Supermarket Sweep

Adam Profili, Brynn Schneberger

4/25/2021

Project Overview

A contestant in a gameshow has a default of 90 seconds to make their way around a grocery store, collect up to a default of 15 items in their cart, and return to the start. Their score is the total cost of the items in their cart. Items are spread around the following aisles, and each have specific prices and locations.

(a) asks:

For each pair of items $i, j \in [\![1, 56]\!]$, compute the shortest time d_{ij} to go from item i to item j, and store these values. You may not use any optimization software to answer this question. Compute also the shortest time to travel between the start/end location and any item.

(a) solution:

```
# initialize an empty two-dimensional list
d = [[0 for i in range(len(item_list))] for i in range(len(
# iterate through all item objects twice for all pairings
for i in range(len(item_list)):
    for j in range(len(item_list)):
        item_i = item_list[i]
        item_j = item_list[j]

        # if item i and j share an aisle:
        if item_i.x == item_j.x:
            d[i][j] = abs(item_i.y - item_j.y) / 10
        # if they don't share an aisle:
        else:
            dist_x = abs(item_i.x - item_j.x)
            dist_y = min(((110 - item_i.y) + (110 - item_j.
            d[i][j] = (dist_x + dist_y) / 10
        # add 2 seconds to the time between items if the t
        # two seconds for the contestant to pick up the it
```

(b) asks: