Yuan Qin
yqin7@uci.edu May.6th

Part 4c

First Part:
My implementation works for all orders under 10 items while problem occurs while the item equals or exceeds 10. For item with 10 items, some orders function well such as order #43 shown in the next section and some would fail like order #5. For it depends on the average case cost of this algorithm, conclusion can't be made given so few test cases, but generally speaking, large orders with items concentrated in same location would luckily survived the test because of the almost best case scenario that their child cost are nearly all 0. Some will run in to infinite loop because of memory limitation.

eg. 10items order #5
As node number exceeds 1024(in second snapshot), where the level close to 10, the dictionary for saving the node information is distorted because of memory. So it will never step out the while loop by returning while the level reaches order length.

```
Computing shortest distance to travel using branch and bound......
initial matrix
[inf, 21, 17, 17, 23, 21, 17, 21, 15, 17, 19]
[10, inf, 4, 4, 0, 0, 4, 0, 6, 4, 6]
[6, 4, inf, 0, 4, 4, 0, 4, 4, 4, 6]
[6, 4, 0, inf, 4, 4, 0, 4, 4, 4, 6]
[23, 2, 6, 6, inf, 2, 6, 2, 8, 6, 8]
[10, 0, 4, 4, 0, inf, 4, 0, 6, 4, 6]
[6, 4, 0, 0, 4, 4, inf, 4, 4, 4, 6]
[10, 0, 4, 4, 0, 0, 4, inf, 6, 4, 6]
[15, 6, 4, 4, 8, 6, 4, 6, inf, 6, 8]
[17, 4, 4, 4, 6, 4, 4, 4, 6, inf, 4]
[19, 6, 6, 6, 8, 6, 6, 6, 8, 4, inf]
initial cost 42
new node  1 ----------------------------------------
```

```
Computing shortest distance to travel using branch and bound......
initial matrix
[inf, 21, 17, 17, 23, 21, 17, 21, 15, 17, 19]
[10, inf, 4, 4, 0, 0, 4, 0, 6, 4, 6]
[6, 4, inf, 0, 4, 4, 0, 4, 4, 4, 6]
[6, 4, 0, inf, 4, 4, 0, 4, 4, 4, 6]
[23, 2, 6, 6, inf, 2, 6, 2, 8, 6, 8]
[10, 0, 4, 4, 0, inf, 4, 0, 6, 4, 6]
[6, 4, 0, 0, 4, 4, inf, 4, 4, 4, 6]
[10, 0, 4, 4, 0, 0, 4, inf, 6, 4, 6]
[15, 6, 4, 4, 8, 6, 4, 6, inf, 6, 8]
[17, 4, 4, 4, 6, 4, 4, 4, 6, inf, 4]
[19, 6, 6, 6, 8, 6, 6, 6, 8, 4, inf]
initial cost 42
new node  1 ----------------------------------------
```

Second Part:

- For one of your smaller orders (4 items):
  - Order # 4

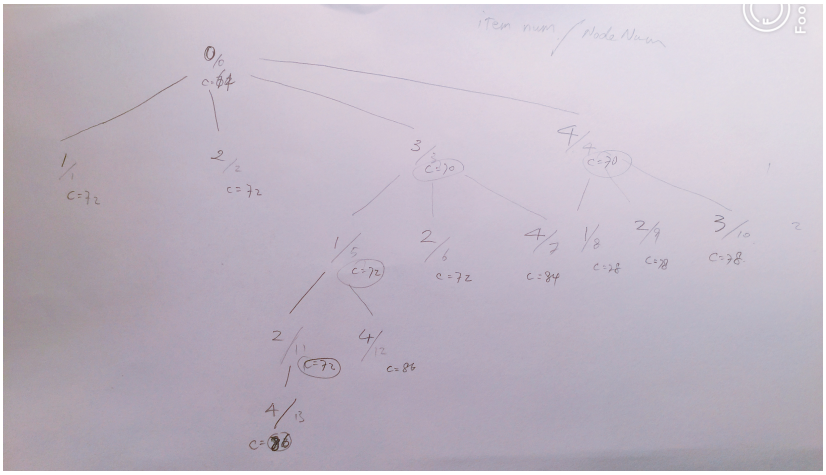
Nearest neighbor:

```
one optimized: [284905, 383599, 381992, 89769]
Minimum travel distance:  70 ,in order of:  start from  (0, 0) [284905, 383599, 381992, 89769] , end at  (0, 20)
go to shelf: [5, 7] on location: [11, 15] pick up item: 284905 , then
go to shelf: [7, 5] on location: [15, 11] pick up item: 383599 , then
go to shelf: [7, 5] on location: [15, 11] pick up item: 381992 , then
go to shelf: [10, 5] on location: [21, 11] pick up item: 89769 , then
drop off at: [0, 20]
Nearest neighbor cost: 0.21373510360717773
```

Branch and bound:

```
The order ready to pick:  [383599, 381992, 89769, 284905]

Distance for one order without optimization 62
Please choose algorithm, 1 for nearest neighbor, 2 for branch and bound:2
Computing shortest distance to travel using branch and bound......
initial matrix
[inf, 25, 25, 31, 25]
[8, inf, 0, 2, 8]
[8, 0, inf, 2, 8]
[31, 8, 8, inf, 14]
[25, 8, 8, 14, inf]
final matrix
[inf, 0, 0, 0, inf]
[8, inf, 0, 2, inf]
[inf, inf, inf, inf, inf]
[6, 0, 0, inf, inf]
[inf, 0, 0, 0, inf]
Minimum travel distance:  86 ,in order of:  start from  (0, 0) [89769, 383599, 381992, 284905] , end at  (0, 20)
go to shelf: [10, 5] on location: [21, 11] pick up item: 89769 , then
go to shelf: [7, 5] on location: [15, 11] pick up item: 383599 , then
go to shelf: [7, 5] on location: [15, 11] pick up item: 381992 , then
go to shelf: [5, 7] on location: [11, 15] pick up item: 284905 , then
drop off at: [0, 20]
Branch and bound cost: 0.0029418468475341797
```

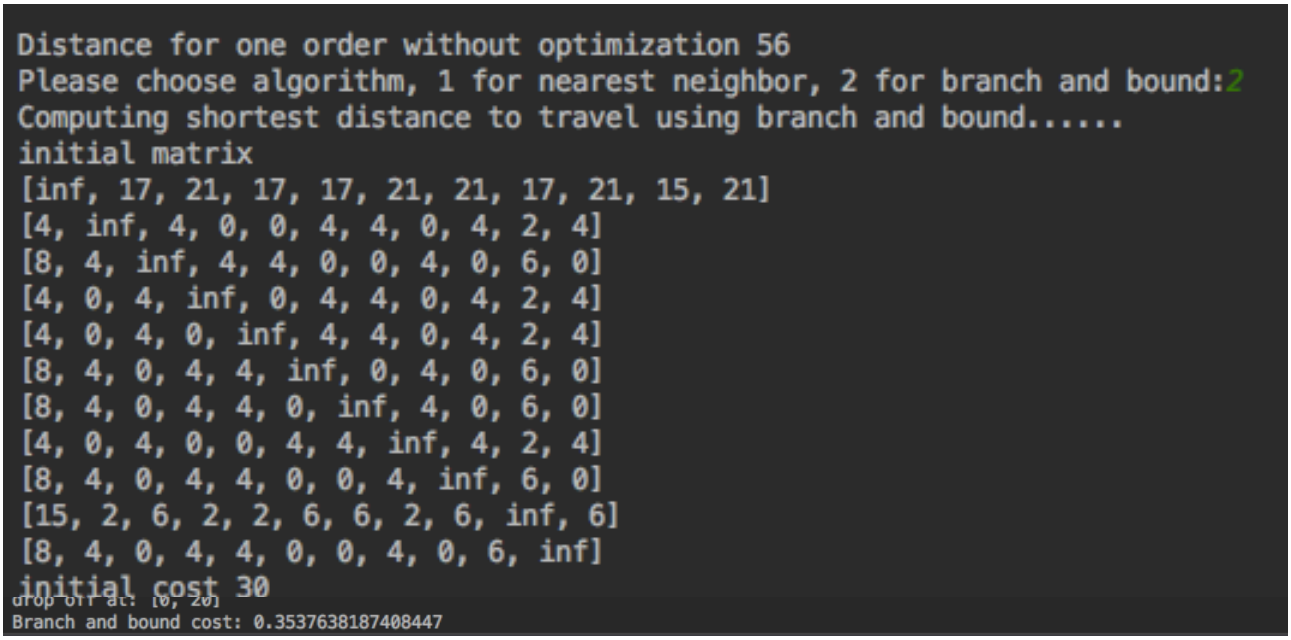


Optimized distance:60 which is better than nearest neighbor and original unoptimized version distance.

- For one of your larger orders (10 or 10+ items):

```
one optimized: [1101259, 5735, 180023, 32895, 111873, 392000, 350562, 22234, 321099, 499508]
Minimum travel distance:  40 ,in order of:  start from  (0, 0) [1101259, 180023, 32895, 111873, 5735, 350562, 22234, 321
go to shelf: [4, 3] on location: [9, 7] pick up item: 1101259 , then
go to shelf: [4, 4] on location: [9, 9] pick up item: 180023 , then
go to shelf: [4, 4] on location: [9, 9] pick up item: 32895 , then
go to shelf: [4, 4] on location: [9, 9] pick up item: 111873 , then
go to shelf: [4, 4] on location: [9, 9] pick up item: 5735 , then
go to shelf: [5, 5] on location: [11, 11] pick up item: 350562 , then
go to shelf: [5, 5] on location: [11, 11] pick up item: 22234 , then
go to shelf: [5, 5] on location: [11, 11] pick up item: 321099 , then
go to shelf: [5, 5] on location: [11, 11] pick up item: 499508 , then
go to shelf: [5, 5] on location: [11, 11] pick up item: 392000 , then
drop off at: [0, 20]
Nearest neighbor cost: 1.344414234161377
```

- Order number#43,10 items
  - Nearest neighbor
  - Branch and bound



Upper bound of the last node of cost is 50 which is lower than the unoptimized version. And the optimized distance is 50, which is not as good as nearest neighbor in this case while the start point and end point are not the same. So traveling to the last point would significantly contribute to the difference of the total distance as it would not function exactly as the B and B problem. But the time cost is much better than brute force.