For task 1:

There is two structs in the program which are “node” and “edge”. Node contains its x, y axes location, energy, id, leader and also some boolean variables elected, remaining, dead and visited. Edge only contains its start and end terminals.

After the input stage, every node will measure the distance to all other nodes.

If this is first level or the merging time of last level is more than 0, then output the base-station of last/first level, else stop the loop.

Create a vector container---low_cost[i] to store the shortest length of edge which links to the component which contains node i. Create a vector container link_to[i] to store the start and end node of this edge.

Then, select two different nodes:

If they belongs to different components and then compare the distance of two nodes with corresponding low_cost[i]

Different leaders represent different components of the network and all nodes in same component have same leader.

If the distance is smaller than low_cost[i], record the start and end terminals of this edge and update the low_cost[i]

If there are edges found between two components, merge these two components and change the leader of the component which has "smaller leader" to the "bigger leader"

If merging time is more than 0, then output the newly elected leaders. Sometimes, multiple parts will be merged into one part, so only output the nodes which have been elected and still remaining(alive).

For Task 2:

What I have done: Every time there is a node down/dead/run out of energy, rebuild a MST for the remaining nodes and every edge is going to/from this specific dead node becomes 10(which means invalid/unconnected in this network). Use breadth first search(BFS) to simulate the process of broadcast.

However, this algorithm just ensures the total energy consumption is least. Considering the case that there is a “strong” node which has infinite energy, broadcast with the route of MST may be wrong. The infinite node should take more responsibility to broadcast, although it will greatly increase the total energy consumption. Therefore, using some heuristic functions to evaluate every active node in every broadcast and make the energy allocated averagely in order to broadcast with the route of MST as far as possible will be a remarkable solution for task 2.