A - Back to Underworld:

There are 2 types of fighter vampires and lykans. A vampire and a lykan will take part in a duel and the winner will fight another fighter of the opposite race.

But we don't know which one is vampire and which one is lykan. So we can randomly assign one to vampire and all the fighters he fought to lykan (or vice versa) using a BFS. After each call of BFS we should update the result.

B - Kefa and Park:

We can traverse the whole graph using DFS. Keep track of the consecutive nodes with cats if the number becomes greater than mxcat we can return from that path.

While traversing if we reach to a leaf node we should increase the value of ans. (As the restaurants are located at the leaf node)

C - Oil Deposits:

Again a simple DFS problem.

Traverse all the cells using nested loop. If we encounter a "@" and that cell is not visited before call DFS for that cell.

The ans is the number of times DFS is called.

D - Travelling cost:

As the graph is a weighted one we need Dijkstra to solve this.

First all the nodes distant is set to Infininity, then we can call Dijkstra for the source node updating the distant of all reachable nodes.

The ans is “No path” id distant of a node is still Infinity else dist[node].

THE END