i. The automaton for gfsa5_count is made to calculate the number of possible paths that can be produced for a string. By assigning each transition a weight of 1.0 as well as setting the weight of the start and ending states with weight of 1.0, it makes sure that each distinct path will add 1.0 to the total count of paths. As the automaton works, it multiplies the weights along with each transition, then whenever it finds distinct paths, it will sum up the different path counts using gen_or in the function f, where gen_or is defined as addition for type Double. The final result will then be the total number of paths a string could have.

j. The automaton for gfsa5_paths is made to calculate the list of possible state sequences for a string. Each transition is assigned with a list containing the current state number, the starting state as an empty list, and the ending state with the ending state number ([[1]]). As the automaton works, it makes sure that it will concatenate the current state's list with previous lists along one path. Then when there are multiple paths for a string, it combines the lists from all the different paths into a single list (list within list), which returns a union of all possible path lists for that string. The final result will then be a list of possible state sequences for a string.

k. No, it is not possible to assign a weight of 1 to all strings whose second half is an exact reversal of the first half, and assign a weight of 0.5 to all strings that do not conform to this pattern. The reason why it wouldn't be possible is because in a "multiplication-max" semiring, the weights are multiplied along the paths and the maximum weight is taken across all paths, therefore it can't check whether the second half of a string matches the first half in reverse. Because of this, the FSA in the "multiplication-max" semiring wouldn't be able to assign these weights as described.