Food Magnate Simulation

Additional Programming Tasks (Extension)

These challenges are presented without solutions, and offer to further explore and understand the skeleton program.

1. Add validation for any user input to ensure that something is entered and a message displayed if nothing is entered. This should include validation according to type, such that inputs required to be numeric are, indeed, numeric
2. Create an additional category of restaurant, with a new company of that type created as part of the default companies
3. Change the program so that all of the hard-coded values such as “fast food” are constants within an appropriate class.
4. Create a random instance of a company or an outlet running a promotion, during which time its expenses go up but its reputation score also rises and it may receive additional visits. Produce a report on the success/failure of the promotion.
5. Close an outlet that runs at a loss for five consecutive days, adding a notification that this has happened to the events
6. Display details of the restaurant which, during a day, was either the most profitable, the most visited or the one with the highest reputation rating
7. Prevent a new outlet being opened within a certain distance of another outlet, or another outlet of the same type
8. Redesign `Company` to be an abstract class, with categories of company each being a subclass
9. Generate a random budget and store it as an attribute within each `Household` object; a household that eats out will only eat out with a company whose prices are within their budget
10. Incorporate weather into the simulation; it can rain at random, in which case the probabilities of eating out are all halved, and the presence of rain is indicated within the events
11. Generate random events, such as power cuts, fuel shortages and festivals, each of which can have an impact on the probability of each house eating out, daily costs, etc.
12. Incorporate a text file from which initial companies and outlets are created (where they differ from the default companies), rather than having the user manually enter them each time the simulation is run. Incorporate the default companies into this text file too. It should be a single file for all companies, ideally using JSON format.
13. Add a feature as an extension to programming Task 8 in which capacity was used to limit visits. You should automatically expand the capacity and/or max capacity at the day end if too many visits were received.
14. Create a feature that will report on the amount of capacity used for each restaurant and close unnecessary outlets to maximise visits to other outlets.
15. Change the basic premise of the simulation so that instead of households choosing a company and then visiting the nearest outlet, there is a more complex formula that reflects reality, e.g. the reputation of the company and the distance of the nearest outlet. A very challenging extension to this would then be to calculate where the best place for a new outlet would be.