussion

1. what aspect of your approach is greedy
2. how you know that it will generate a valid graph.
3. any limitations of your approach

In the graph, you get to choose vertex or edge and put it in the tree if it's not in the tree. After that, you have to choose another vertex or edge with minimum or maximum distance between tree and if it is not in the tree then adds that edge or vertex in the tree. So, the process should be repeating again and again until last node. So, we used greedy approach here as we used minimum or maximum distance to add second node or edge.

With that done adding vertex or edges, it would create graph for sure. It is understandable that graph would not be guaranteed optimal because it would keep adding nodes or edges until it reaches last one so it would create the graph which would be valid.

A greedy algorithm, as the name suggests, always makes the choice that seems to be the best at that moment.

Assume that you have a **function** that needs to be optimized (either maximized or minimized) at a given point. A Greedy algorithm makes greedy choices at each step to ensure that the function is optimized. The Greedy algorithm has only one shot to compute the optimal solution so that **it never goes back and reverses the decision**.

We can make whatever choice seems best at the moment and then solve the subproblems that arise later. The choice made by a greedy algorithm may depend on choices made so far, but not on future choices or all the solutions to the subproblem. It iteratively makes one greedy choice after another, reducing each given problem into a smaller one. In other words, a greedy algorithm never reconsiders its choices. This is the main difference from backtracking approach, which is guaranteed to find the optimal solution.

For example, in knapsack problem, greedily choose items based on their $ value by choosing items with the HIGHEST value first.

Hence, Greedy approach is not optimum approach in every case, but it may be optimum sometimes.