1-a)

Example solution representation:
[6, 0, 7, 8, 0, 0, 5, 0, 4, 0, 3, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

When iterating through the solution, we find one of two alternatives. If our next value is an integer greater than 0, we know that the ship we are currently considering is moving to the node with the index of our next value.

If our next value is zero, however, we must check whether we are currently considering the parent ship, or a dispatched ship. If we are currently considering the parent ship, we dispatch a new ship, set the ship's home node to be the parent ship's current location, and set the ship we're currently considering to be the dispatched ship. If we are currently considering a dispatched ship, we instruct the ship to return to its home node, and set the ship we're currently considering to be the parent ship.

From this we can extract a few invariants.

1. Each node to be visited must be found in our solution array exactly once.
2. The number of zeros found in our solution must be equal to two times the number of nodes in our problem.

By representing the solution this way we cover any feasible or infeasible solution, from the extreme case where our parent ship visits all nodes to the extreme case where every node is visited by a separate dispatch ship.

The solution representation differentiates between which ship is dispatched first, whenever more than one ship is dispatched from the same node. This means that many solutions can be represented by more than one representation.

The solution 'dispatches and returns' a series of ships after visiting all nodes in the problem. This allows us to dispatch new ships to perform deliveries without adjusting the length of our solution array.

1-b)

Example solution representation:
[0, -1, 11, 9, -1, 10, -2, 12, 9, -2, 9, 8, 7, -2, 4, 3, -2, 3, -2, 3, 5, -2, -1, 1, 5, -1, 5, 6]

When iterating through our solution array, we keep track of which entity we are currently considering. We start off considering our main truck with entity index 0. If whenever we're considering our main truck we find that the next value in our array is negative, we switch to considering the entity with index equal to the absolute value of the negative value (e.g. if the next value in our array is -1, we switch to consider entity 1).

If the next value in our array is nonnegative, we move the entity we are currently considering to the node corresponding to the next value.

From this we can extract a few invariants.

1. Every node in the problem must be found in the array at least once.
2. The number of negative values in the array must be even.
3. Any negative value found in the array must be followed by the same negative value before another negative value appears.

The representation is entirely unambiguous, and can only be understood by a single solution to the problem.

The representation differentiates between which entity was dispatched from the main truck first, if more than one entity is dispatched from the same node. This means one solution is represented by more than one representation.

The representation does not have a fixed size.