Approach:

I took a object oriented approach and created 3 structures. The ant which is a basic structure that holds its name, the number of moves, and its current location, and the ability to move to another location through a tunnel. The colony which holds all of the world map information as well as the information about the ants that are currently in the colony. And the hive which holds all of the logic of creating and adding new ants, destroying ants, creating colonies, destroying colonies, and a simulate function. The simulate function moves all ants at the same time, then destroys all necessary colonies.

Assumptions:

1. Data format is consistent and correct, no overlapping names
2. The end result map may be different with each run and with N number of ants.
3. “Each iteration” = All ants move randomly 1 tunnel space at the same time, therefore there can be multiple ants in a single colony (all ants die, not just pairs)
4. All ants move if possible
5. All ants are spawned in random locations
6. We don’t care about trapped ants, so we end their suffering sooner

Problems:

1. Syntax related issues:
    a. Borrowing as mut for a HashMap to change the object value
    b. Properly using references for keeping track of ants
2. Structure related issues:
    a. How much control do I want to give the hive, colony or ant?
    b. How do I move ants efficiently?
3. Spent too much time worrying about optimizations
    a. Trying to write too much “clean code” instead of implementing functionality and moving on

Notes

Intuitions:

1. What is the best way to structure the map
    a. Struct(Hashmap(int:id)->ants, Hashmap(String) -> Struct(String:LocationName, Vec:(ants), Hashmap(String)->(String:Locations)))
2. What does an ant look like?
    a. Struct(int: id, int: move_counter)
3. Multithread the simulation work?
4. How to retain data format?
    a. Hashmap(location:String, Hashmap(location:String, cardinal_direction: uint8))